

# Inhoud

<b>1</b>	<b>Kennismaken met Vue.js</b>	<b>1</b>
	<b>Wat is Vue.js?</b>	<b>2</b>
	Libraries en frameworks	2
	Omschrijving van Vue.js	4
	Progressieve ontwikkeling	4
	Vue op internet	6
	Geen groot bedrijf	6
	7	
	<b>Populariteit van Vue.js</b>	<b>8</b>
	Github sterren	8
	Npm downloads	9
	Google Trends	10
	<b>Wie gebruiken Vue.js?</b>	<b>11</b>
	<b>Het Vue.js-ecosysteem</b>	<b>12</b>
	<b>Architectuur van Vue-applicaties</b>	<b>13</b>
	Single Page Application	14
	Vue-begrippen	16
	<b>De Vue-instantie</b>	<b>16</b>
	De hoofdcomponent in Vue 2 en in Vue 3	18
	<b>Vue-apps met routing</b>	<b>19</b>
	<b>Boomstructuur van componenten</b>	<b>20</b>
	De boomstructuur visueel maken	20
	<b>Benodigde voorkennis</b>	<b>21</b>
	Tips voor meer lezen	23
	<b>Waarom een boek?</b>	<b>23</b>
	<b>De ontwikkelomgeving inrichten</b>	<b>24</b>
	Editor en browser	24
	Node.js	25
	Vue CLI	27
	Vue.js devtools	29
	<b>Oefenbestanden downloaden</b>	<b>30</b>
	<b>Conclusie</b>	<b>31</b>
	<b>Praktijkoefeningen</b>	<b>32</b>

<b>2</b>	<b>Hello World in Vue.js</b>	<b>37</b>
	<b>Mogelijkheden voor Vue-projecten</b>	<b>38</b>
	<b>Vue CLI installeren</b>	<b>39</b>
	<b>Nieuw project starten en draaien</b>	<b>40</b>
	<b>Zijstap – de grafische interface voor Vue CLI</b>	<b>45</b>
	Vue Project Manager	45
	Mogelijkheden van Vue ui	46
	Vue ui stoppen	48
	<b>Het project openen en aanpassen</b>	<b>49</b>
	Component aanpassen	50
	<b>Theorie – de bestandsstructuur verkennen</b>	<b>51</b>
	Mappen	51
	<b>De rol van package.json</b>	<b>53</b>
	<b>Overige belangrijke bestanden</b>	<b>55</b>
	Main.js	55
	App.vue	56
	HelloWorld.vue	58
	<b>Verschillende bestandstypen?</b>	<b>59</b>
	Alles in één bestand	60
	<b>Nieuwe componenten toevoegen</b>	<b>60</b>
	Export default	62
	De component insluiten in de applicatie	63
	Naamgeving van componenten	65
	Slots	65
	Conclusie	65
	<b>Samenvatting</b>	<b>66</b>
	<b>Praktijkoefeningen</b>	<b>68</b>
<b>3</b>	<b>Componenten en databinding</b>	<b>71</b>
	<b>Single file components en globale componenten</b>	<b>72</b>
	Single file components	72
	Globale componenten	73
	<b>Zijstap – Bootstrap toevoegen</b>	<b>74</b>
	Bootstrap installeren	75
	Standaardstijlen verwijderen	75
	<b>Dit gaan we maken: VacationPicker</b>	<b>76</b>
	Eisen aan de applicatie	76
	<b>Stap 1 – Het gegevensbestand maken</b>	<b>77</b>
	<b>Stap 2 – Nieuwe component maken</b>	<b>79</b>
	<b>Stap 3 – gegevens in de applicatie importeren</b>	<b>80</b>
	<b>Stap 4 – Gegevens binden; de directive v-for</b>	<b>81</b>
	Directives	81
	Template uitbreiden	82

<b>Stap 5 – App.vue aanpassen</b>	<b>83</b>
Index gebruiken	85
<b>Meer voorbeelden van databinding</b>	<b>86</b>
<b>Databinding met v-bind</b>	<b>87</b>
Attributen title en id dynamisch maken	88
Sleutelwaarde opgeven bij v-for	89
Speciaal attribuut key	90
<b>Korte notatie voor v-bind</b>	<b>91</b>
Geen interpolatie binnen attributen	92
<b>Gebeurtenisbinding met v-on</b>	<b>92</b>
Counter declareren	93
Korte notatie voor v-on	94
<b>Funcies aanroepen met v-on</b>	<b>94</b>
ES6-notatiewijze voor methoden	96
<b>Samenvatting</b>	<b>97</b>
<b>Praktijkoefeningen</b>	<b>98</b>
<b>4 Meer over componenten</b>	<b>103</b>
<b>Berekende eigenschappen (computed properties)</b>	<b>104</b>
Betere prestaties	105
Component aanpassen met berekende eigenschap	106
View aanpassen	108
Functie versus eigenschap	108
<b>V-if gebruiken</b>	<b>109</b>
v-if versus v-show	110
<b>Binden aan afbeeldingen</b>	<b>111</b>
Webpack en require()	112
getImgUrl()	113
<b>Mixins gebruiken</b>	<b>113</b>
Kenmerken van mixins	115
Afbeelding laden als mixin	115
Mixin toevoegen aan component	116
Regels voor mixins	117
<b>Lifecycle hooks</b>	<b>118</b>
Voorbeeld lifecycle hook	120
Typisch gebruik van enkele lifecycle hooks	121
<b>Werken met v-model</b>	<b>122</b>
V-model voor tekstinvuervelden	123
Nieuwe array vullen	123
Correcte waarden	126
<b>V-model bij een keuzelijst</b>	<b>126</b>
<b>Samenvatting</b>	<b>129</b>
<b>Praktijkoefeningen</b>	<b>130</b>

<b>5</b>	<b>Communicatie tussen componenten</b>	<b>135</b>
	<b>Werken met meerdere componenten</b>	<b>136</b>
	Voordelen van werken met componenten	136
	<b>De component CountryDetail maken</b>	<b>137</b>
	Stap 1 – Een nieuwe component maken en toevoegen	137
	De component CountryDetail	138
	Stap 2 – Kindcomponent voorbereiden om gegevens te ontvangen: props	139
	Props schrijven	139
	De component uitbreiden	139
	Stap 3 – De oudercomponent bijwerken	141
	Stap 4 – Logica verplaatsen	142
	Mixin toevoegen	143
	<b>Props typeren en valideren</b>	<b>144</b>
	Objectnotatie voor props	144
	Verplichte props	145
	Verkeerd type voor props	146
	Validatiefuncties	146
	<b>Werken met custom events</b>	<b>148</b>
	Events verzenden met this.\$emit()	149
	Een gebeurtenis opvangen	149
	Een waardering verzenden	150
	Aangeven dat event wordt verzonden	151
	De gebeurtenis verwerken in de oudercomponent	152
	Samenvatting	153
	<b>Inhoud hergebruiken met slots</b>	<b>154</b>
	Een accordeonstructuur maken	155
	De component CollapsibleSection.vue	155
	<b>Animaties met het element &lt;transition&gt;</b>	<b>157</b>
	Enter- en leave-overgangen	158
	De HTML uitbreiden	159
	De CSS-klassen schrijven	159
	<b>Samenvatting</b>	<b>160</b>
	<b>Praktijkoefeningen</b>	<b>161</b>
<b>6</b>	<b>Werken met de router</b>	<b>163</b>
	<b>Kennismaken met routing</b>	<b>164</b>
	Single page application of SPA	164
	Vue Router	165
	Architectuur bij routing	167
	<b>Routing toevoegen aan een bestaande app</b>	<b>169</b>
	Stap 1 – Routing installeren	170
	Stap 2 – Een directory maken voor de router	171
	Stap 3 – De routetabel definiëren	172

De homepage	173
Stap 4 – Routes toevoegen aan de Vue-configuratie	173
Stap 5 – Vue vertellen waar de inhoud getoond moet worden	174
Stap 6 – Componenten maken	174
Stap 7 – De applicatie testen	175
<b>HTML5-modus inschakelen</b>	<b>177</b>
<b>Een hoofdmenu maken – koppelen naar pagina's</b>	<b>178</b>
Geen <a href> meer	178
<router-link> gebruiken	179
<b>Actieve links markeren</b>	<b>182</b>
Vue 2.x: exacte links markeren	183
<b>Navigeren via code</b>	<b>186</b>
Naar de detailpagina	186
Stap 1 – Route toevoegen	187
Stap 2 – Code toevoegen	187
<b>Dynamische routes met routeparameters</b>	<b>189</b>
Stap 1 – Dubbele punt voor dynamische parameters	189
Stap 2 – CountryDetail aanpassen	189
Stap 3 – Routerlink aanpassen	190
<b>Het juiste land ophalen</b>	<b>192</b>
<b>Het verschil tussen \$router en \$route</b>	<b>193</b>
<b>Lazy loading</b>	<b>194</b>
Voorbeeld van lazy loading	194
<b>Meer over routing</b>	<b>196</b>
<b>Samenvatting</b>	<b>196</b>
<b>Praktijkoefeningen</b>	<b>198</b>
<b>7 State management: vuex</b>	<b>201</b>
<b>Wat is state management?</b>	<b>202</b>
Props en events	202
Store	203
Vuex	203
Data in één richting	204
Single source of truth	205
<b>Concepten bij stores</b>	<b>206</b>
<b>Kennismaken – een tellerapplicatie</b>	<b>207</b>
Stap 1 – Nieuwe applicatie maken	207
Stap 2 – De store instellen	209
Stap 3 – Gegevens aan de store toevoegen	211
Stap 4 – Mutations toevoegen	212
Stap 5 – Actions toevoegen	213
Stap 6 – Component bijwerken (HTML)	214
Stap 7 – Component bijwerken (JavaScript)	215

Counter ophalen	216
Applicatie testen	217
<b>State in een andere component gebruiken</b>	<b>218</b>
<b>State management met complexe objecten</b>	<b>219</b>
API's op internet	220
<b>Communiceren met externe API's</b>	<b>221</b>
<b>Axios gebruiken</b>	<b>222</b>
Axios installeren en gebruiken	223
<b>Axios toevoegen aan de store</b>	<b>224</b>
Stap 1 – De state aanpassen	225
Stap 2 – Mutations toevoegen	225
Stap 3 – Actions toevoegen	226
Stap 4 – Nieuwe component maken	228
Logica voor de component	229
Stijl voor de vlag toevoegen	230
Routing aanpassen	231
<b>De applicatie testen</b>	<b>231</b>
Een fout simuleren	232
<b>Getters gebruiken</b>	<b>234</b>
De store aanpassen	234
Klassieke notatie	235
Nieuwe component maken	235
Countries.vue aanpassen	236
Router aanpassen	237
<b>Werken met store modules</b>	<b>238</b>
Directory /store/modules maken	239
De hoofdmodule bijwerken	241
De componenten bijwerken	242
Notatie van getter met modules	243
State management in de Vue DevTools	244
<b>Meer over state management</b>	<b>245</b>
<b>Samenvatting</b>	<b>246</b>
<b>Praktijkoefeningen</b>	<b>247</b>
<b>8 Vue-applicaties uitrollen naar de server</b>	<b>253</b>
<b>Productiebuild met Vue CLI</b>	<b>254</b>
Build met Vue CLI	255
<b>Een pad aangeven in de productiebuild</b>	<b>256</b>
Vue.config.js	256
<b>Distribueren naar een productieserver</b>	<b>257</b>
<b>Publiceren naar Netlify</b>	<b>257</b>
Aanmelden bij Netlify	259
Continuous deployment	260

Slepen-en-neerzetten	261
De site publiceren	262
De site eigen naam geven	264
Een nieuwe versie publiceren	265
Redirect naar de homepage instellen	266
Redirect altijd meesturen	266
<b>Tot slot</b>	<b>267</b>
<b>Samenvatting</b>	<b>269</b>
<b>Praktijkoefeningen</b>	<b>269</b>
<b>Index</b>	<b>271</b>

# Kennismaken met Vue.js

*Van enkele eenvoudige HTML-pagina's in de jaren 1990 is het web uitgegroeid tot een van de meest complexe systemen die we kennen. Internet wordt gebruikt voor eenvoudige hobbysites, maar ook voor onlinebetaalsystemen, crm- en klantbeheersystemen, verzekerings- en schademodelen, sociale media en ontelbare andere zaken. Vue.js is een framework voor het programmeren van dergelijke ingewikkelde webapps. In Vue.js werkt u niet meer met losse HTML-pagina's, maar met webcomponenten. De componenten werken met elkaar samen en vormen zo een complete webapplicatie. Vue.js maakt het mogelijk in hoge mate modulair te programmeren. JavaScript is de programmeertaal die hiervoor wordt ingezet. Dit boek geeft een inleiding op al deze zaken. Na afloop kunt u vol vertrouwen aan de slag met uw eigen Vue-applicaties.*

## In dit hoofdstuk:

*Wat Vue.js is, en wat het niet is.*

*Concepten en kenmerken van Vue.*

*Benodigde voorkennis en software.*

*Wat hebt u nodig? De werkomgeving inrichten.*

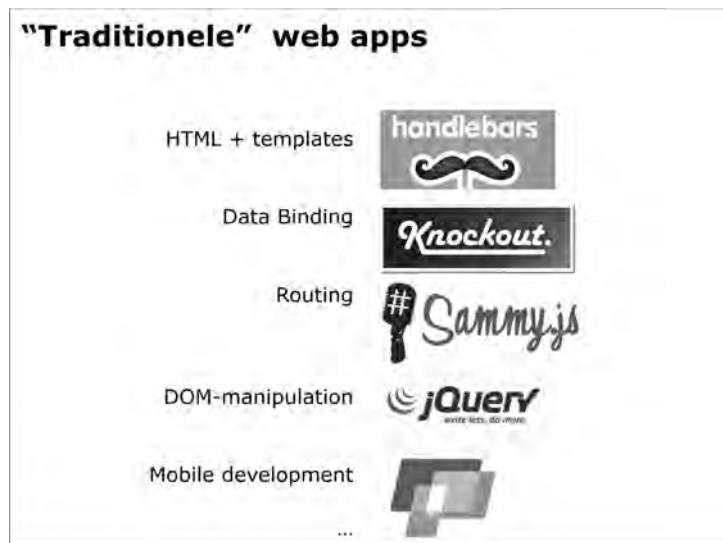


## Wat is Vue.js?

Het aloude HTML is prima om tekst en afbeeldingen te tonen in de browser, maar is oorspronkelijk nooit ontwikkeld voor het maken van dynamische webapplicaties. Voor dat doel is JavaScript rond 1995 ontworpen. Samen met CSS (dat rond dezelfde tijd opkwam) behoort JavaScript op dit moment tot de basisvaardigheden van elke webdeveloper. JavaScript was in het begin lastig te leren. Verschillende browsers hadden hun eigen ideeën over de implementatie van JavaScript.

### Libraries en frameworks

Pas sinds de opkomst van aanvullende bibliotheken zoals jQuery in 2006 heeft JavaScript een enorme vlucht genomen. Behalve jQuery zijn tal van andere bibliotheken (*library's*) ontwikkeld, elk met hun eigen doel. Er zijn bibliotheken voor DOM-manipulatie



**Afbeelding 1.1** In traditionele webapplicaties neemt elke bibliotheek een van de eisen die aan de applicatie wordt gesteld voor zijn rekening. Er is kans op onderlinge incompatibiliteit.

(zoals jQuery), routing (sammy.js), databinding (backbone, knockout.js), werken met datums en tijden (moment.js) en nog veel meer.

Maar Vue is geen bibliotheek zoals de hiervoor genoemde. Vue is een compleet *framework* voor het realiseren van clientside webapplicaties. Een framework vervult *alle* taken die aan moderne webapplicaties worden gesteld.

- **Bibliotheken** Als we de zaken erg vereenvoudigd voorstellen, kun je zeggen dat bibliotheken in het algemeen één ding heel goed doen.
- **Frameworks** Een framework zoals Vue.js, biedt oplossingen voor alle niveaus van applicatieontwikkeling. Van het structureren en binden van data tot http-communicatie met webserver, het verwerken van geretourneerde gegevens in de HTML-template en het maken van herbruikbare componenten.

Bibliotheken kunnen in het algemeen gecombineerd worden in een project om gezamenlijk het beste resultaat te bereiken. Bij frameworks maak je daarentegen één keuze en bouw je de app volgens de richtlijnen en kenmerken van het gekozen framework.



### Geen combinaties

We zullen in de praktijk bijvoorbeeld nooit zien dat een app *zowel* Vue.js als React (een alternatief framework) gebruikt. Ook combinaties van Vue met Angular of Polymer (andere alternatieven) komen niet voor. U bouwt de site ofwel in Vue, ofwel in Angular. Niet in beide.

---



**Afbeelding 1.2** *In een framework als Vue.js, maar ook in alternatieven zoals Angular, of React, worden alle taken van een applicatie gebundeld en geïntegreerd aangeboden. De leercurve is steiler, maar het resultaat is een consistentere en eenvoudigere (en dus goedkoper te onderhouden) applicatie.*

## Omschrijving van Vue.js

Vue.js wordt op de officiële site ([www.vuejs.org](http://www.vuejs.org)) omschreven als:

*The progressive JavaScript framework*

Hiermee wordt bedoeld dat, hoewel Vue.js als één framework wordt aangeboden, het in beginsel een kleine kern heeft. Deze kern kan naar behoeven worden uitgebreid ('progressief'). Hiervoor gebruikt Vue aanvullende bibliotheken. U mag ze inzetten, maar het hoeft niet. De kernbibliotheek is voornamelijk gericht op de weergavelaag (*view layer*) van applicaties en richt zich met name op HTML-templates en data die daarin wordt weergegeven.

## Progressieve ontwikkeling

Als de applicatie meer nodig heeft, zoals routeren naar verschillende componenten of het communiceren met een backend, kunt u daarvoor aanvullende bibliotheken inzetten. Deze zijn vaak speciaal voor Vue.js ontworpen. Ze zijn echter niet verplicht om een werkende Vue-applicatie te maken.

Soms zijn de aanvullingen ook gemaakt door andere ontwikkelaars. Ze zijn geadopteerd in het Vue.js-framework. Bekende aanvullende bibliotheken zijn bijvoorbeeld:

- **Vue Router** Voor het routeren in applicaties van de ene component naar de andere component, waarbij de URL in de adresbalk van de browser wordt aangepast;
- **Axios** Voor het communiceren met API's om gegevens/data op te halen en vervolgens te tonen in de applicatie;
- **Vuex** Een oplossing voor state management, om alle data in de applicatie op een centrale plek te beheren;
- **Vue CLI** De opdrachtregelomgeving voor het instellen van projecten, het regelen van afhankelijkheden (*dependencies*), testen en meer;
- **Vue.js devtools** Een uitbreiding voor Google Chrome en Firefox die het inspecteren en debuggen van Vue-applicaties vereenvoudigt.

Er zijn er uiteraard meer, maar dit zijn de bibliotheken waar u als Vue.js-ontwikkelaar ongetwijfeld mee te maken krijgt. Lees eventueel meer over de kenmerken van Vue.js op [v3.vuejs.org/guide/](https://v3.vuejs.org/guide/).

Vue.js is een compleet clientside framework. Het is in JavaScript geschreven en draait ook volledig in de browser. In de ideale situatie is de app compleet losgekoppeld van de server en database waar de gegevens vandaan komen. Alle communicatie vindt plaats via http-aanroepen. Uiteraard gaan we hier later in dit boek nog dieper op in. Om Vue.js-applicaties te maken hebt u geen enkele kennis nodig van een backendtechnologie zoals PHP, Java of C#.



**Afbeelding 1.3** De homepage van Vue op [vuejs.org](https://vuejs.org). Start hier voor officiële downloads, documentatie en meer.



### Vue of Vue.js?

De begrippen Vue en Vue.js worden door elkaar gebruikt. De officiële en volledige naam is Vue.js, maar in het algemeen spraakgebruik wordt vaak kortweg gesproken van Vue. Vue wordt uitgesproken als het Engelse *view*. In dit boek bedoelen we er hetzelfde mee: uw applicatie die draait in de browser en is geschreven in het framework Vue.js.

## Vue op internet

De homepage van Vue is te vinden op **[vuejs.org](https://vuejs.org)**. Hier kunt u artikelen lezen, online lessen volgen, deelnemen aan discussies en zelfs Vue-T-shirts en andere merchandise aanschaffen. Ook is dit het startpunt voor de officiële documentatie. Kies hiervoor de optie **Learn, Guide** uit het hoofdmenu.



## Verschillende versies

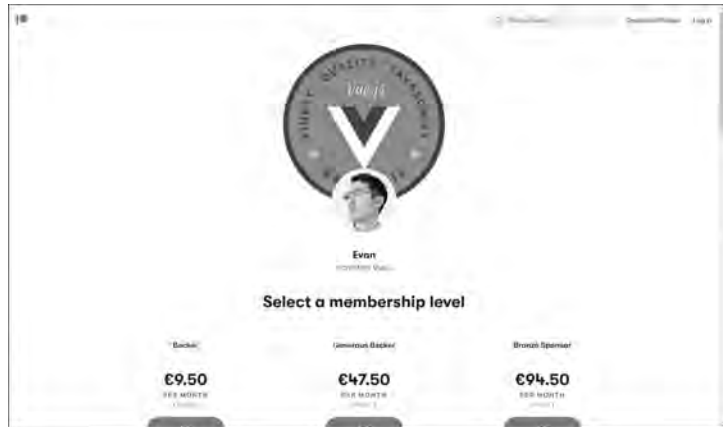
Op het moment van schrijven verwijst de documentatie op **vuejs.org** (nog) naar de documentatie van Vue 2. Maar in het najaar van 2020 is Vue 3 verschenen. Hierover schrijven we in dit boek. De documentatie vindt u op **v3.vuejs.org**. In de toekomst zal Vue 2 steeds ouder worden en verwachten we dat u ‘vanzelf’ bij de beschrijving van Vue 3 uit zult komen als u het hoofdadres invoert. Maar check het altijd wel even.

## Geen groot bedrijf

In tegenstelling tot andere frameworks staat achter Vue.js geen groot bedrijf. Het concurrerende framework Angular is gemaakt (en wordt financieel onderhouden) door Google. React is afkomstig uit de stal van Facebook. TypeScript is oorspronkelijk gemaakt door Microsoft.

Maar Vue.js is in beginsel een eenmansproject van Evan You (@youyuxi op Twitter). Wel werken er op dit moment veel meer mensen aan Vue.js. Dit is mogelijk gemaakt door middel van donaties en contributies uit de opensourcegemeenschap. Vue.js is gestart in 2014 en op het moment van schrijven dus bijna zeven jaar oud.

Maakt uw bedrijf of organisatie ook gebruik van Vue.js, overweeg dan een donatie aan het Vue-project via Patreon. De site is te vinden op **www.patreon.com/evanyou**. Zo ondersteunt u de makers en kunnen zij zich bezig houden met verbeteringen, het repareren van fouten en meer.



**Afbeelding 1.4** *Vue.js wordt volledig gefinancierd door donaties uit de opensourcegemeenschap. Het maandbedrag wordt verdeeld onder alle ontwikkelaars.*

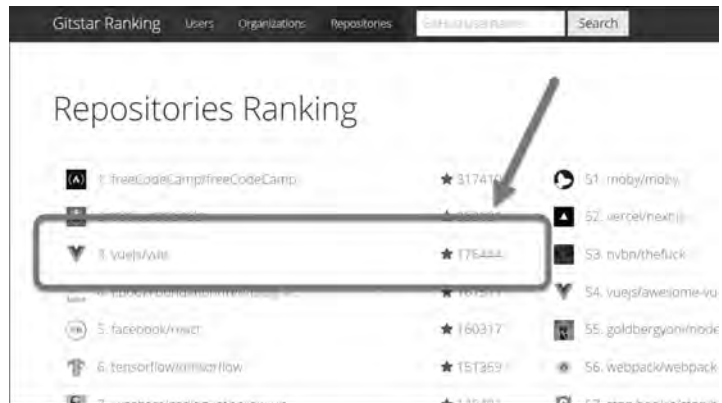
## Populariteit van Vue.js

Er zijn verschillende manieren om de populariteit te meten, en geen van de manieren is de enige juiste wijze. Gecombineerd laten ze echter het volgende beeld zien. Op het moment van schrijven van dit boek (voorjaar 2021) is Vue.js in omvang en populariteit het derde framework. React is het grootst/populairst, gevolgd door Angular. De frameworks Polymer en Aurelia worden verreweg het minst gebruikt en zijn niet meegenomen in de afbeeldingen.

### Github sterren

Op **github.com** (waar alle populaire opensourcerepositories zijn gehost) kunnen ontwikkelaars een ster geven aan een repository om zo een lijstje met favoriete repositories bij te houden. Hierbij zien we dat Vue.js het populairst is, met bijna 180.000 sterren:

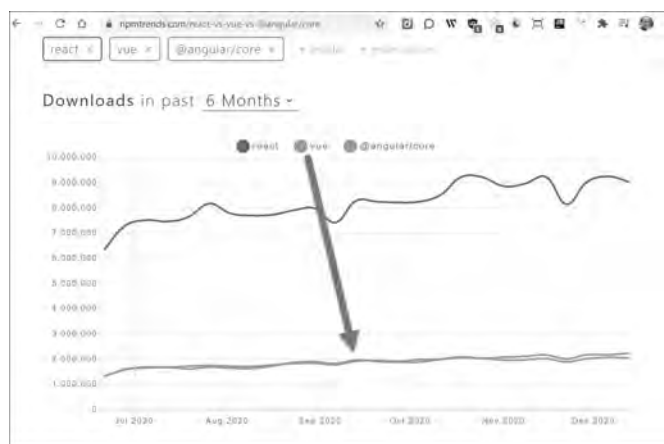
- 1 **Vue.js** 177.000 sterren
- 2 **React** 160.000 sterren
- 3 **Angular** 68.000 sterren



**Afbeelding 1.5** Vue.js staat wereldwijd op de derde plaats van populaire repositories. Het is het populairste frontend framework bij ontwikkelaars. Bekijk zelf de huidige positie op [gitstar-ranking.com/repositories](https://gitstar-ranking.com/repositories).

## Npm downloads

Wanneer de opdracht `npm install` voor een project wordt uitgevoerd, wordt de package opgehaald uit het npm-register. Dit zult



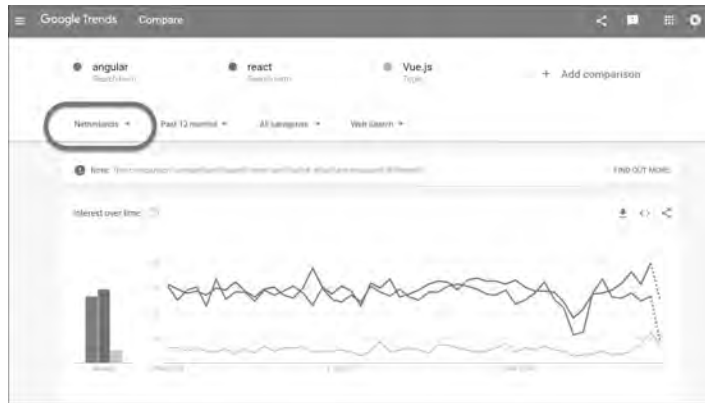
**Afbeelding 1.6** Een vergelijking in aantallen downloads voor React, Angular en Vue bij [npm-trends.com](https://npm-trends.com).



u zelf nog talloze malen doen als u de oefeningen in dit boek volgt of de voorbeeldprojecten download en uitvoert. De website **npm trends.com** houdt bij hoe vaak een bepaalde package wordt geïnstalleerd. Hier is duidelijk te zien dat React het meest worden geïnstalleerd. Angular en Vue delen de tweede en derde plaats.

## Google Trends

Voor een indruk van de populariteit van Vue kijken we tot slot naar Google Trends (**trends.google.com**). Hiermee is te onderzoeken hoe vaak een zoekvraag wordt gebruikt in relatie tot andere zoekvragen. Ook dan zien we min of meer hetzelfde beeld. React en Angular staan op de plekken één en twee, Vue is nummer drie. Boze tongen zullen overigens beweren dat React bovenaan staat omdat het zo verduiveld moeilijk is om te leren en er dus veel zoekvragen voor nodig zijn. Vue.js daarentegen is een stuk eenvoudiger en staat dus logischerwijs op plek drie...



**Afbeelding 1.7** De interesse bij Google voor de drie grote frameworks, op een rijtje gezet door Google Trends. De regio Netherlands is geselecteerd. Zelf kunt u ook allerlei andere filters toepassen.

Het algemene beeld van al deze vergelijkingen is echter duidelijk. React is het grootste, gevolgd door Angular en Vue. Deze laatste wint echter aanzienlijk aan populariteit en kent een stijgende lijn.

Het is niet ondenkbaar dat in de toekomst React of Angular van de troon wordt gestoten door Vue.

## Wie gebruiken Vue.js?

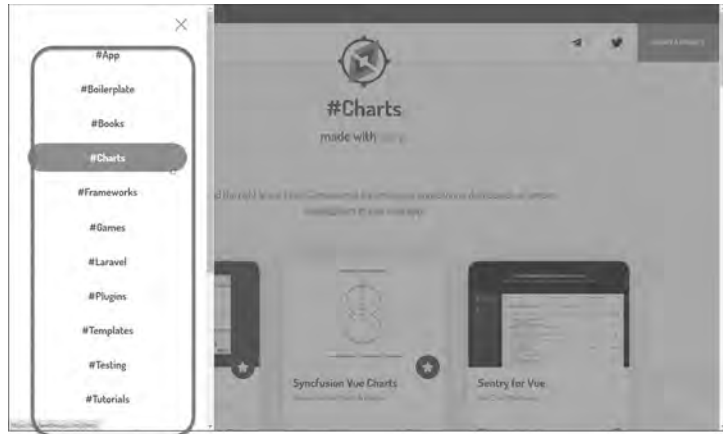
Inmiddels wordt Vue.js door allerlei bedrijven, van klein tot (zeer) groot ingezet. Enkele voorbeelden zijn:

- **Nintendo** Bekend van gameconsoles en allerlei Mario-games.
- **L'Oréal** Bekend van talloze verzorgingsproducten, make-up, stylingadviezen en (veel) meer.
- **Alibaba** Het grote Aziatische e-commercebedrijf.
- **Gitlab** Een verzameling tools voor *software development life-cycles*.
- **Ziggo** Een groot telecombedrijf dat in Nederland (en elders ter wereld) internet, televisie en mobiele telefonie aanbiedt. Vue is dus geschikt voor bedrijven met miljoenen klanten.



Afbeelding 1.8 In Nederland is bijvoorbeeld de consumentensite van Ziggo gemaakt met Vue.js.

Maar Vue wordt niet alleen ingezet bij dergelijke grote bedrijven. Er zijn talloze andere sites, variërend van onlineapplicaties en -boeken tot reisbureaus, contentmanagementsystemen, WordPress-templates en ‘gewone’ websites. Bekijk eventueel een overzicht op [madewithvuejs.com](http://madewithvuejs.com).



**Afbeelding 1.9** *Op [madewithvuejs.com](http://madewithvuejs.com) kunt u meer voorbeelden vinden van sites en applicaties die met Vue zijn gemaakt. In sommige gevallen is ook de broncode beschikbaar.*

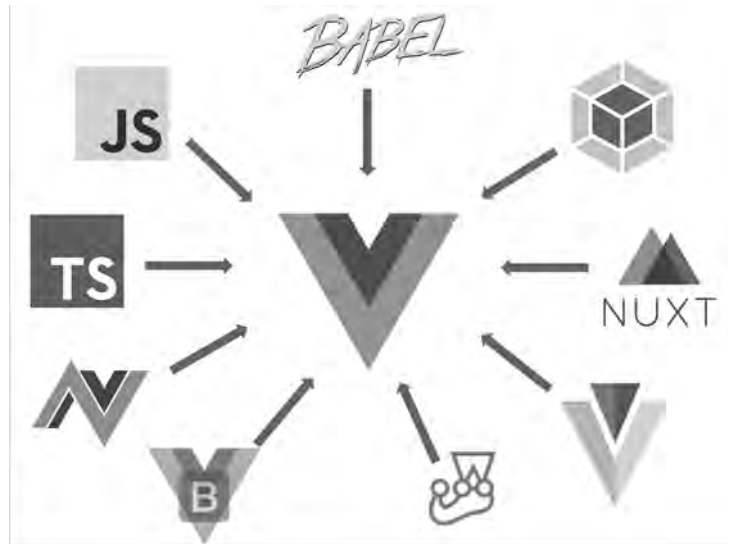
## Het Vue.js-ecosysteem

Vue.js staat echter niet eenzaam in een woestijn, wachtend om gebruikt te worden. Er is een compleet ecosysteem ontwikkeld rondom Vue. We zagen al enkele aanvullende bibliotheken die in combinatie met Vue worden gebruikt, maar er is nog veel meer.

Er zijn aanvullende bibliotheken voor gebruikersinterfaces (bijvoorbeeld `vue-bootstrap` en `vuetify`), frameworks voor applicatieontwikkeling (bijvoorbeeld `Nuxt`) en aanvullende bibliotheken om mobiele applicaties voor iOS en Android mee maken (`vue-native`). Wilt u een eCommerce-applicatie ontwikkelen? Dan is `Vue Storefront` ([www.vuestorefront.io](http://www.vuestorefront.io)) de aangewezen

keuze. U ziet, op alle vlakken in het frontendlandschap is Vue aanwezig.

Uiteraard is Node.js nodig als JavaScript-ontwikkelomgeving en nog veel meer. De afbeelding geeft een indruk van het ecosysteem dat in de loop der jaren is ontwikkeld. Maar ongetwijfeld komen hier nog veel meer onderdelen bij en is dit plaatje moeiteloos uit te breiden.

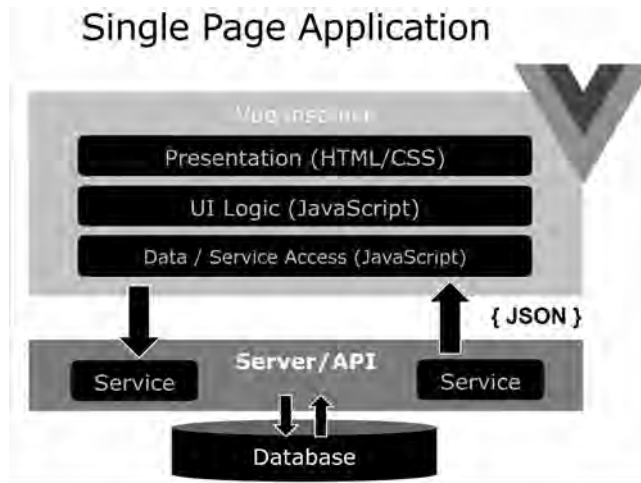


**Afbeelding 1.10** *Er is een compleet ecosysteem rondom Vue ontwikkeld. Dit zijn aanvullende bibliotheken en technieken die goed samenwerken met Vue. Ze zijn niet verplicht (op JavaScript na), maar zorgen er vaak wel voor dat wensen eenvoudiger kunnen worden gerealiseerd.*

## Architectuur van Vue-applicaties

In principe zijn Vue-applicaties toepassingen die volledig zelfstandig in de browser draaien. Er zijn ook varianten, zoals Vue in een standalone app die is gemaakt met Electron, Ionic of NativeScript. Daar gaan we in dit boek echter niet op in. We concentreren ons

op webapplicaties met een architectuur zoals in afbeelding 1.11 te zien is.



**Afbeelding 1.11** De architectuur die we nastreven in Vue-applicaties: de logica- en presentatielaag draaien in de browser, datatoegang en authenticatie draaien op de server.

## Single Page Application

De architectuur die in afbeelding 1.11 wordt getoond, wordt ook wel het principe van *single page applications* genoemd. Dit betekent dat in één pagina (`index.html`) de logica van de applicatie geladen wordt en dat deze pagina steeds zichtbaar is in de browser. Vanuit `index.html` vindt alle navigatie plaats.

De app zelf bestaat natuurlijk uit veel meer dan één bestand. Elke component staat in zijn eigen bestand, er zijn CSS-bestanden, afbeeldingen enzovoort.

- De rol van de browser is:
  - *Presentatielaag* tonen aan de gebruiker, via de view van componenten. Dit is gewoon HTML en CSS zoals u kent, verrijkt met Vue-specifieke toevoegingen.

- *Gebruikersinterfacelogica*, via het script van een view. Hierin staat logica om de gebruikersinterface samen te stellen, events te verwerken zoals muiskliks of het verwerken van Ajax-aanroepen en meer. Het scriptblok van een component wordt ook wel de controller genoemd.
- *Data- en servicetoegang*, oftewel communiceren met een of meer RESTful API's op de server. De API praat vervolgens met de database en retourneert gegevens, het liefst in JSON-formaat. Datatoegang kan rechtstreeks vanuit een component, maar vaker wordt hiervoor een speciale service geschreven of wordt een *state management store* gebruikt. Hier gaan we later in het boek nog op in.
- De rol van de server is:
  - *Databasetoegang* via een API. Oneerbiedig gezegd wordt de rol van de server in moderne SPA-applicaties steeds verder teruggedrongen. Een server is niets meer dan een 'JSON-fabriek die data serveert'.
  - *Authenticatie*. Op het terrein van toegangsrechten is de server nog steeds onontbeerlijk. Immers, de complete Vue-app draait in de browser en is daarmee onveilig. Authenticatie en autorisatie zijn per definitie het domein van de server. Hier moet worden ingelogd met gebruikersnaam en wachtwoord, via sociale netwerken of anderszins. De server moet vervolgens een token of authenticatiecookie uitreiken (afhankelijk van de wijze van beveiliging die door de serverbeheerder is gekozen). De Vue-app is er verantwoordelijk voor dat dit token of de cookie met elk volgend verzoek wordt meegezonden.

In dit boek gaan we niet verder in op het inrichten of beheren van een webserver. In hoofdstuk 7 gaat u wel aan de slag met het communiceren met (bestaande) databases vanuit Vue.