Timothy Markus

t.markus@student.fontys.nl

Abstract

De huidige tijd kent een hoop verschillende frameworks voor front-end development. Maar welke is nu het meest geschikt om een ticketing systeem te bouwen tegen een C# REST API backend? Binnen dit document is verslag gelegd naar de mogelijkheden middels verschillende onderzoeksstrategieën.

O-S71 Onderzoek 2020

Front-end Frameworks

Inhoudsopgave

[Doel van het onderzoek 2](#_Toc56711151)

[Deelvragen 2](#_Toc56711152)

[Aanpak 2](#_Toc56711153)

[Planning 3](#_Toc56711154)

[Welke frameworks en libraries zijn er op dit moment? 4](#_Toc56711155)

[Available product analysis 4](#_Toc56711156)

[Document Analysis 11](#_Toc56711157)

[Conclusie: beschikbare front-end frameworks en libraries. 13](#_Toc56711158)

[Welke frameworks worden gebruikt in reeds bestaande CRM/ticketing systemen? 14](#_Toc56711159)

[Available product analysis 14](#_Toc56711160)

[Overzicht 21](#_Toc56711161)

[Conclusie: Frameworks in bestaande CRM & Ticketing systemen 22](#_Toc56711162)

[Wat maakt een framework geschikt voor een ticketsysteem? 23](#_Toc56711163)

[Explore user requirements 23](#_Toc56711164)

[SWOT analysis, pros and cons 24](#_Toc56711165)

[Functionaliteiten en SWOT tegenover frameworks en libraries 25](#_Toc56711166)

[Conclusie: Frameworks geschikt voor Ticketing systemen 26](#_Toc56711167)

[Welke relevante voordelen hebben bepaalde frameworks? 27](#_Toc56711168)

[Prototyping 27](#_Toc56711169)

[Verwijzingen 28](#_Toc56711170)

Hoofdvraag

Welk front-end framework is het meest geschikt voor het ontwikkelen van een efficiënt ticketing systeem, wanneer deze moet communiceren met een C# REST API backend?

# Doel van het onderzoek

Het doel van het onderzoek is om erachter te komen welke front-end frameworks er zijn en welke van die frameworks nu het meest geschikt is om een front-end voor een ticketing systeem te bouwen. Hieraan hangen al wel een aantal voor gedefinieerde eisen.

* De backend waarmee gecommuniceerd moet worden is een C# REST API.
* Het systeem moet schaalbaar zijn: Er moeten bijvoorbeeld ‘custom’ velden kunnen worden toegevoegd aan klantkaarten / tickets.
* Het onderzoek moet leiden tot een ‘product’ of aantal deel producten, welke moeten worden opgeleverd aan het einde van het semester (18 weken). Hiermee zal het uiteindelijke advies ook ondersteund moeten worden.

Indien mogelijk zal dit onderzoek in een onderbouwd advies resulteren welke de hoofdvraag beantwoord. Dit advies zal worden gebruikt voor de bouw van een ticketing systeem binnen dit semester.

# Deelvragen

* Welke frameworks zijn er op dit moment?
* Welke relevante voordelen hebben bepaalde frameworks?
* Welke frameworks worden gebruikt in reeds bestaande CRM/ticketing systemen?
* Wat maakt een framework geschikt voor een ticketsysteem?

# Aanpak

De hoofdvraag (SMART) is opgedeeld in een aantal deelvragen. Het doel is om de deelvragen doormiddel van triangulatie te beantwoorden. Dit betekend dat voor het beantwoorden van een deelvraag 3 verschillende invalshoeken gebruikt moeten worden. Zodat de kwaliteit van de informatie gewaarborgd kan worden. Hiervoor wordt het onderzoeksframework zoals gegeven op de Fontys gehanteerd. Per deelvraag binnen dit document is aangegeven welke onderzoeksstrategieën zijn toegepast.

(Methods, sd) – Voor meer informatie van het onderzoeksframework

Het onderzoek zal worden afgesloten met een eindconclusie die alle deelvragen samen pakt om één concreet antwoord te kunnen geven. Aan de hand van de conclusie zal er ook een advies worden uitgegeven.

# Planning

|  |  |
| --- | --- |
| Sprint 2 |  |
| Week 1 | * **Initiatie van het onderzoeksdocument.** * **Welke frameworks zijn er op dit moment?**   + Available product analysis. |
| Week 2 | * + Available product analysis. |
| Week 3 | * + Document analysis.   + Conclusies. |

|  |  |
| --- | --- |
| Sprint 3 |  |
| Week 4 | * **Welke frameworks worden gebruikt in reeds bestaande CRM/ticketing systemen?**   + Available product analysis.   + Conclusies. * **Wat maakt een framework geschikt voor een ticketsysteem?**   + Best good and bad practices.   + Explore user requirements.   + Conclusies. |
| Week 5 | * **Welke relevante voordelen hebben bepaalde frameworks?**   + Prototyping. |
| Week 6 | * + Prototyping.   + Component analysis.   + Conclusies. |

|  |  |
| --- | --- |
| Sprint 4 |  |
| Week 7+ | * **Verwerking in project.** * *Mogelijk nieuwe onderzoek punten.* * *Prototyping & Component tests* * *Conclusies* |
| Week 16 | * **Beantwoording hoofdvraag.**   + Conclusies.   + Advies. |

# Welke frameworks en libraries zijn er op dit moment?

|  |  |
| --- | --- |
| Strategie | Methode |
|  | Available product analysis |
|  | Document analysis |

## Available product analysis

### ‘Populaire’ frameworks

Er is op het internet veel informatie te vinden over front-end frameworks. Vooral de ‘top 10’ blogs en discussies overheersen. Daarom is het nuttig om eerst te bepalen welke frameworks het meest relevant zijn. Hoewel een afzonderlijke ‘top 10’ blog niet erg betrouwbaar is, lijkt het erop dat al deze blogs ongeveer dezelfde set aan frameworks behandelen.

De volgende zoektermen zijn gebruikt:

* ‘Front end frameworks 2020’ ,
* ‘Best front end frameworks’,
* ‘Front end frameworks’,
* ‘All front end frameworks’.

Ter referentie leek ‘Best Frontend Frameworks of 2020 for Web Development’ - (Simform., sd), één van de meer verzorgde resultaten op de genoemde zoekopdrachten. Alle bronnen leiden tot een overeenkomende set resultaten waarin de top 3 altijd uit dezelfde 3 frameworks bestaat: Angular, React en Vue.

|  |  |  |  |
| --- | --- | --- | --- |
| 1. React 2. Angular 3. Vue | * jQuery * EmberJS * Backbonejs | * Semantic-UI * Foundation * Svelte | * MeteorJS * Preact |

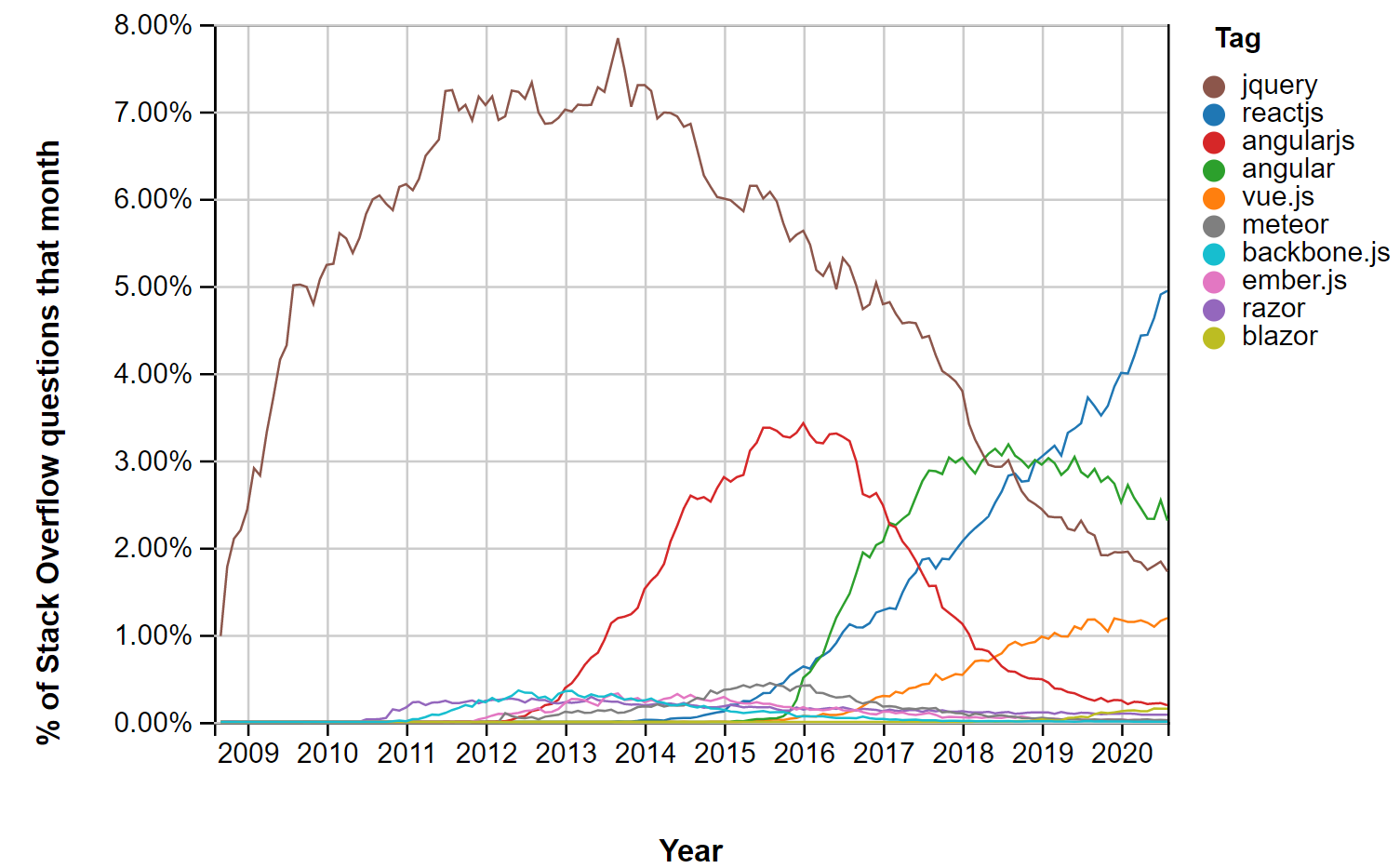
### Stack overflow trends

Om beter te kunnen bepalen in hoeverre de blogs een mening weergeven over persoonlijke favorieten of wat daadwerkelijk algemeen populaire frameworks zijn. Is er met ‘Stack Overflow Insights’ (Trends, sd) gekeken naar de waargenomen trends. De trend charts (hieronder weergeven) zijn gebaseerd op de gebruikte tags van de verschillende technologieën en de gestelde vragen per maand.

Wat hieraan op valt is:

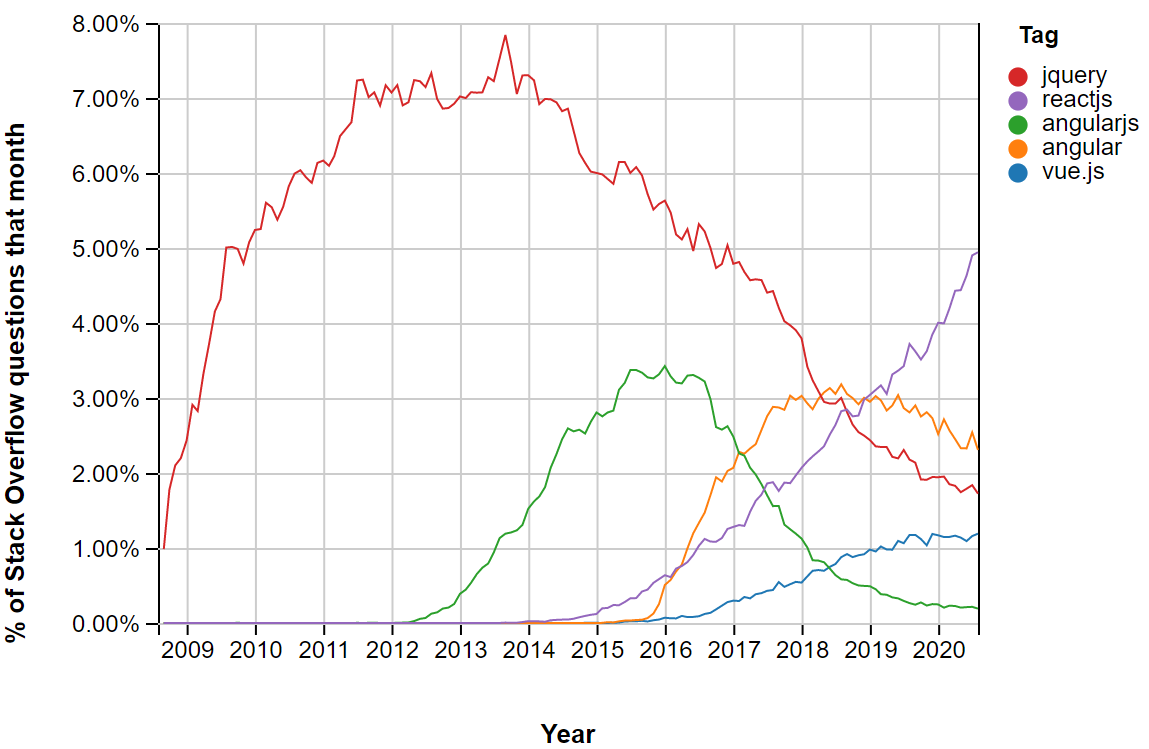
* Dat de gebruikersgemeenschappen van de top 3 frameworks erg actief zijn op stack overflow.
* Vooral React.js de laatste jaren een sterke stijging door gaat.
* De meest actieve frameworks/libraries allemaal Javascript (/typescript) zijn.

Frameworks en llibraries zoals gevonden op de ‘top 10’ sites.



(Stack overflow Trends, sd)

Meest actieve frameworks uitgelicht.



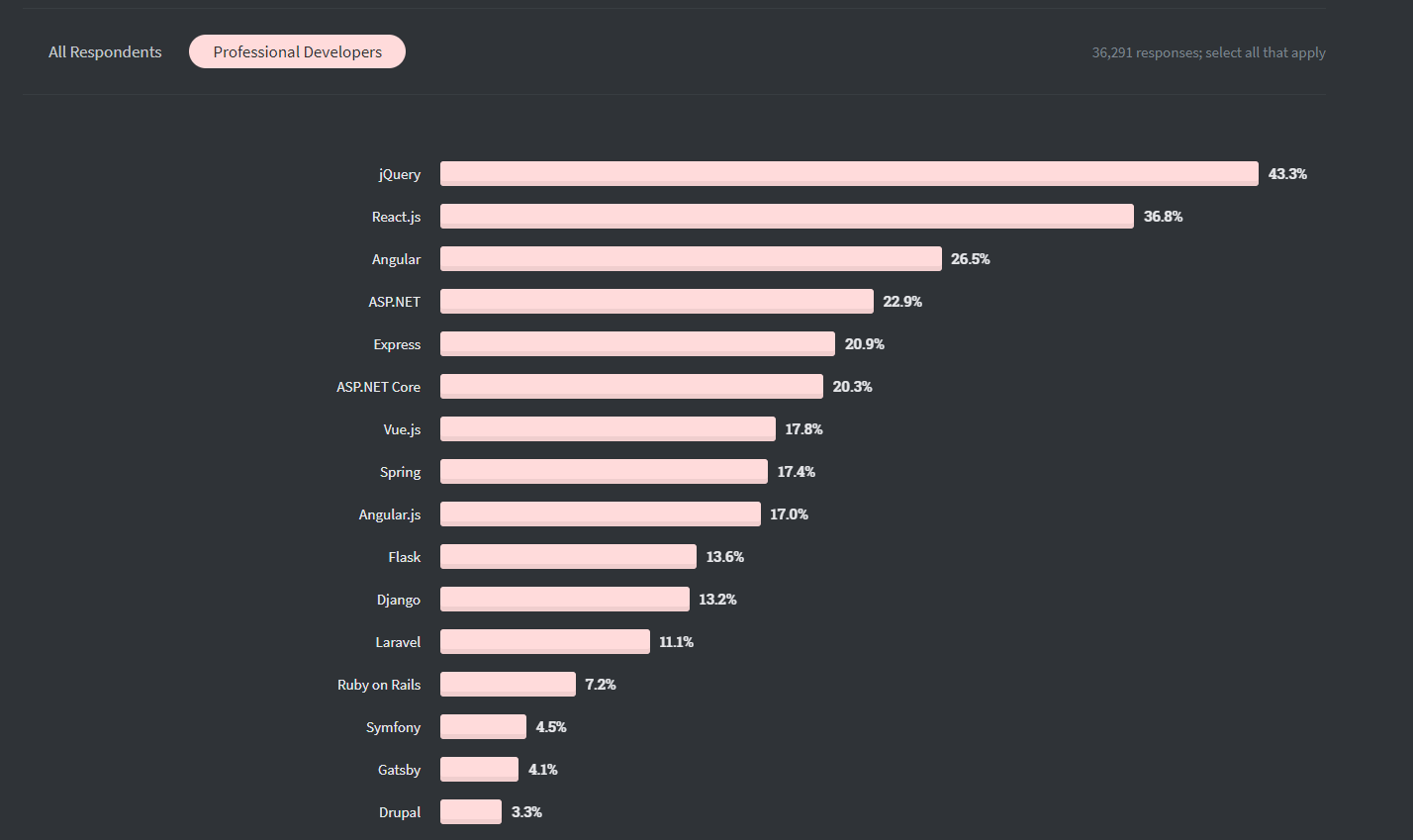
(Stack overflow Trends, sd)

### Zijn Stack overflow trends positief of negatief?

De trend charts op zichzelf geven geen beeld van of de trends positief zijn of negatief. Het enige dat ze aantonen is dat de gemeenschappen voor deze frameworks erg leven. Om hier een beter beeld op te krijgen is er gekeken naar de laatste ‘Developer survey’ van Stack overflow. Welke dateert uit Februari 2020 en is afgenomen onder 65.000 ontwikkelaars. (Insights, Developer Survey, sd)

#### Web Frameworks

De grafiek hieronder geeft een beeld van meest gebruikte web frameworks. Let er op dat dit niet allemaal front-end frameworks zijn. Daarnaast is gekozen om hier uitsluitend te kijken naar wat professionals aan hebben gegeven.



(Insights, Developer Survey, sd)

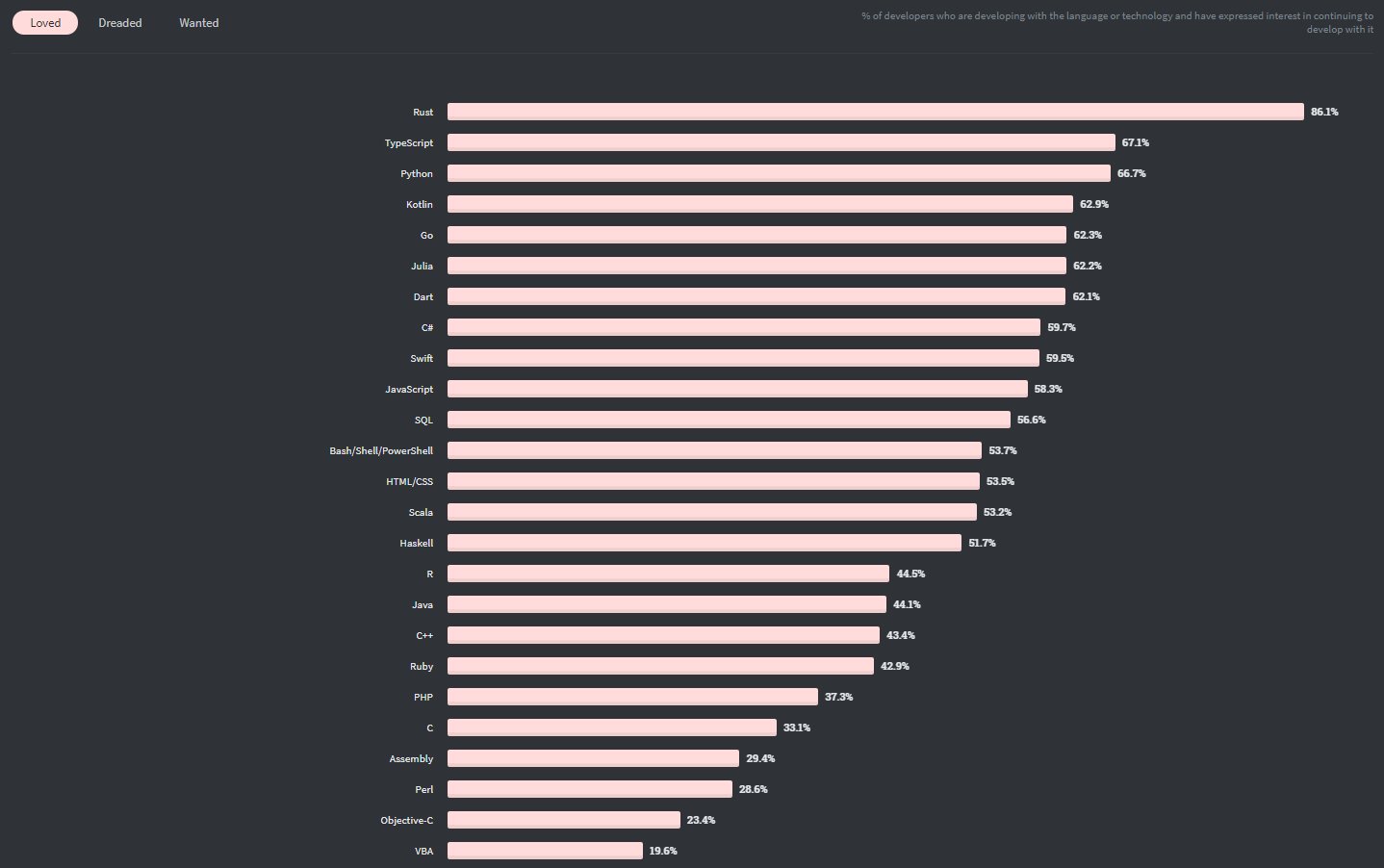
Wat hierin opvalt is:

* Dat jQuery nog steeds het meest gebruikte framework is voor front-end development.
* Dat React.js ook hier weer hoog geplaatst is.
* Dat Angular (Typescript) meer gebruikt wordt dan Angular.js.
* Dat buiten de ‘Top 3’ geen van de andere ‘Top 10 Frameworks’ in deze lijst voor komen.

#### Most loved, Dreaded, and Wanted: Talen

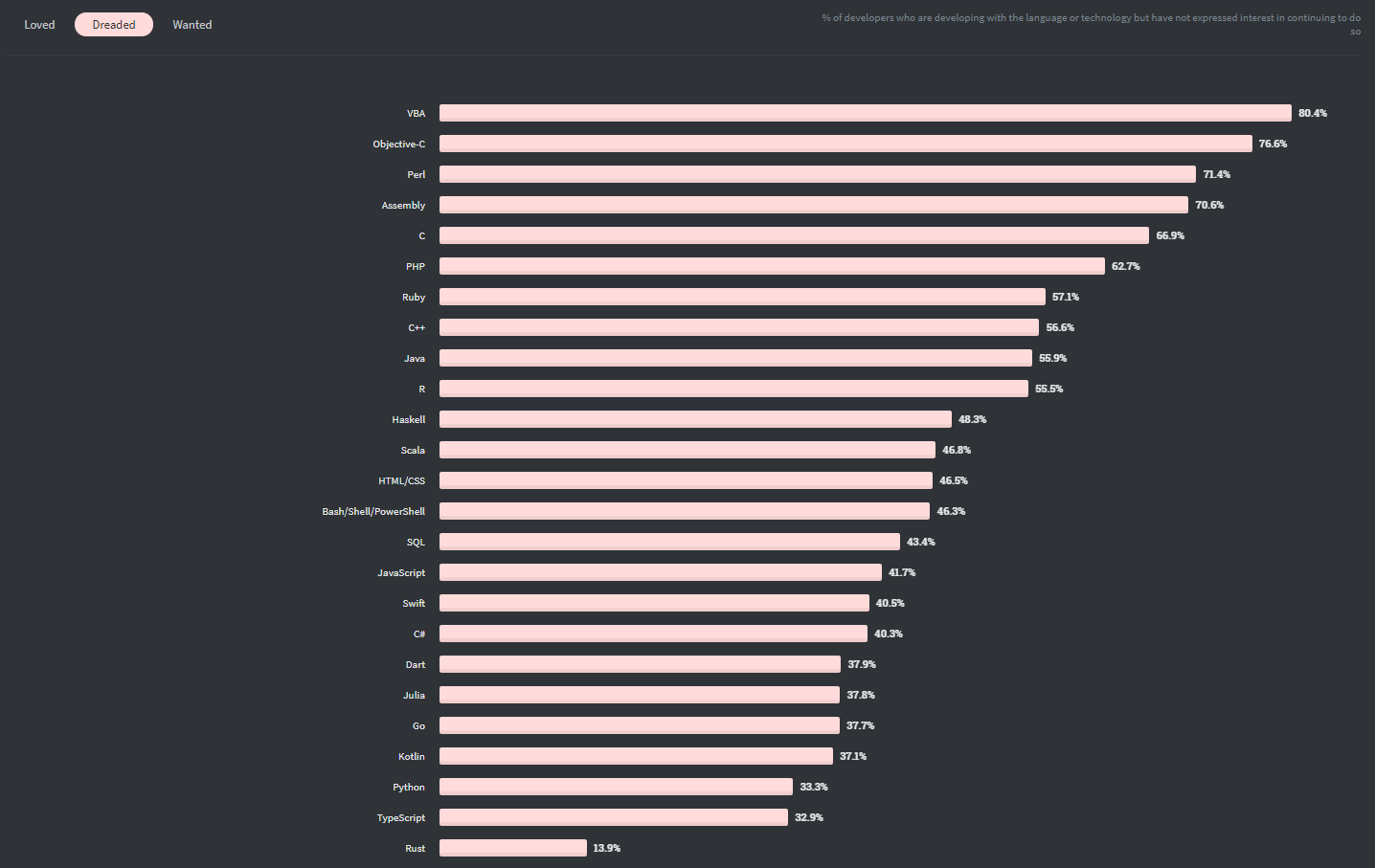
De ‘Developer Insights survey’. Geeft ook een beeld op welke technologieën het meest en het minst gewaardeerd worden door programmeurs, te beginnen met de talen.

*Languages - Loved*



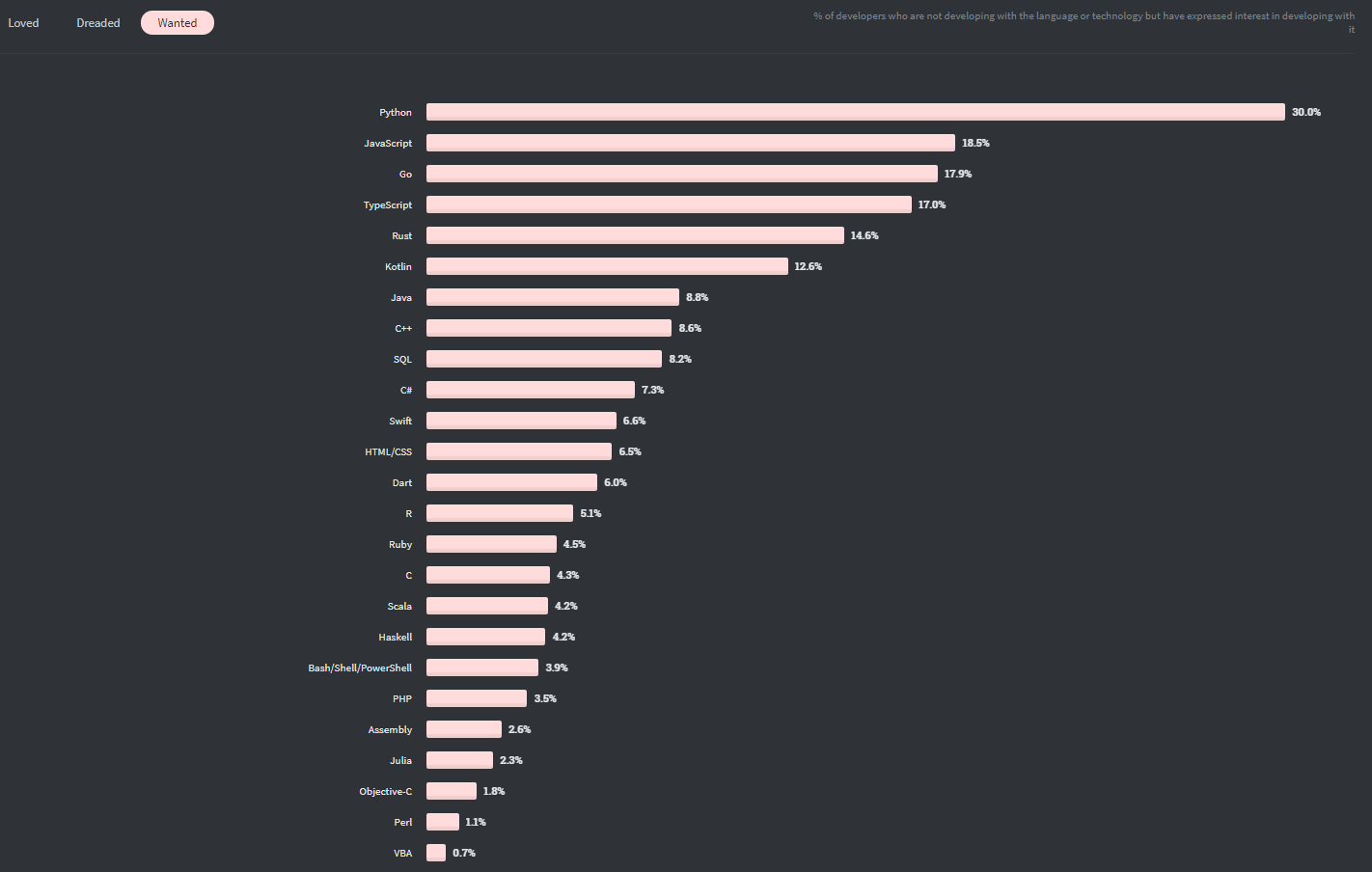
(Insights, Developer Survey, sd)

Languages - Dreaded



(Insights, Developer Survey, sd)

Languages – Wanted (to learn)



(Insights, Developer Survey, sd)

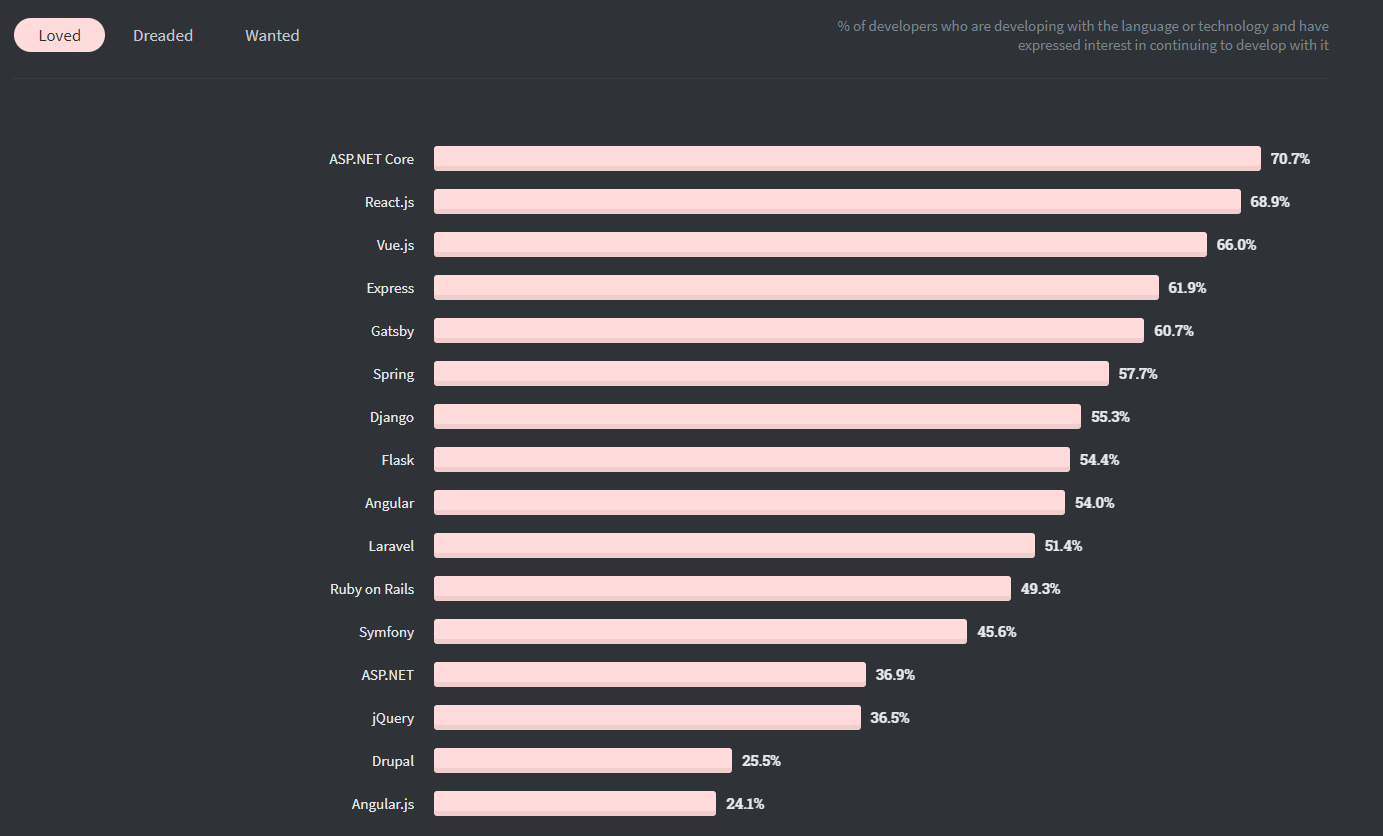
De grafieken hierboven geven datasets waarin zowel front-end als backend talen genoemd worden. Hierin valt het volgende op:

* Van de talen die gebruikt worden in de eerder genoemde front-end frameworks is Typescript meer geliefd dan Javascript.
* Javascript is daarentegen wel net iets meer gewenst om te leren voor programmeurs die de taal nog niet gebruiken. Maar dit is minimaal.

#### Most loved, Dreaded, and Wanted: Frameworks

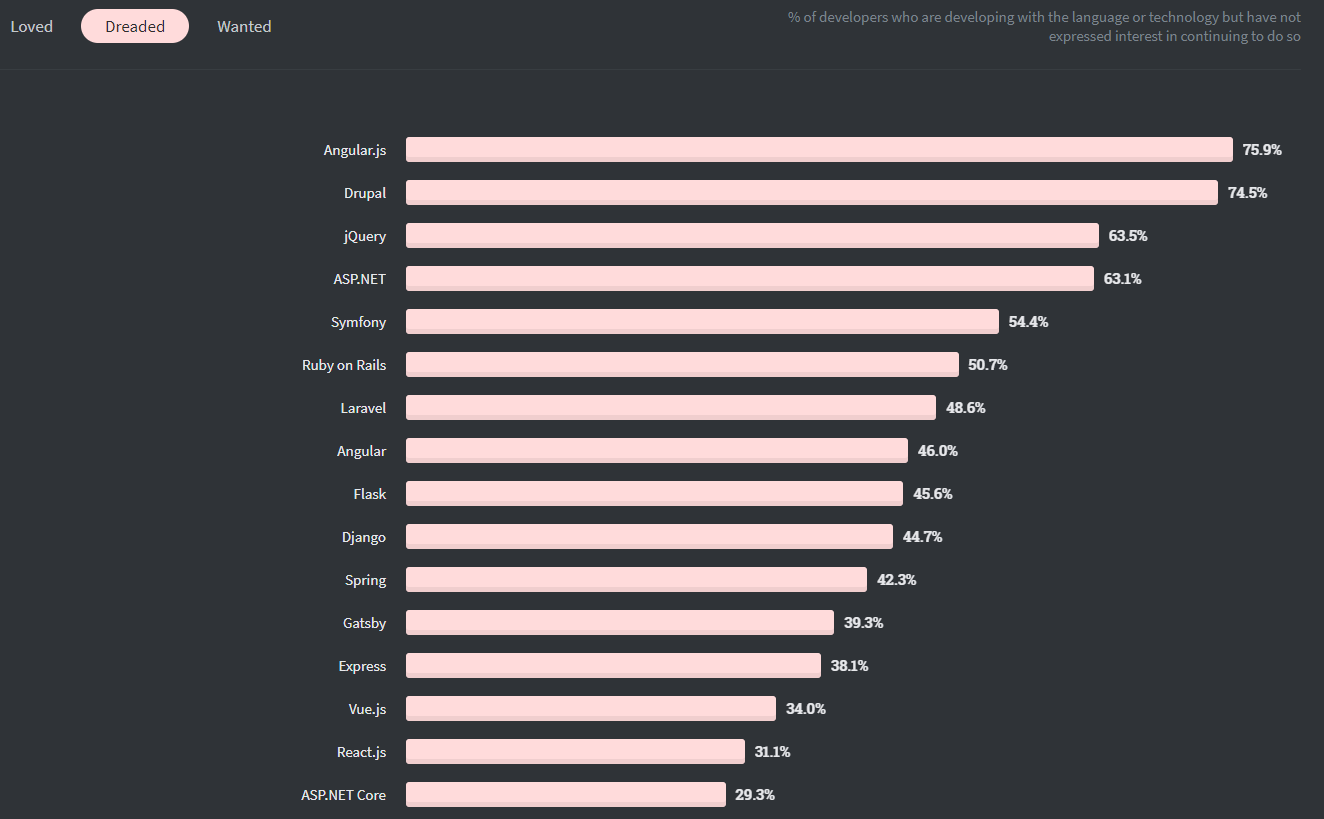
De volgende grafieken geven een weergave van hoe verschillende frameworks ervaren worden door programmeurs.

*Frameworks - Loved*



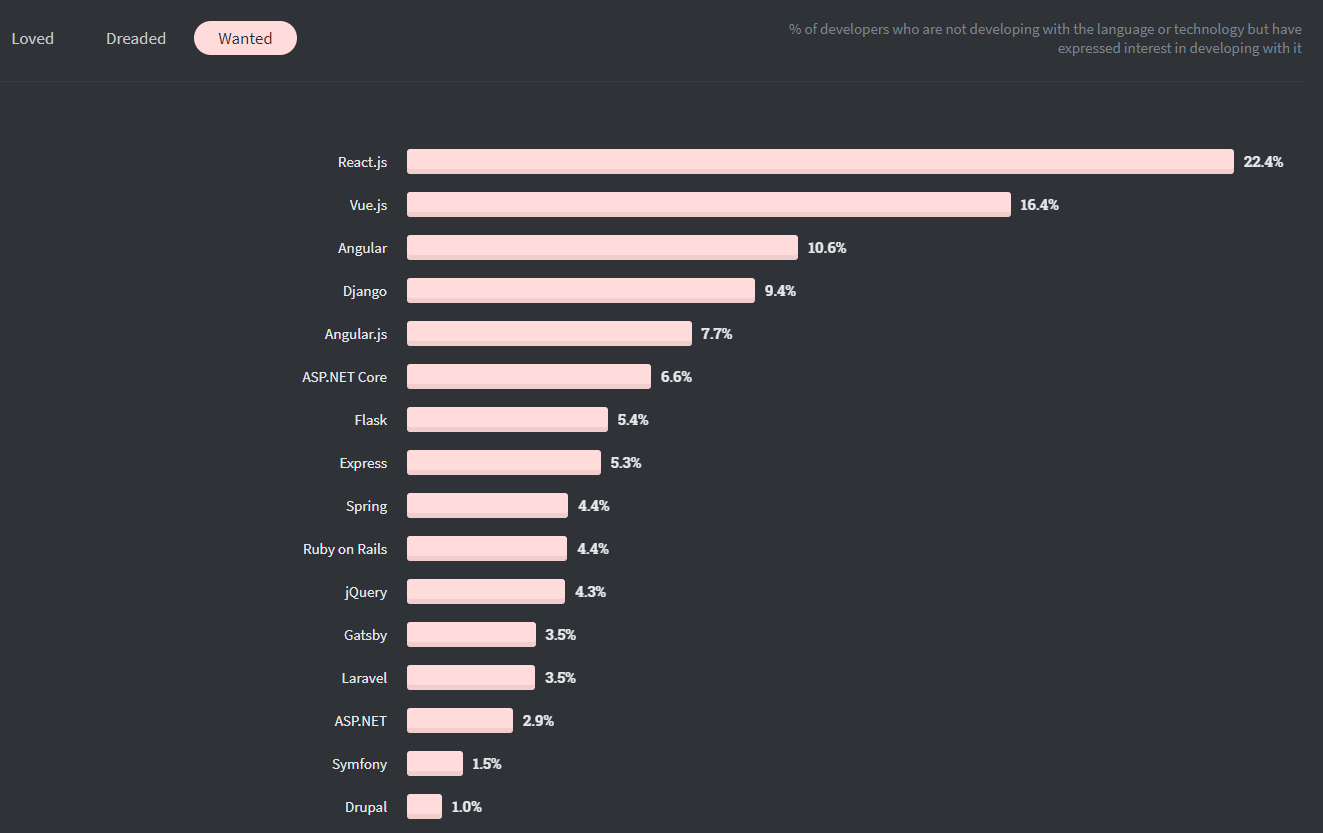
(Insights, Developer Survey, sd)

*Frameworks - Dreaded*



(Insights, Developer Survey, sd)

*Frameworks – Wanted (to learn)*



(Insights, Developer Survey, sd)

Het volgende valt op aan deze grafieken:

* React.js en Vue.js zijn erg geliefd onder de programmeurs die dit gebruiken.
* Angular.js is veruit de meest gehate taal.
* Angular (Typescript) wordt door veel developers niet positief ervaren.

### Conclusie – Available product analasys

Wanneer de trendgrafieken worden vergeleken met de survey-grafieken is er een correlatie te vinden tussen hoeveel vragen er op Stack Overflow worden gesteld en welke talen en frameworks het meest gewaardeerd worden door professionele ontwikkelaars. Het lijkt er daarmee op dat hoe geliefder een taal is, hoe meer vragen hierover te vinden zijn op Stack Overflow. Ook lijkt het er op dat er buiten de ‘Top 3’ weinig tot vrijwel geen interesse is in de overige ‘Top 10’ talen die op alle blogs worden gegeven.

Op basis van de grafieken lijken de volgende technologieën het beste naar voren te komen:

1. **jQuery:** *Nog steeds de meest gebruikte library.*
2. **React.js:** *Sterkst opkomende framework.*
3. **Vue.js:** *Zou Angular in de komende maanden kunnen passeren in het aantal gebruikers.*
4. **Angular (Typescript):** *Een veel gebruikt framework dat bijna evenveel geliefd lijkt als gehaat.*

## Document Analysis

Naast wat populair en geliefd is onder programmeurs is het ook belangrijk om te weten hoe goed de frameworks en libraries zijn gedocumenteerd. Hiervoor wordt de eerder vastgestelde ‘top 10’ als uitgangspunt genomen.

### jQuery

|  |  |  |
| --- | --- | --- |
| *Library* |  | |
| Website | | <https://jquery.com> |
| Documentatie | | Onder de kop ‘API Documentation’ is alles terug te vinden over de library. Ieder onderwerp is voorzien van duidelijke uitleg en code snippets. |

### React.js

|  |  |  |
| --- | --- | --- |
| *Library* |  | |
| Website | | <https://reactjs.org/> |
| Documentatie | | Op de website van React.js is de documentatie is in 2 vormen opgezet. De website biedt:   * Een tutorial die stap voor stap een uitgebreide uitleg geeft ondersteund door code snippets. * De ‘Docs’ pagina welke een uitgebreide documentatie van alle mogelijkheden van React.js geeft. Ook hier is ieder onderwerp uitgebreid omschreven en voorzien van code snippets. |

### Vue.js

|  |  |  |
| --- | --- | --- |
| *Framework* |  | |
| Website | | <https://vuejs.org/> |
| Documentatie | | Ook Vue.js bied 2 vormen van documentatie.   * De ‘Docs’ sectie bevat verschillende guides uitgebreide informatie en code snippets. * De ‘API Reference’ sectie bevat alle onderdelen van de Vue API. Ook hier wordt ondersteund met code snippets. |

### Angular (Typescript)

|  |  |  |
| --- | --- | --- |
| *Framework* |  | |
| Website | | <https://angular.io/> |
| Documentatie | | De ‘Docs’ pagina heeft een uitgebreid menu met verschillende concepts, features, best practices en tutorials. Deze zijn allemaal voorzien van een duidelijke uitleg en snippets. |

### Ember.js

|  |  |  |
| --- | --- | --- |
| *Framework* |  | |
| Website | | <https://emberjs.com/> |
| Documentatie | | Ember.js heeft onder het kopje ‘Docs’ uitgebreide documentatie, Api Reference en CLI Guides. De pagina’s lezen wat minder prettig dan de voorgaande frameworks maar zijn compleet en uitgebreid. |

### Backbone.js

|  |  |  |
| --- | --- | --- |
| *Library* |  | |
| Website | | <https://emberjs.com/> |
| Documentatie | | De website van backbone.js is 1 lange pagina met daarop een hoop informatie. Lange lappen tekst afgewisseld met snippets vertellen wat kan met backbone.js. De pagina geeft geen duidelijk beeld hoe de code in elkaar past en samen tot een goed project kan leiden. |

### Semantic-UI

Semantic-ui is terug te vinden op: <https://semantic-ui.com/>.

|  |  |  |
| --- | --- | --- |
| *Framework* |  | |
| Website | | <https://semantic-ui.com/> |
| Documentatie | | Semantic UI heeft een uitgebreide documentatie op hun website staan met practische voorbeelden, welke ondersteund worden door korte code snippets. De getting started sectie leid de geïntereseerden door het setup proces om met Semantic aan de slag te kunnen.  Semantic richt zich vooral op view design en bied verschillende plugins om o.a. te kunnen koppelen met de eerder genoemde ‘react’ en ‘vue’. |

### Foundation

|  |  |  |
| --- | --- | --- |
| *Framework* |  | |
| Website | | <https://get.foundation/> |
| Documentatie | | De website van Foundation bied uitgebreide documentatie in verschillende vormen. Zo is er een ‘getting started’ sectie welke templates bevat voor allerlei soorten projecten, zijn er video’s te vinden in de tutorials sectie en is er een uitgebreide ‘Docs’ pagina die alle onderdelen van foundation stap voor stap af gaat. |

### Svelte

|  |  |  |
| --- | --- | --- |
| *Framework* |  | |
| Website | | <https://svelte.dev/> |
| Documentatie | | Svelte bied documentatie in de vorm van tutorials, API beschrijvingen en voorbeelden. Opvallend hierbij is dat svelte de tutorials en voorbeelden ondersteund met interactieve code. Hierdoor kan er met voorbeelden gespeeld en getest worden. |

### Meteor

|  |  |  |
| --- | --- | --- |
| *Framework* |  | |
| Website | | <https://www.meteor.com/> |
| Documentatie | | Meteor heeft 2 secties met documentatie, welke te bereiken zijn via de ‘Developers’ dropdown in het hoofdmenu van de website. De tutorial sectie heeft 3 tutorials die variëren in moeilijkheidsgraad. De ‘Docs’ sectie heeft documentatie over ieder aspect van het framework. Deze documentatie bestaat uit uitgebreide uitleg en code snippets met links naar volledige code voorbeelden. |

### Preact

|  |  |  |
| --- | --- | --- |
| *Library* |  | |
| Website | | <https://preactjs.com/> |
| Documentatie | | De Preact website heeft een aantal pagina’s met ‘documentatie’. Het bestaat namelijk uit korte beknopte omschrijvingen en links naar github projecten waar volledige code is te vinden van o.a. de Preact website zelf. Verder bied het nog een korte uitleg over de CLI en de installatie daarvan. |

**Preact / React:**

*Preact is een variatie op React en is in bijna alle aspecten gelijk. Het verschil is dat de library zo klein mogelijk wordt gehouden door nier iedere feature die in React aanwezig is te op te nemen.*   
(Differences to React, sd)

### Conclusie Document analasys

De meeste frameworks en libraries uit de ‘top 10’ hebben een website met uitgebreide documentatie en voorbeelden die het mogelijk maken om de technologie aan te leren. Het valt hierbij op dat de ‘top 3’: Angular (Typescript), React.js en Vue.js de meest omvattende documentatie bevatten. Dit is voor ontwikkelaars in de meeste gevallen een groot voordeel, omdat het de kans op het op kunnen lossen van codeer problemen waarschijnlijker maakt.

## Conclusie: beschikbare front-end frameworks en libraries.

De ‘top 10’ aan frameworks en libraries zijn uitgebreid gedocumenteerd wat meedraagt aan hun populariteit onder ontwikkelaars. Uit de trendgrafieken van Stack Overflow is te concluderen dat er binnen deze top 10 een onwijs grote voorkeur leeft voor: Angular (Typescript), React.js, Vue.js en jQuery. Dit is tevens terug te leiden uit de statistieken van meest besproken frameworks en libraries op Stack Overflow.

Voor het bouwen van een front-end voor een ticketing systeem is het op dit punt van het onderzoek aan te raden Angular (Typescript), React.js, Vue.js of jQuery te gebruiken. Deze frameworks en libraries bieden de meest uitgebreide documentatie en zijn het meest besproken binnen de ontwikkelaarsgemeenschappen op Stack Overflow. Dit maakt de kans op onoplosbare problemen tijdens de ontwikkeling van de front-end kleiner. ***Met deze insteek zal de verdere loop van dit onderzoek zich vooral, maar niet uitsluitend richten om hier de beste technologie in te vinden.***

# Welke frameworks worden gebruikt in reeds bestaande CRM/ticketing systemen?

|  |  |
| --- | --- |
| Strategie | Methode |
|  | Available product analysis |

## Available product analysis

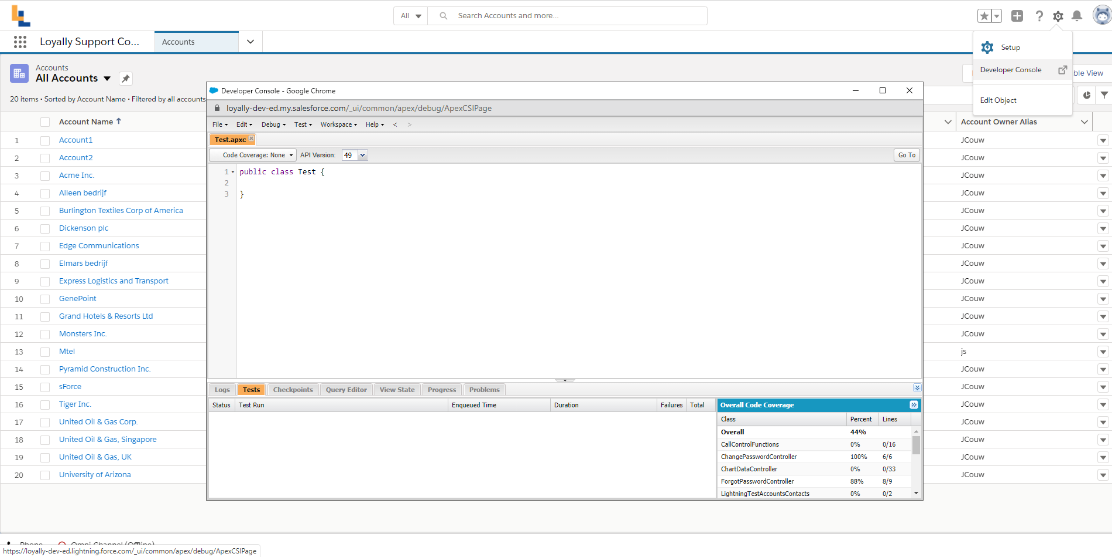
### Salesforce

Salesforce is een bekend CRM systeem dat wereldwijd gebruikt wordt. Er kan een ‘developer edition’ worden aangemaakt via <https://developer.salesforce.com/signup>. Salesforce gebruikt voor de front-end development verschillende frameworks die zij zelf hebben gebouwd en onderhouden. Deze frameworks kunnen door ontwikkelaars worden opgepakt om zelf omgevingen te ontwikkelen en in te richten.

Het hoofd component van Salesforce is de ‘***Lightning Component Framework’***. Dit is een UI framework voor het ontwikkelen van single page applicaties voor zowel mobiel als desktop. (Lightning Aura Components Developer Guide, sd). De componenten die hiermee gebouwd worden heten ‘***Aura Components***’ en deze kunnen een combinatie bevatten van aura elementen, ***HTML, CSS, Javascript*** en andere Web-enabled code talen (Aura Components, sd).

Daarnaast maakt Salesforce gebruik van ‘***Apex’***. Dit is een ‘Multi tenant, on-demand programming language’ die client side draait. Het is een mengeling van Java, javascript, .Net en andere programmeer talen. (Apex Developer Guide, sd). Apex is daarmee een sterk getypte, object-georiënteerde programmeertaal.

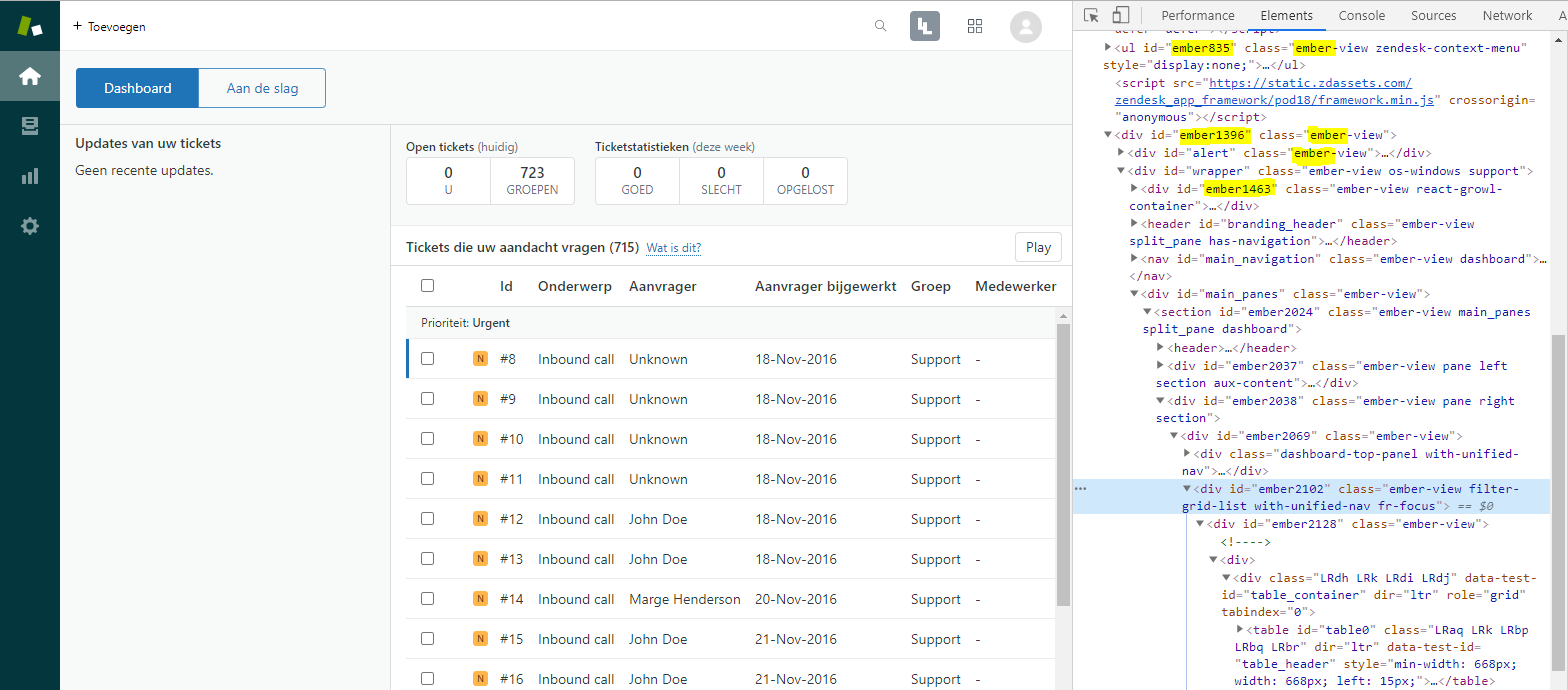
Als laatste maakt Salesforce ook gebruik van ***SOQL***: Salesforce Object Query Language. Hiermee kunnen SQL statements worden uitgevoerd om o.a. entiteit data uit Salesforce op te halen (SOQL and SOSL Reference, sd).



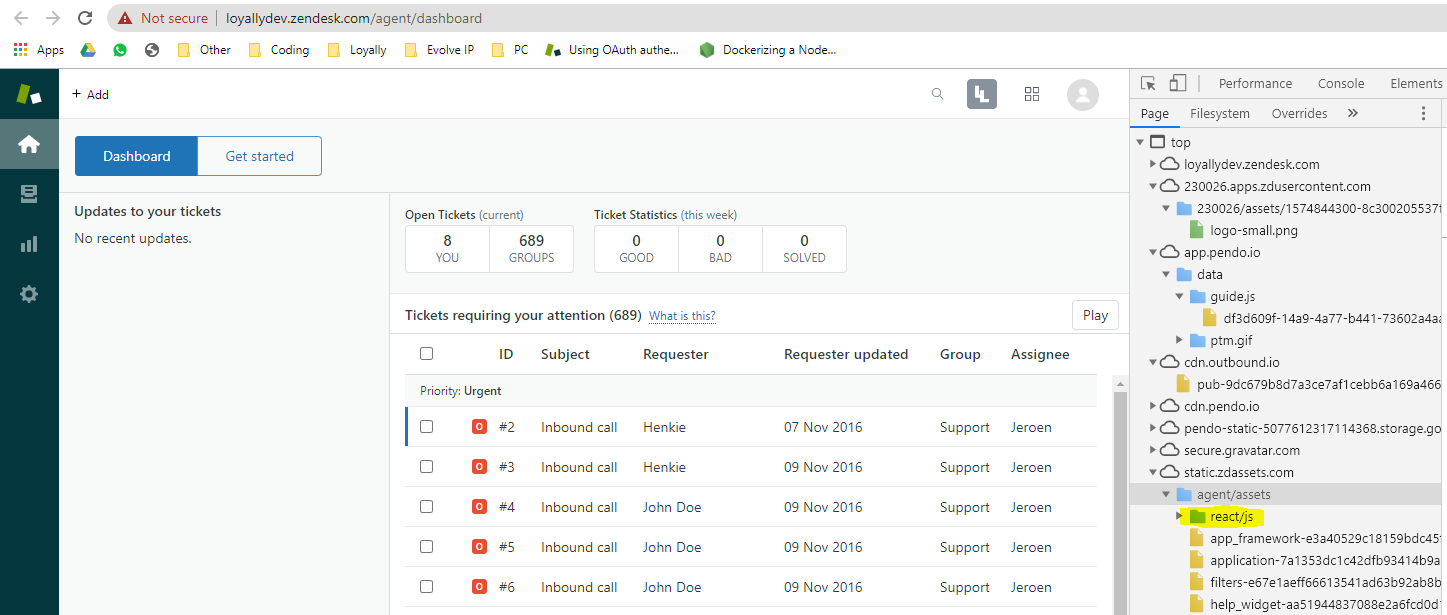
(Salesforce Portal, sd)

### Zendesk

Zendesk is één van de meest bekende ticketing systemen en wordt wereldwijd gebruikt. Er kan een trial account aangemaakt worden op <https://www.zendesk.com/support/pricing/>. Wanneer er is ingelogd op de portal en de browser console wordt geopend op de ‘Elements’ tab, kan er herleid worden dat er voor de front-end ontwikkeling gebruik is gemaakt van ***Ember.js.***Daarnaast is uit de ‘Sources’ tab worden herleid dat er daarnaast ook gebruik is gemaakt van***React.js***. Ook is in de code het gebruik van ***$*** terug te vinden wat duid op het gebruik van ***jQuery.***



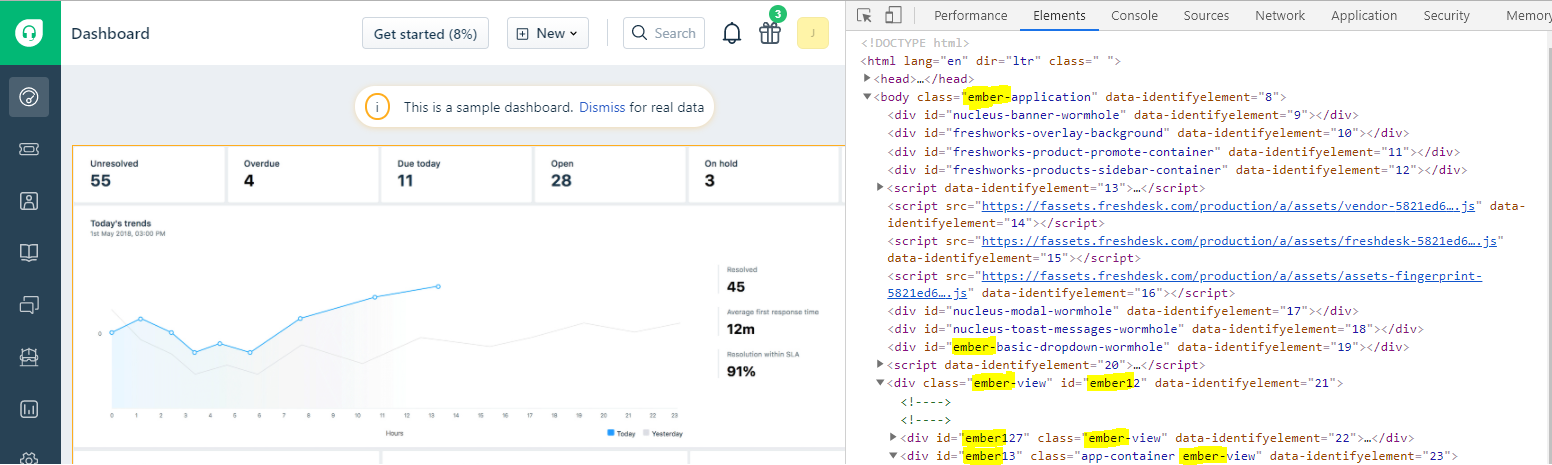
(Zendesk Portal, sd)



(Zendesk Portal, sd)

### Freshdesk (FreshService en Freshsales)

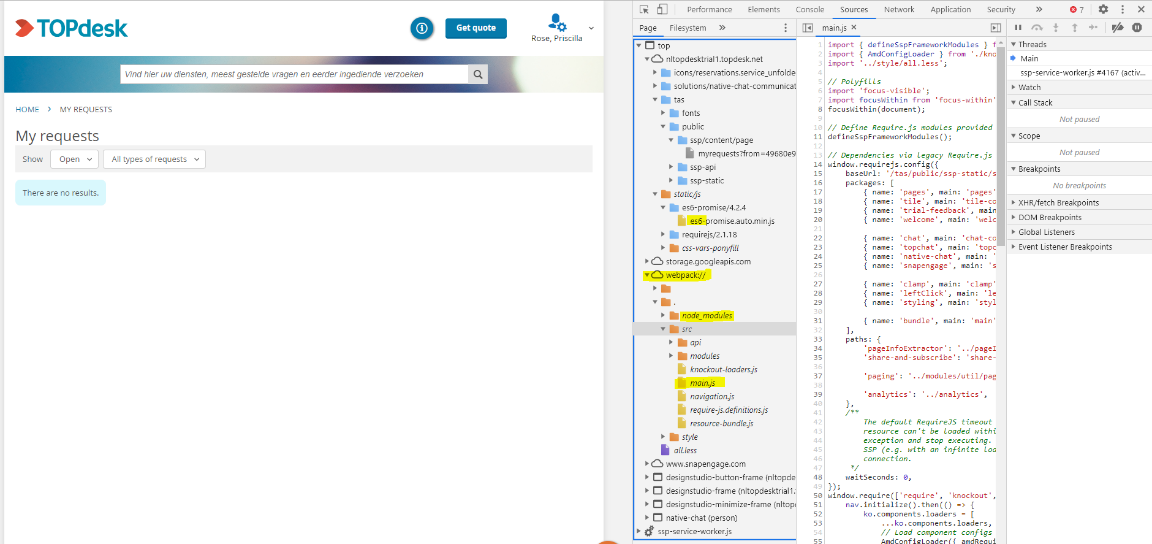
Freshdesk is een ticketing systeem van Freshworks, afkomstig uit India en heeft sterke gelijkenissen met Zendesk. Er kan een trial omgeving worden aangemaakt via <https://freshdesk.com/signup>. Dit genereert een Freshdesk omgeving waar ook apps tegenaan gebouwd en getest kunnen worden. Wanneer er is ingelogd op de portal en de browser console wordt geopend op de ‘Elements’ tab, kan er herleid worden dat er voor de front-end ontwikkeling gebruik is gemaakt van ***Ember.js***. Daarnaast is in de code terug te vinden dat er gebruik is gemaakt van ***$*** wat duid op jQuery.



(Freshdesk portal, sd)

### TOPDesk

TOPdesk is een ticketing systeem van Nederlandse komaf en is ontstaan in Delft (TOPdesk Journey, sd). Er kan een trial omgeving aangemaakt worden via <https://page.topdesk.com/nl/probeer-topdesk>. Na het aanmaken kan er worden ingelogd op de ‘Behandelaars’ portaal. Wanneer hier de browser console wordt geopend is er in de sectie ‘Sources’ te herleiden dat de front-end is gebouwd met ***jQuery*** (Javascript). De code is ondersteund door ***node.js*** elementen en vervolgens met ***webpack*** omgezet. Naast jQuery is er gebruik gemaakt van ***HTML en CSS.***



(TOPdesk Portal, sd)

### Omnidesk

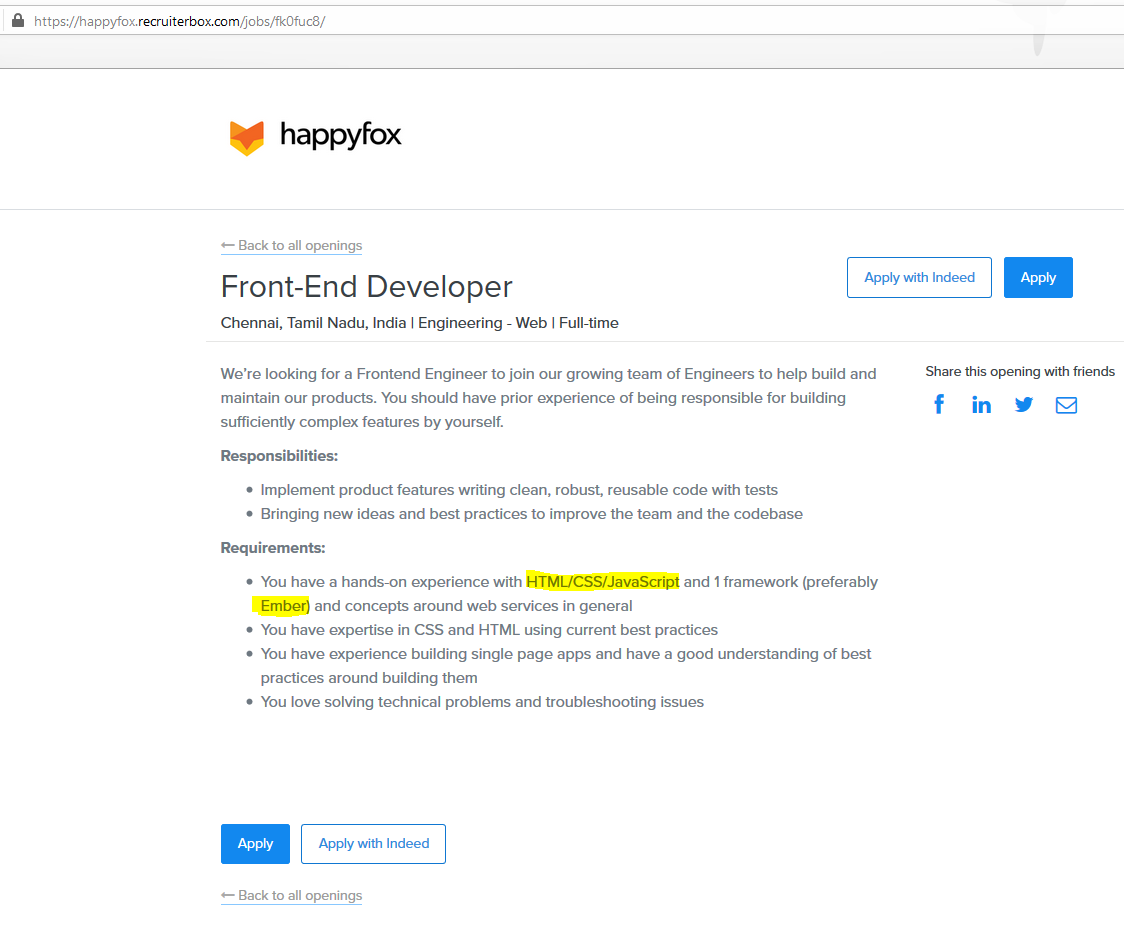
Omnidesk is een bedrijf uit leiden en biedt een gelijknamig ticketing platform. Er zijn geen mogelijkheden voor trials. Daarom is er niet met zekerheid te zeggen welk front-end framework ze gebruiken voor hun ontwikkeling. Wel zijn er vacatures te vinden waarin gevraagd wordt voor kennis van: PHP, Node.js HTML en Javascript***.*** Dit sluit niet uit dat er een javascript framework wordt gebruikt. Gezien het wel normaal is om dit te vragen in een vacature wanneer dit van toepassing is, lijkt de meest aannemelijke conclusie dat er ***jQuery*** wordt gebruikt.



(Omnidesk vacatures, sd)

### Happy Fox

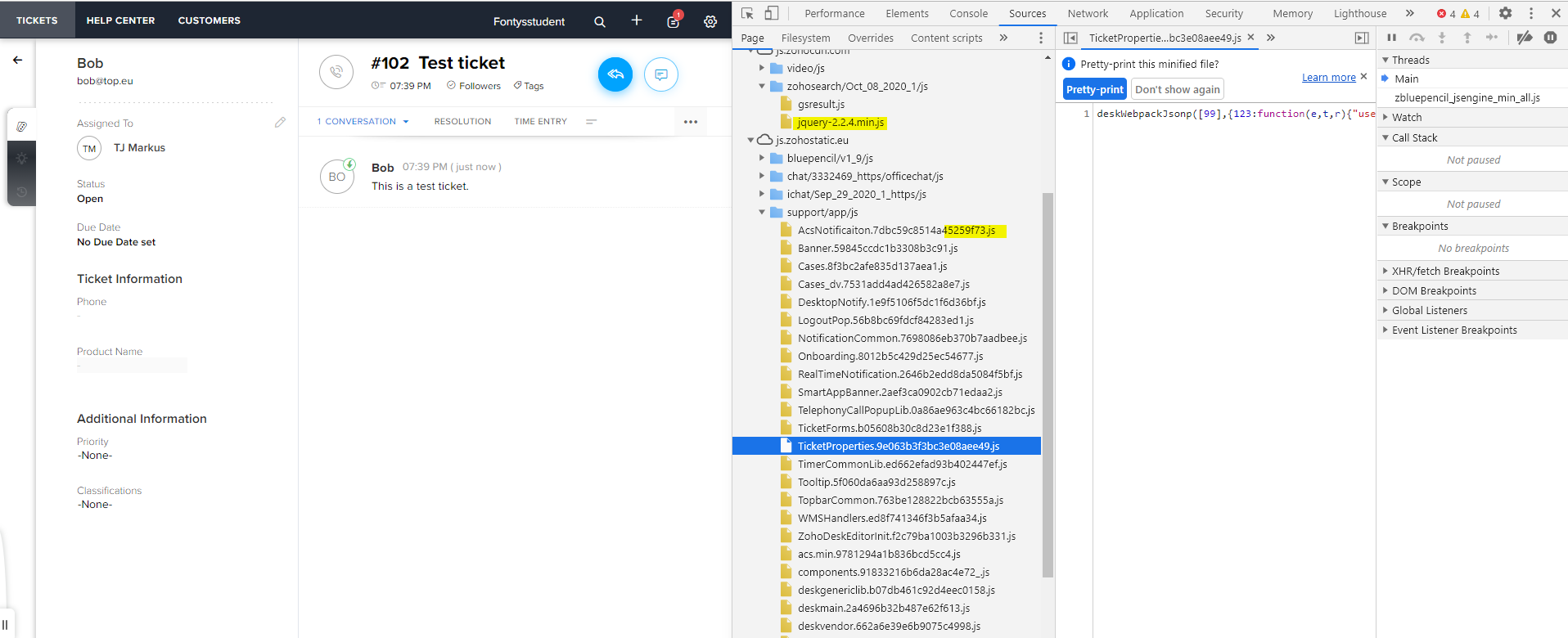
Happyfox is een helpdesk ticketing system gemaakt door HappyFox, Inc, California. Daarnaast werken ze ook vanuit Chennai, India. Het is niet mogelijk om via Happy Fox een gratis trial aan te vragen. Een gevonden ‘Front-End Developer’ vacature geeft hier wel meer zicht op. Hierin wordt gesproken over HTML/CSS/JavaScript en ***Ember.js***. Hieruit kunnen we concluderen dat het zeer waarschijnlijk is dat dit het front-end framework is dat ze gebruiken voor hun ticketingsysteem.



(Happy Fox Jobs, sd)

### Zoho Desk

Zohodesk is een ticketing systeem uit Chenai, India. Er kan een trial omgeving worden aangemaakt via <https://www.zoho.com/nl/desk/>. Na het aanmaken kom je als gebruiker terecht in de ticketing omgeving. Hier is een ticket aangemaakt waarna de browser console is geopend om de applicatie te inspecteren.

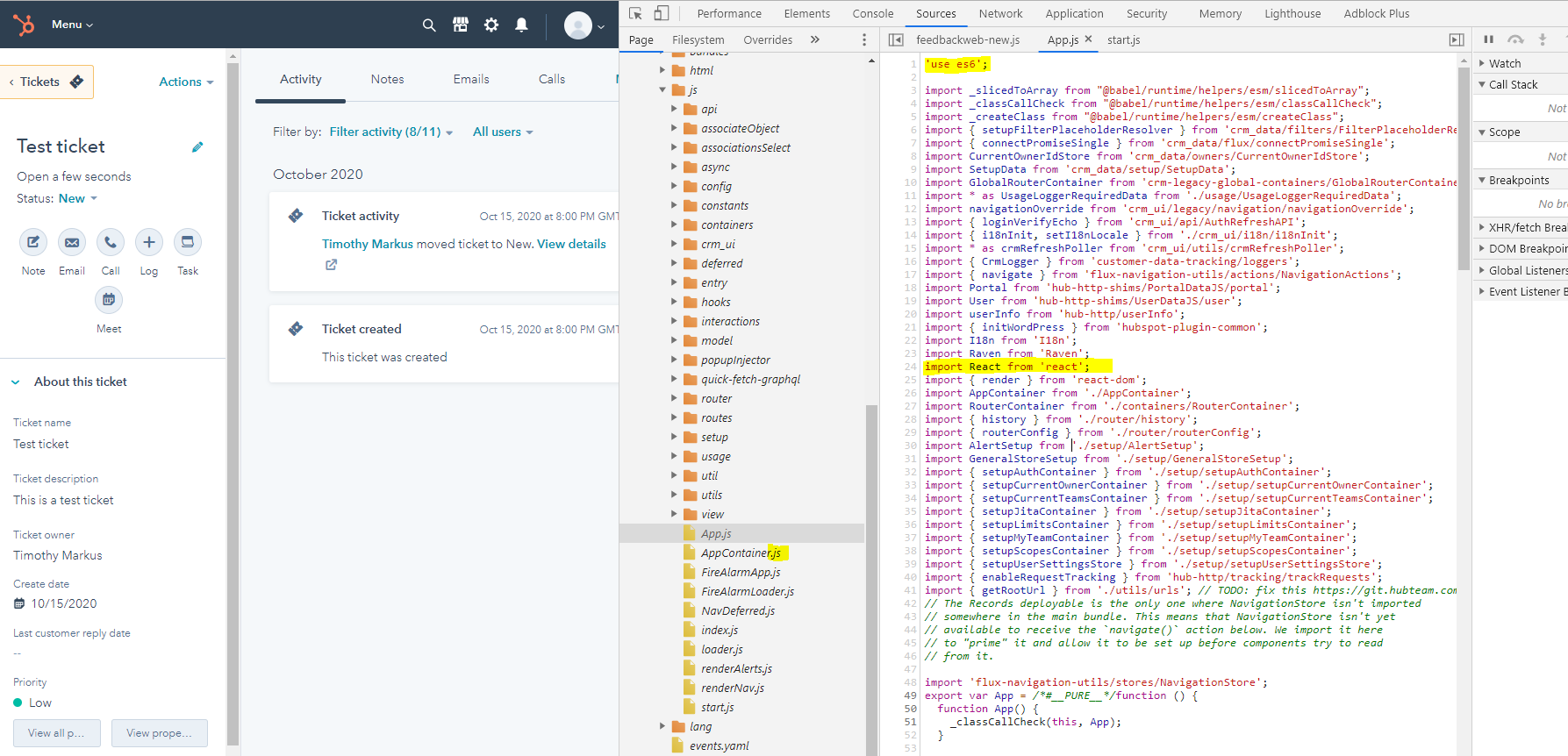


(Zoho Desk Portal, sd)

Uit de ‘sources’ is af te leiden dat er gebruik is gemaakt van jQuery. Ook is te zien dat de Javascript files ge-minimized en ge-uglified zijn wat de waarschijnlijkheid van het gebruik van ‘webpack’ hoog maakt. Verder zijn er geen elementen te vinden die duiden op een ander framework. Hierom is de conclusie dat de front-end is gebouwd met ondersteuning van de ***jQuery*** library.

### Hubspot

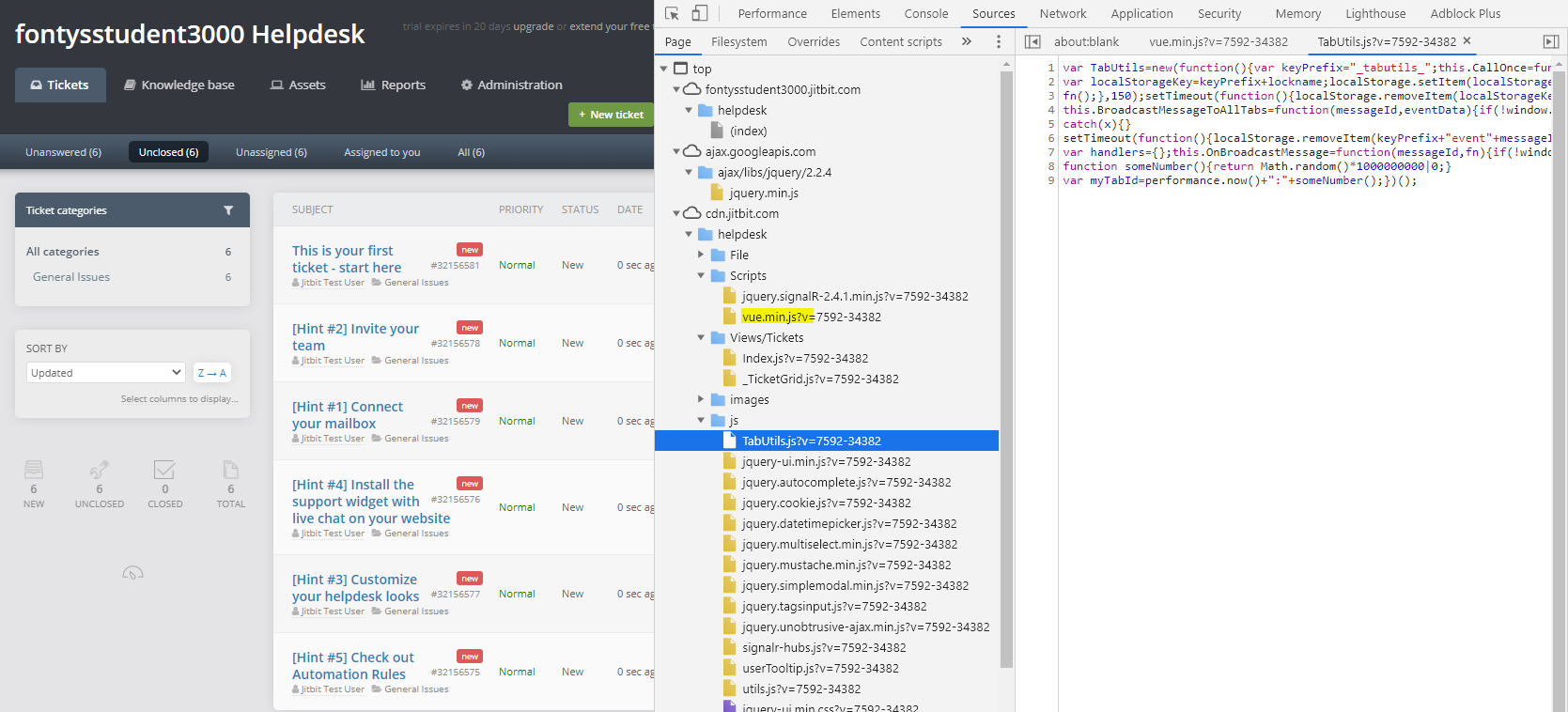
Hubspot is een amerikaans bedrijf dat is opgericht door studenten van de MIT. Het bied een aantal manieren voor het opzetten van een CRM platform waaronder de optie voor ‘Service Hub’, dit is een ticketing systeem. Er kan een volledig gratis omgeving worden aangemaakt via <https://www.hubspot.com/products/get-started>. Vanuit de hoofdpagina van de Portal kunnen tickets worden aangemaakt en verwerkt. Wanneer hier de browser console wordt geopend is er te herleiden dat er gebruik is gemaakt van ***React.js.*** Er zijn in de code geen tekenen terug te vinden van jQuery, in tegenstelling tot de andere onderzochte frameworks.



(HubSpot Portal, sd)

### Jitbit

Jitbit is een bedrijf met een gelijknamig ticketing systeem gevestigd in Eddinburgh, UK. Het ticketing systeem wordt in 2 varianten aangeboden: SAAS en On-Premise. Voor dit onderzoek is er gekeken naar de SAAS variant zodat er gelijkenis is met de overige onderzochte ticketing systemen. Er kan een trial worden aangemaakt via <https://www.jitbit.com/saas-helpdesk/>. Wanneer de gebruiker op de hoofdpagina komt zijn er vooraf al tickets aangemaakt. Bij het openen van de console kan er worden achterhaald de front-end is gebouwd met behulp van ***jQuery en Vue.js.***



(Jitbit Portal, sd)

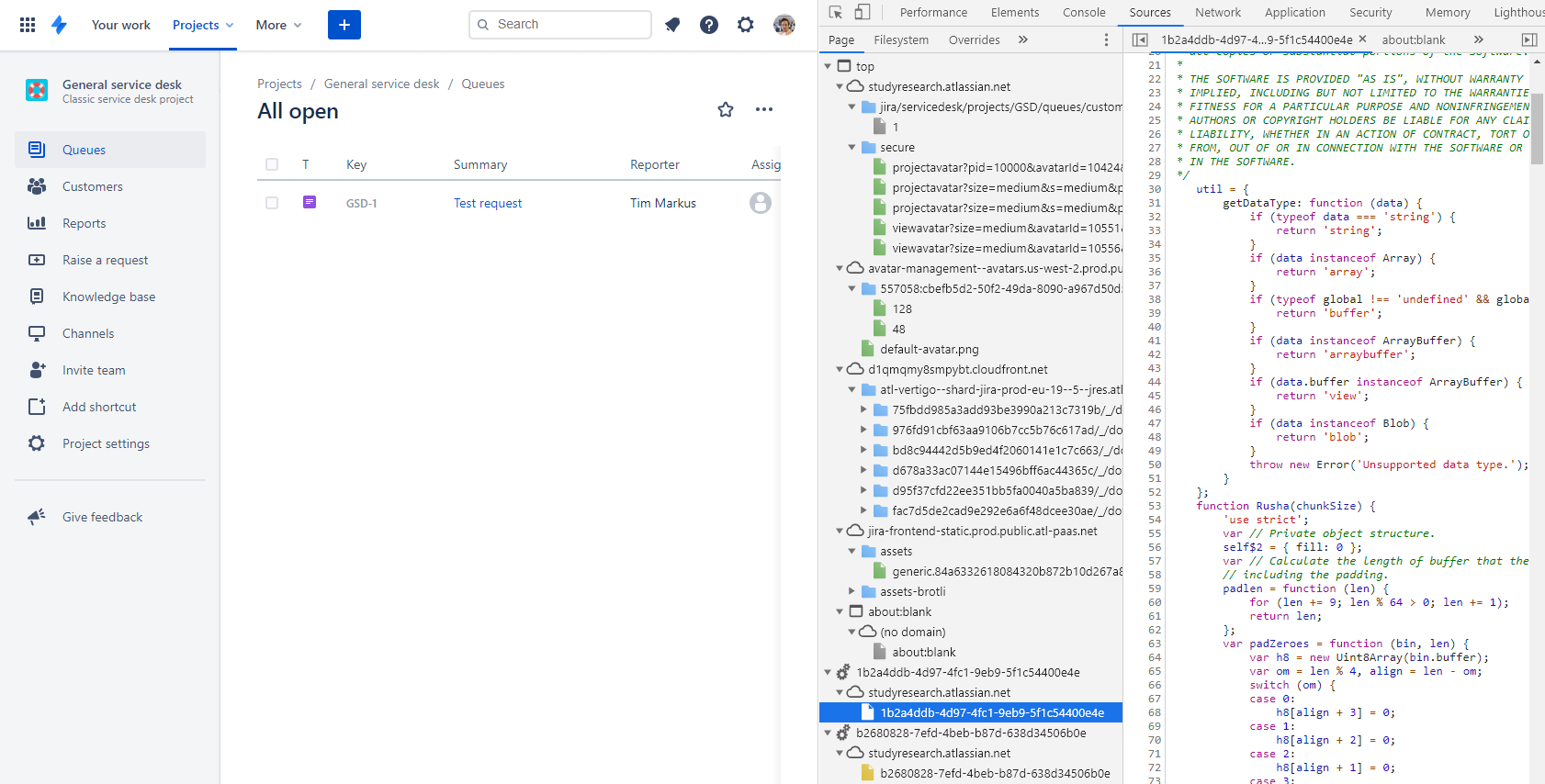
### Uitgesloten van conclusie

#### Jira Service Desk

Jira service desk is van Atlassian en bied verschillende mogelijkheden voor het opzetten van een customer support omgeving. Er kan een gratis trial omgeving worden opgezet via <https://www.atlassian.com/nl/software/jira/service-desk/free>. Voor dit onderzoek zijn er in de vervolg opties tijdens het opzetten van de omgeving de volgende keuzes gemaakt:

* Beste beschrijving voor de omgeving: Customer service.
* Functie van de gebruiker: support engineer.
* Type service desk: General service desk.

Het was bij Jira Service Desk niet duidelijk of er een specifiek framework is gebruikt. Wanneer de browser console op de portal pagina wordt geopend zijn er geen elementen te vinden die herkenbaar zijn voor één van de bekende frameworks. Wel is te herleiden dat er javascript is gebruikt.



(Jira Servicedesk Portal, sd)

Verder is er ook gekeken naar vacatures bij Atlasian maar hier waren geen Servicedesk specifieke vacatures te vinden. Bijna alle front-end vacatures geven meerdere frameworks aan: React, Angular en Vue.

Een aantal voorbeelden hiervan zijn de volgende:

<https://www.atlassian.com/company/careers/detail/41b3cad3-12c6-45c4-b5c8-f5f6357e0c90>

<https://www.atlassian.com/company/careers/detail/e718a93c-59fb-46ce-80d9-d952d191f32c>

<https://www.atlassian.com/company/careers/detail/7f57ba7b-548b-47ae-9356-34d7c3967c1d>



(Atlassian Careers, sd)

Omdat alle frameworks worden aangegeven in de vacatures is het niet vast te stellen welke exact wordt gebruikt voor de Service Desk. ***Hierom zal Jira service desk niet meegenomen worden in de uiteindelijke conclusie.***

## Overzicht

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Angular | React.JS | Vue.JS | Ember.JS | jQuery | Eigen framework |
| Salesforce |  |  |  |  |  | **X** |
| Zendesk |  | **X** |  | **X** | **X** |  |
| Freshworks |  |  |  | **X** | **X** |  |
| Topdesk |  |  |  |  | **X** |  |
| Omnidesk |  |  |  |  | **X** |  |
| Hapyfox |  |  |  | **X** |  |  |
| Zohodesk |  |  |  |  | **X** |  |
| Hubspot |  | **X** |  |  |  |  |
| Jitbit |  |  | **X** |  | **X** |  |
| *~~Jira Servicedesk~~* | ? | ? | ? | ? | ? |  |
| TOTAALEN | **0** | **2** | **1** | **3** | **6** | **n.v.t.** |

*Overzicht van de onderzochte frameworks.*

Freshdesk, Freshsales en Freshservice zijn software pakketten van Freshworks. Deze 3 lijken zeer sterk op elkaar en zullen voor dit onderzoek als 1 product worden beschouwd. Dit om mogelijke vertekening van het framework/library gebruik te voorkomen.

## Conclusie: Frameworks in bestaande CRM & Ticketing systemen

jQuery is veruit de meest gebruikte Library in de onderzochte CRM en ticketing systemen. Daarnaast sluit het gebruik van jQuery het gebruik van andere frameworks niet uit. Bij de onderzochte pakketten worden de 3 grote frameworks minder vaak gebruikt dan Ember.js. Angular kwam niet voor.

*Belangrijk bij deze conclusie is dat er is gekeken naar een ‘top 10’ CRM/Ticketing systemen. Het toevoegen van andere pakketten zou het huidige beeld kunnen veranderen.*

# Wat maakt een framework geschikt voor een ticketsysteem?

|  |  |
| --- | --- |
| Strategie | Methode |
|  | Explore user requirements |
|  | SWOT Analysis |

## Explore user requirements

Om vast te stellen wat een framework geschikt maakt voor een ticketing systeem moet er eerst vastgesteld worden wat er in het beoogde ticketing systeem verwacht wordt. Hiervoor worden de user requirements van het ‘Nemesis Project Pan’ gebruikt. Daarin wordt het volgende omschreven.

*1.1 Als support medewerkers wil ik kunnen zien welke meldingen nog open staan, zodat ik weet waar ik aan moet werken. [M]*

*1.2 Als klant wil ik meldingen kunnen indienen, zodat ik kan laten weten waar Xedule verbeterd kan worden. [M]*

*2.1 Als support medewerkers wil ik meldingen verschillende types meegeven zodat het overzichtelijk is wat iedere melding betreft. [S]*

*2.2 Als support medewerker wil ik meldingen verschillende statussen geven zodat overzichtelijk is wat er nog voor de melding gedaan moet worden. [S]*

*2.3 Als support medewerker wil ik diverse overzichten van groepjes meldingen. [S]*

*2.4 Als klant wil ik een overzicht van mijn meldingen zodat ik na kan gaan wat de status is per melding. [C]*

*2.5 Als medewerker wil ik de per ticket-type de velden van tickets kunnen toevoegen zodat ik naar eigen hand het ticket-formulier kan uitbreiden. [C]*

*2.6 Als support medewerker wil ik zelf nieuwe overzichten van meldingen kunnen maken. [C]*

*3.1 Als Service Manager wil ik een analytisch overzicht van tickets per onderwijsinstelling zodat ik deze kan bespreken. [S]*

*3.2 Als klant wil ik mijn gegeven kunnen aanpassen zodat ik ook zelf kan bijhouden wie meldingen mag indienen. [C]*

*3.3 Als ontwikkelaar wil ik dat gesloten bugs en taken in DevOps direct een notificatie naar de melding sturen en de melding klaarzetten om te sluiten. [C]*

(Nemesis Project Plan)

### Functionaliteiten op basis van de user requirements

Om de user requirements te kunnen realiseren zijn de volgende technische aspecten van belang voor de front-end.

1. Er moet vanuit de front-end gecommuniceerd worden met een backend (C#). Dit moet m.b.v. REST.
2. De front-end moet dynamisch en snel ticket informatie kunnen weergeven en voorzien van bewerkingen.
3. In de front-end moeten ticket categorieën gemakkelijk gefilterd kunnen worden.
4. De front-end moet ‘custom’ ticket Velden kunnen tonen. Deze moeten configureerbaar zijn door bevoegde gebruikers. Dit moet voor verschillende ticket types in te stellen zijn.
5. Het ticketing systeem moet informatie tonen afhankelijk van het type gebruiker. Hierbij kan o.a. gedacht worden aan: Klant, Service medewerker en Service manager.

Van de genoemde functionaliteiten is vooral de eerste belangrijk. Voor alle frameworks is documentatie te vinden voor het implementeren van REST communicatie:

* jQuery: <https://api.jquery.com/jquery.ajax/>
* Angular: <https://angular.io/guide/http>
* Vue.js: <https://vuejs.org/v2/cookbook/using-axios-to-consume-apis.html>
* React.js: <https://reactjs.org/docs/faq-ajax.html>
* Ember.js: <https://guides.emberjs.com/v1.10.0/models/connecting-to-an-http-server/>

Voor de overige functionaliteiten kunnen binnen de frameworks meerdere technische oplossingen worden bedacht. Een verdere uitwerking is daarom niet van belang.

## SWOT analysis, pros and cons

Het uitvoeren van een volledige SWOT analyse is voor dit onderzoek niet nuttig. De ‘opportunities’ en ‘threats’ van de frameworks bieden geen toegevoegde waarde. De ‘strengths’ en ‘weaknesses’ daarentegen kunnen wel degelijk van invloed zijn op de uiteindelijke ‘beste framework of library’ keus.

### jQuery

|  |  |
| --- | --- |
| STRENGTH | WEAKNESS |
| * Eenvoudig * Flexibel door plug-ins (b.v. voor CSS) * Goed met verschillende browsers | * Jquery.js is één bestand welke tegenwoordig behoorlijk groot is * De library is ongestructureerd |

**Relevantie tot het te bouwen ticketing systeem:** Dat de library zelf 1 groot bestand is en ongestructureerd is maakt voor het gebruik niet veel uit. Deze willen we immers zelf niet aanpassen. Een groot pluspunt is dat jQuery als eenvoudig word beschouwd en dus makkelijk te gebruiken is in een project.

### Angular (Typescript)

|  |  |
| --- | --- |
| STRENGTH | WEAKNESS |
| * MVC architectuur * Modules voor het groeperen van components, pipes en services. * Dependency injection * Typescript maakt coderen makkelijker * 2-way databinding | * Complex door uitgebreidheid * Steile leercurve * Verdeelde community |

**Relevantie tot het te bouwen ticketing systeem:** Complexiteit en een steile leercurve zouden een probleem kunnen vormen voor de bouw van een ticketing systeem. Maar de structuur die met Angular geboden kan worden zou de bouw van een front-end erg overzichtelijk kunnen maken.

### Vue.Js

|  |  |
| --- | --- |
| STRENGTH | WEAKNESS |
| * Klein (wanneer gecompileerd) * Makkelijk leesbaar * 2-way reactieve databinding * Typescript support * Browser devtools * Virtuele DOM | * Kleine community * Te flexibel (dit kan projecten te complex maken) * Weinig plug-ins |

**Relevantie tot het te bouwen ticketing systeem:** Een kleine community kan een groot nadeel zijn wanneer er tijdens de bouw van een ticketing systeem tegen problemen wordt aangelopen. Met uitzondering van de virtuele DOM lijken de voordelen van Vue geen toegevoegde waarde te bieden ten opzichten van de andere frameworks.

### React.Js

|  |  |
| --- | --- |
| STRENGTH | WEAKNESS |
| * Virtuele DOM * Herbruikbare componenten * One-way dataflow * Grote community * React developer tool extensie voor browsers | * Verwarrende JSX * Focus uitsluitend op de view * Documentatie wordt als slecht beschouwd. * Steile leercurve |

**Relevantie tot het te bouwen ticketing systeem:** Een grote community zou het oplossen van problemen kunnen versnellen en de slecht beschouwde documentatie kunnen compenseren. De steile leercurve zou een nadeel kunnen zijn voor de bouw van een ticketing systeem.

### Ember.Js

|  |  |
| --- | --- |
| STRENGTH | WEAKNESS |
| * Goede documentatie * Synchroon tussen model en view * Volwassen community * URL support * Server-side rendering * Snel en flexibel | * Kleine community * Complex * Weinig libraries en code voorbeelden * Geen standaard set voor user interface elementen. |

**Relevantie tot het te bouwen ticketing systeem:** Het gebrek aan libraries en code voorbeelden in combinatie met een kleine community zou voor problemen kunnen zorgen tijdens de ontwikkeling van het ticketing systeem. De complexiteit zou ook erg nadelig kunnen zijn.

Uit de SWOT zijn geen voor- of nadelen te halen die de bouw van een ticketing systeem in de weg zou kunnen staan. Alle frameworks en libraries hebben pluspunten en complexiteiten die het makkelijker of moeilijker kunnen maken, maar niet onmogelijk.

## Conclusie: Frameworks geschikt voor Ticketing systemen

**Grote communities** kunnen een belangrijk voordeel bieden bij de bouw van een ticketing systeem. *Als alleen op dit aspect gewogen zou worden dan zouden Vue.js en Ember.js minder geschikt zijn. Daarnaast zou de verdeeldheid in de Angular community ook Angular minder geschikt kunnen maken.*

**Goede documentatie** **en code voorbeelden** kunnen ook een groot voordeel zijn voor het werken met de frameworks en libraries. *Als alleen op dit aspect gewogen zou worden dan zouden React.js en Ember.js minder geschikt zijn. Ondanks dat de community van Ember.js als volwassen wordt beschouwd.*

**Complexheid** van de frameworks en libraries zouden nadelig kunnen zijn. *Op dit aspect komt alleen jQuery als positief uit de test.*

**Overzichtelijkheid en structuur** kan het bouwen van code in de frameworks en het vinden van fouten en oplossingen makkelijker maken. Een framework dat structuur van code en bestanden en mappen forceert kan hier een groot voordeel geven. *Hierin zijn Angular en Ember.js sterker dan de andere frameworks en libraries. Al houd niks een programmeur tegen om in de projecten binnen de andere frameworks een goede structuur aan te brengen.*

**Two-way databinding en one-way data binding** hebben beide hun voor en nadelen. Two way zorgt er voor dat aanpassingen in view 1 de states van view 2 en 3 kunnen veranderen. Dit kan voordelen hebben binnen een klein project maar in een groot project kan het fouten opsporen en herstellen erg moeilijk maken. Het is niet duidelijk welke van deze 2 manieren voordeliger zijn voor het te bouwen ticketing systeem.

**Vue.js:** De kleine community, gebrek aan plug-ins en over flexibiliteit van vue.js zouden een groot risico kunnen vormen op de bouw van een ticketing systeem. Het lijkt vooral geschikt voor hele kleine projecten. Vue.js zal daarom niet verder mee worden genomen in het onderzoek.

**React.js:** Hoewel de grote community de slechte documentatie kan compenseren zou de verwarrende JSX van React.js

**Ember.js:** De nadelen van Ember.js zouden een groot risico kunnen vormen op de bouw van een ticketing systeem. Toch bleek uit de *‘Available product* analysis’ van frameworks in bestaande CRM en ticketing systemen. Dat deze veel gebruikt werd. Hierom zal deze worden meegenomen in het prototyping onderzoek.

**jQuery:** Het is duidelijk dat jQuery vaak een positieve toevoeging kan zijn aan een project. Mocht er binnen een framework een probleem ontstaan welke met behulp van jQuery op te lossen is zal deze dan ook zeker goed ingezet kunnen worden.

# Welke relevante voordelen hebben bepaalde frameworks?

|  |  |
| --- | --- |
| Strategie | Methode |
|  | Prototyping |
|  | Component test |

## Prototyping

Om beter te kunnen bepalen welke van de frameworks er meer of het meest geschikt is voor de bouw van een ticketing systeem zullen deze tegen elkaar afgewogen moeten worden. Prototypes zijn hier uitermate geschikt voor. Daarom zullen er met de overgebleven frameworks een aantal prototypes worden gemaakt om te bepalen welke het meest geschikt is/zijn voor de bouw van een ticketing systeem.

*Binnen het prototyping onderdeel van het onderzoek zal jQuery niet specifiek worden getest. Het is duidelijk dat jQuery een goede toevoeging is aan het project en erg goed in combinatie met de andere framworks en libraries kan worden gebruikt.*

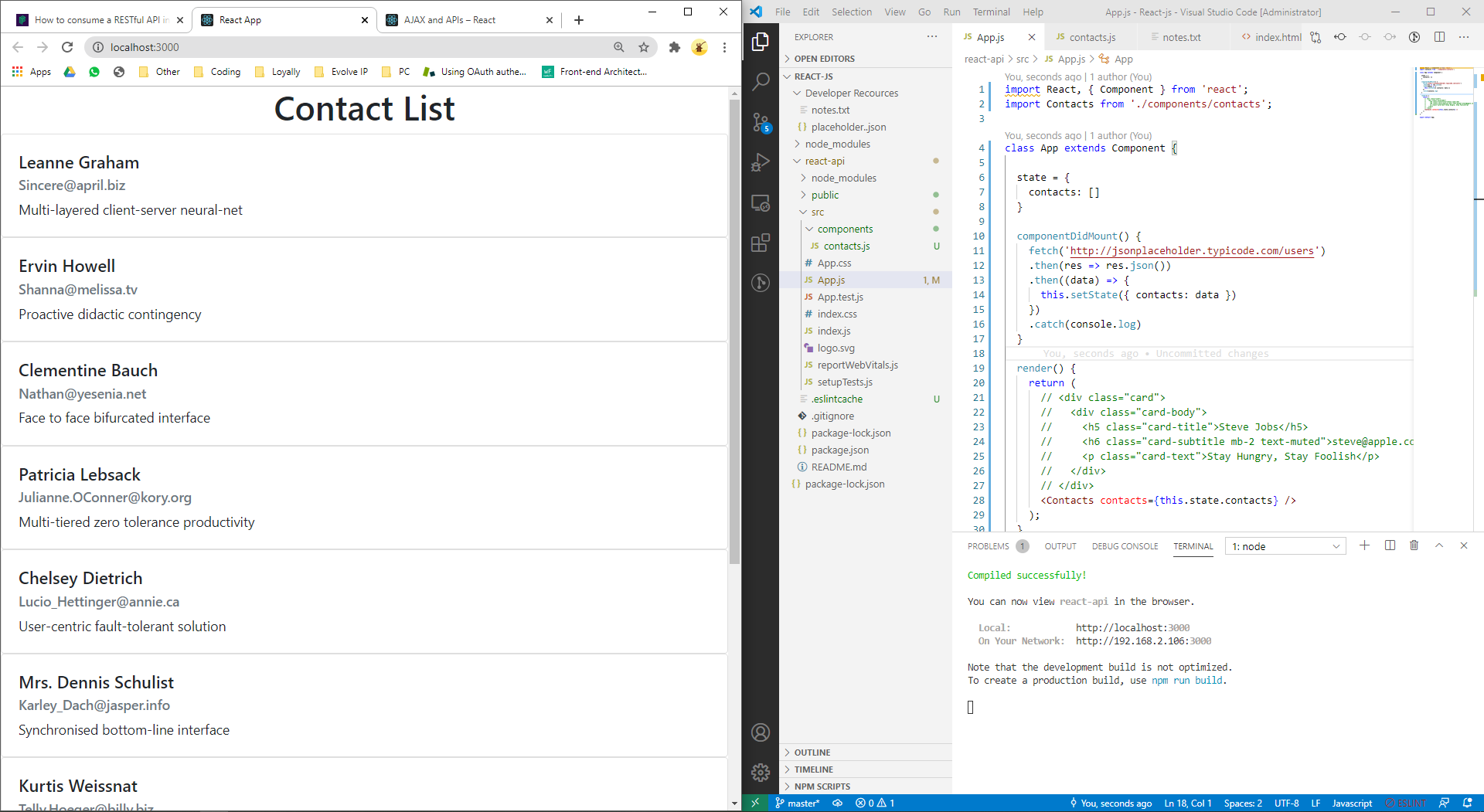
### Hello BACKEND!

Deze eerste prototypes zijn *‘Hello World’* apps gebouwd binnen de overgebleven frameworks: Angular, React.js en Ember.js. Het doel is om iets verder te gaan dan een simpele ‘hello world’ applicatie en te focussen op RESTfull communicatie.

#### Angular

#### React.js

Voor het verkennen van React.js op het gebied van RESTfull communicatie is de volgende tutorial gebruikt: <https://pusher.com/tutorials/consume-restful-api-react>.



*Prototype project.*

**Het volgende stukje code heeft de daadwerkelijke API call uitgevoerd:**

  componentDidMount() {

    fetch('http://jsonplaceholder.typicode.com/users')

    .then(res => res.json())

    .then((data) => {

      this.setState({ contacts: data })

    })

    .catch(console.log)

  }

**Alternatief zou de code in de volgende vorm kunnen worden opgezet.**   
*Op deze manier kan er ook een body worden meegegeven aan het request naar de API.*

  componentDidMount() {

    // Simple POST request with a JSON body using fetch

    const requestOptions = {

        method: 'POST',

        headers: { 'Content-Type': 'application/json' },

        body: JSON.stringify({ title: 'React POST Request Example' })

    };

    fetch('https://jsonplaceholder.typicode.com/posts', requestOptions)

        .then(response => response.json())

        .then(data => this.setState({ postId: data.id }));

}

(react-fetch-http-post-request-examples, sd)

#### Ember.js

# Verwijzingen

(sd). Opgehaald van Full Stack Academy: https://www.fullstackacademy.com/blog/nine-best-programming-languages-to-learn

*Angular Docs*. (sd). Opgehaald van angular.io: https://angular.io/docs

*Apex Developer Guide*. (sd). Opgehaald van https://developer.salesforce.com/docs: https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex\_intro.htm

*Application config*. (sd). Opgehaald van v3.vuejs.org: https://v3.vuejs.org/api/application-config.html

*Atlassian Careers*. (sd). Opgehaald van atlassian.com/company/careers: https://www.atlassian.com/company/careers/all-jobs?team=Engineering%2CInterns&location=&search=

*Aura Components*. (sd). Opgehaald van https://developer.salesforce.com/docs: https://developer.salesforce.com/docs/atlas.en-us.lightning.meta/lightning/intro\_components.htm

*Backbone.js*. (sd). Opgehaald van backbonejs.org: https://backbonejs.org/#

*Differences to React*. (sd). Opgehaald van https://preactjs.com/: https://preactjs.com/guide/v8/differences-to-react/

*Ember.js Tutorial*. (sd). Opgehaald van guides.emberjs.com: https://guides.emberjs.com/release/tutorial/part-1/orientation/

*Freshdesk portal*. (sd). Opgehaald van freshdesk.com: https://lyl.freshdesk.com/a/admin/

*Getting Started*. (sd). Opgehaald van v3.vuejs.org: https://v3.vuejs.org/guide/introduction.html#getting-started

Hamedani, M. (sd). Opgehaald van Programming with mosh: https://programmingwithmosh.com/javascript/top-frontend-frameworks-to-learn-in-2020/

*Happy Fox Jobs*. (sd). Opgehaald van happyfox.recruiterbox.com: https://happyfox.recruiterbox.com/jobs/fk0fuc8/

*HubSpot Portal*. (sd). Opgehaald van app.hubspot.com/contacts: https://app.hubspot.com/contacts/8649879/ticket/219414507

*Insights, Developer Survey*. (sd). Opgehaald van https://insights.stackoverflow.com/: https://insights.stackoverflow.com/survey/2020#technology-most-loved-dreaded-and-wanted-web-frameworks-loved2

*introducing JSX*. (sd). Opgehaald van react.js.org: https://reactjs.org/docs/introducing-jsx.html

*Jira Servicedesk Portal*. (sd). Opgehaald van https://studyresearch.atlassian.net/: https://studyresearch.atlassian.net/jira/servicedesk/projects/GSD/queues/custom/1

*Jitbit Portal*. (sd). Opgehaald van fontysstudent3000.jitbit.com: https://fontysstudent3000.jitbit.com/helpdesk/

*jQuery.get*. (sd). Opgehaald van api.jquery.com: https://api.jquery.com/jQuery.get/

*Lightning Aura Components Developer Guide*. (sd). Opgehaald van https://developer.salesforce.com/docs: https://developer.salesforce.com/docs/atlas.en-us.lightning.meta/lightning/intro\_framework.htm

*Methods*. (sd). Opgehaald van ICT Research Methods: http://ictresearchmethods.nl/Methods

Nemesis Project Plan. (sd).

*Omnidesk vacatures*. (sd). Opgehaald van www.omnidesk.nl/vacatures: https://www.omnidesk.nl/vacatures/medior-developer

*passing-data-through-props*. (sd). Opgehaald van reactjs.org: https://reactjs.org/tutorial/tutorial.html#passing-data-through-props

*Salesforce Portal*. (sd). Opgehaald van https://loyally-dev-ed.lightning.force.com/: https://loyally-dev-ed.lightning.force.com/lightning/o/Account/list?filterName=00B58000003UsJBEA0

Simform. (sd). *https://www.simform.com/best-frontend-frameworks/*. Opgehaald van Simform.

*SOQL and SOSL Reference*. (sd). Opgehaald van https://developer.salesforce.com/docs: https://developer.salesforce.com/docs/atlas.en-us.soql\_sosl.meta/soql\_sosl/sforce\_api\_calls\_soql.htm

*Stack overflow Trends*. (sd). Opgehaald van https://insights.stackoverflow.com/: https://insights.stackoverflow.com/trends?tags=reactjs%2Cvue.js%2Cangular%2Cangularjs%2Crazor%2Cjquery%2Cember.js%2Cmeteor%2Cblazor%2Cbackbone.js

The jQuery Foundation. (sd). *team*. Opgehaald van jquery.org: https://jquery.org/team/

*TOPdesk Journey*. (sd). Opgehaald van https://page.topdesk.com/our-journey-so-far: https://page.topdesk.com/our-journey-so-far

*TOPdesk Portal*. (sd). Opgehaald van https://nltopdesktrial1.topdesk.net: https://nltopdesktrial1.topdesk.net/tas/public/ssp/content/page/myrequests?from=49680e9d-2c0c-4562-b600-56b3640b8cf0

*Trends*. (sd). Opgehaald van Stack overflow: https://insights.stackoverflow.com/survey

*Zendesk Portal*. (sd). Opgehaald van zendesk.com: https://loyallydev.zendesk.com/agent/dashboard

*Zoho Desk Portal*. (sd). Opgehaald van desk.zoho.eu/support/fontysstudent3000: https://desk.zoho.eu/support/fontysstudent3000/ShowHomePage.do#Cases/dv/58108000000107148