Kwetter Sprint documenten

# Projectbeschrijving

Kwetter is een webapplicatie die het voor gebruikers mogelijk maakt snel alles te delen wat in ze opkomt. Een gebruiker kan namelijk een bericht plaatsen van maximaal 160 tekens. Deze tweet wordt dan gedeeld op de tijdlijn van iedereen die deze persoon volgt. Ook kan een gebruiker een andere gebruiker taggen in een tweet. Alle andere gebruikers kunnen dan een tweet liken of erop reageren.

Ook heeft elke gebruiker een profielpagina. Hierop staan, naast een overzicht van alle tweets van deze gebruiker, ook nog andere gegevens als de locatie, website en een korte biografie. Een gebruiker kan ook een profielfoto toevoegen aan zijn/haar account.

Middels de zoekbalk kunnen alle tweets gevonden worden die ooit geplaatst zijn. Door een zoekopdracht in te voeren met het @-teken ervoor kan gezocht worden op andere gebruikers. Je kan er dan voor kiezen deze gebruiker te volgen of de profielpagina te bezoeken.

De frontend van de applicatie gaat gebouwd worden in React/JS. De backend zal bestaan uit verschillende (micro)services. Deze worden geschreven in Java met behulp van het Springboot of Quarkus framework. Voor het direct plaatsen van tweets op de tijdlijn van andere gebruikers zal er eventueel gebruik worden gemaakt van websockets.

Voor de CI/CD van het project wordt gebruik gemaakt van Gitlab. Meer informatie hierover komt een latere sprint.

# Sprint 0 doelen

* Bedenken van een applicatie
* Nadenken over de te gebruiken technieken en frameworks
* Opstellen van een eerste architectuur
* Opzetten van een repository
* Event storming toepassen
* Backlog opzetten middels Trello

# Sprint 0 prestaties

* Kwetter casus uitgedacht
* Eerste keuze gemaakt in de te gebruiken technieken (ReactJS, Java, Springboot of Quarkus)
* Architectuur van frontend naar backend (microservices) uitgetekend
* Git repository geïnitialiseerd
* Event storming toegepast en uitgetekend
* Trello backlog staat

# Sprint 1 doelen

* Opzetten architectuur applicatie
* API gateway opzetten
* Profielservice maken
* Profiel frontend maken

# Sprint 1 prestaties

* Alle backendservices en databases staan
* CI/CD pipelines draaien (deployment naar devops)
* API gateway opgezet
* Profielservice zowel frontend als backend gemaakt
* Backend van followservice en tweetservice grotendeels gemaakt

# Sprint 2 doelen

* Authenticatie & Verificatie
* Inloggen en Uitloggen in frontend
* Tweet sturen
* Tweets zien op profielpagina
* Tweet verwijderen
* Automatisch pushen naar DockerHub
* Docker container resource managen met Kubernetes

# Sprint 2 prestaties

* Backend service opgezet voor authenticatie en verificatie
* Inloggen en uitloggen Frontend en Backend
* Bearer tokens worden opgeslagen in localstorage
* Registreren
* Alle Lokale databases omgezet naar Azure databases
* Tweetcomponent gemaakt voor het versturen van tweets (indien token klopt)
* Tweets zien op profielpagina
* Profiel aanpassen (indien eigen profiel en token klopt)
* Tweet verwijderen (indien eigen tweet en token klopt)
* Dockerhub repositories aangemaakt
* Dockerhub profileservice automatisch gepusht
* Profielservice in Kuburnetes voor resource management

# Sprint 3 doelen

* Automatisch deployen naar DockerHub voor alle services
* Messaging toepassen
* Notificatieservice aanmaken
* Volgen en ontvolgen
* Bugs fixen
* Onderzoeken
* Testen

# Sprint 3 prestaties

* Alle microservices staat in worden automatisch deployed naar DockerHub
* Alle services werken in Docker (bugs met dependencies gefixt)
* RabbitMQ messaging werkt (tussen profileservice en notificationservice)
* Gebruikers kunnen andere gebruikers volgen en ontvolgen
* Automatisch profiel aanmaken bij registratie aangepast
* C4 modellen gemaakt
* Functional requirements uitgebreid en definitions of done bepaald

# Sprint 4 doelen

* Testen, unit tests, integration tests, system tests
* Security document maken
* PDR uitbreiden
* Feedback toepassen

# Sprint 4 prestaties

* Testen geschreven voor verschillende services
* Code coverage rapport in pipeline
* Pipelines hersteld door contact met Azure
* Security document opgesteld
* PDR uitgebreid na feedback

# Sprint 5 doelen

* Nog meer testen
* Applicatie afronden
* PDR afronden
* Applicatie volledig draaien in Azure Kubernetes

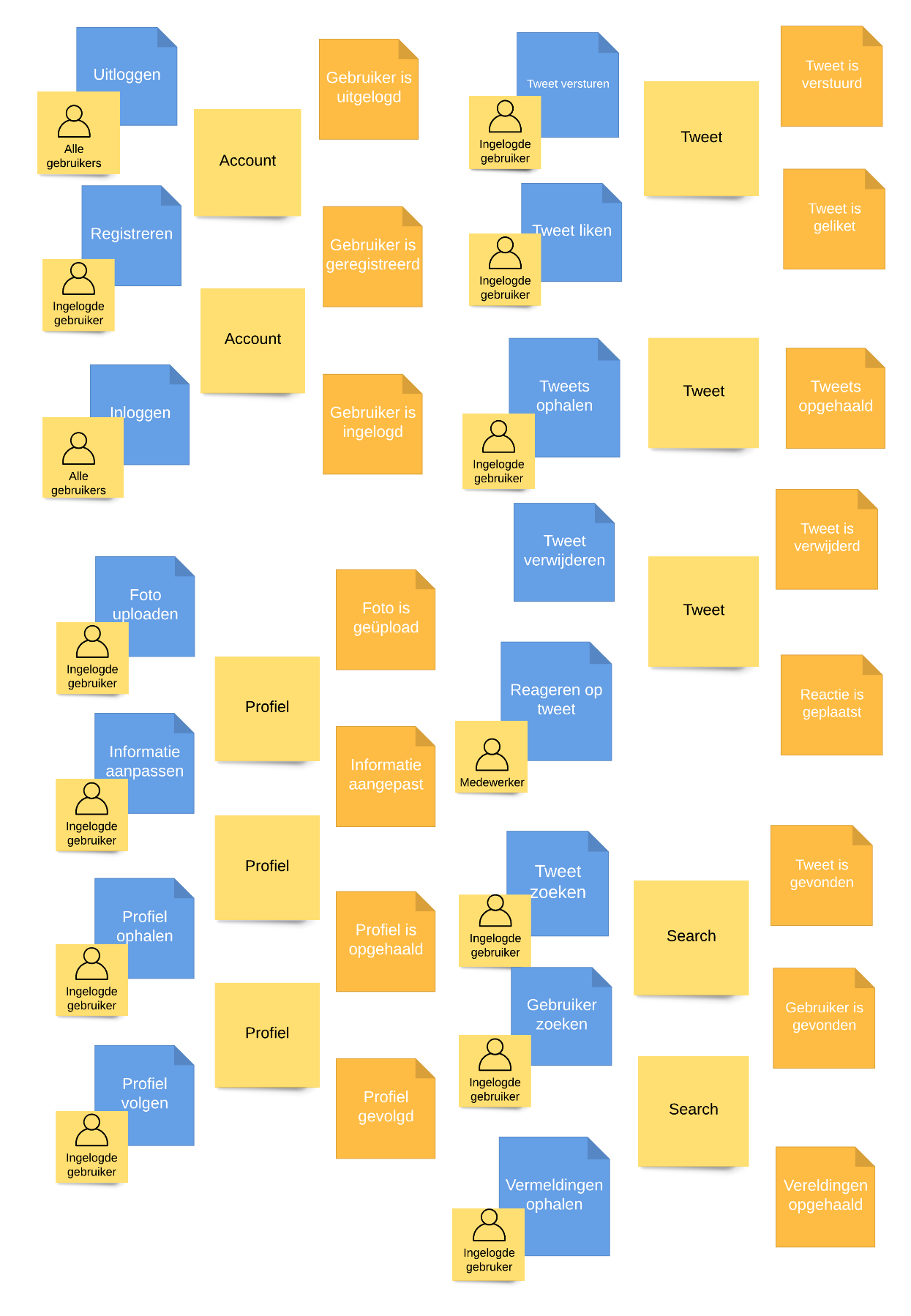
# Sprint 5 prestaties

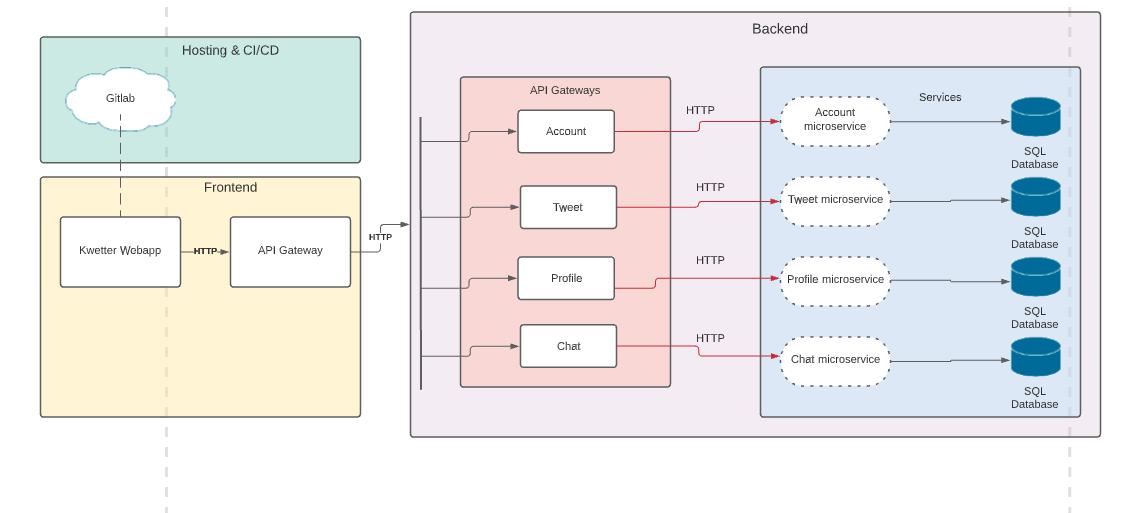
* Alle backend services draaien in Kubernetes in een Azure Kubernetes cluster
* PDR uitgebreid en volledig afgerond
* Laatste case study over Blockchain gemaakt
* Veel feedback gevraagd en toegepast
* Helaas geen testen meer kunnen schrijven

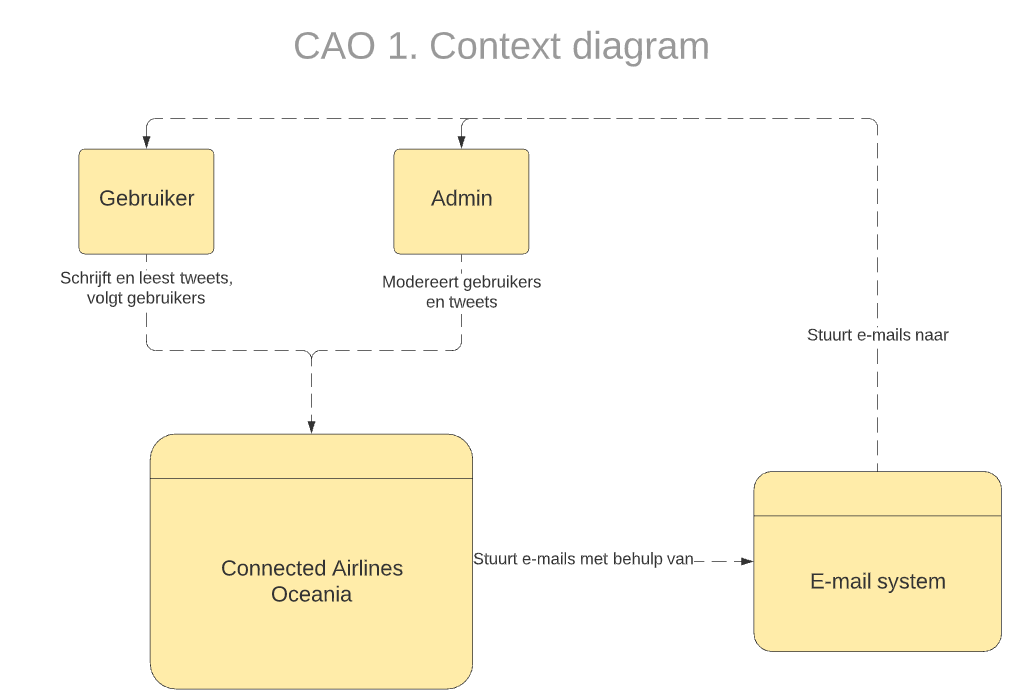
# Trello link

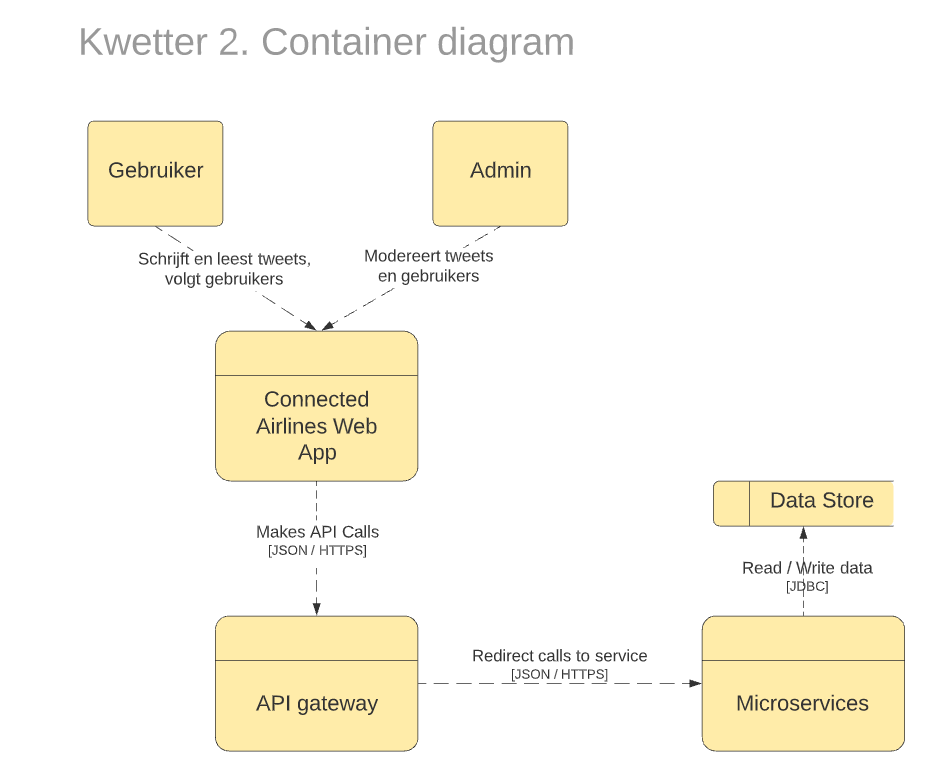
<https://trello.com/invite/b/F6o64jrq/7aec25f909c87acaec62fbac355c8795/semester-6-kwetter>

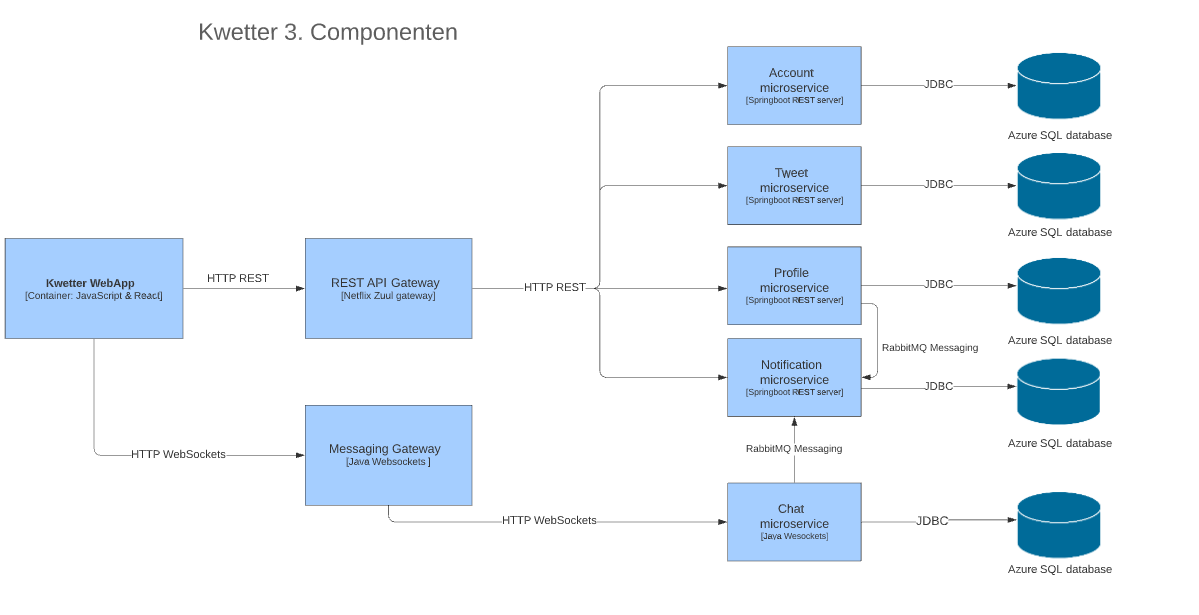
# Architectuur ontwerpen











# Functional requirements

|  |  |  |  |
| --- | --- | --- | --- |
| **Functionaliteit** | **Gebruikersrol** | **User Story ID** | **Omschrijving** |
| Tweet sturen | Gebruiker | 1 | Gebruiker kan een tweet sturen |
| **Definition of done:**   * Ingelogde gebruiker krijgt een invoerveld te zien * Invoerveld accepteert alleen tekst en andere unicode karakters * Er wordt gebruik gemaakt van input filtering (indien nodig, onderzoek) * Er wordt een melding getoond wanneer de tweet verstuurd is of als er iets fout gaat * De tweet wordt in de database opgeslagen | | | |
| Tweet verwijderen | Gebruiker | 2 | Gebruiker kan een tweet verwijderen |
| **Definition of done:**   * Ingelogde gebruiker krijgt een “tweet verwijderen” knop te zien * Alleen de eigenaar van een tweet kan de tweet verwijderen * De gebruiker krijgt een melding dat de tweet verwijderd is * De tweet wordt volledig uit de database verwijderd (niet middels een property niet weergegeven) | | | |
| Tweet liken | Gebruiker | 3 | Gebruiker kan een tweet liken |
| **Definition of done:**   * Ingelogde gebruiker krijgt een “tweet liken” knop te zien (kan ook een duim zijn of iets dergelijks) * De eigenaar van een tweet kan niet zijn eigen tweet liken * De liker van de tweet krijgt visuele confirmatie dat de tweet geliket is * Een tweet kan nooit meerdere malen geliket worden door een gebruiker * De eigenaar van de tweet krijgt een notificatie dat de tweet een like heeft * De combinatie tweet-liker van de tweet wordt opgeslagen in de database * Het aantal likes van de tweet wordt in de database met 1 verhoogd | | | |
| Gebruiker volgen | Gebruiker | 4 | Gebruiker kan een andere gebruiker volgen |
| **Definition of done:**   * Ingelogde gebruiker krijgt een “profiel volgen” knop te zien op een profielpagina * Een gebruiker kan nooit zijn eigen profiel volgen * De gebruiker krijgt een melding dat hij nu het andere profiel volgt * De gebruiker die hoort bij het profiel wat gevolgd wat krijgt een notificatie dat hij een nieuwe volger heeft * De volgers en volgende worden op beide profielen getoond * De combinatie volgende-volger wordt opgeslagen in de database | | | |
| Gebruiker ontvolgen | Gebruiker | 5 | Gebruiker kan een andere gebruiker ontvolgen |
| **Definition of done:**   * Ingelogde gebruiker krijgt een “profiel niet meer volgen” knop te zien als hij een profiel bezoekt dat hij al volgt * Deze knop zal nooit zichtbaar zijn wanneer een gebruiker zijn eigen profiel bekijkt * De gebruiker krijgt een melding dat hij het profiel niet meer volgt * De combinatie volgende-volger wordt verwijderd uit de database * Het is altijd mogelijk om toch weer opnieuw een gebruiker te volgen | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Tijdlijn inzien | Gebruiker | 6 | Gebruiker kan alle tweets zien van gebruikers die hij volgt |
| **Definition of done:**   * De gebruiker krijgt op de hoofdpagina een overzicht met alle tweets * De lijst bestaat uit de laatste x (100?) tweets * De lijst is chronologisch gesorteerd * De lijst bestaat alleen uit tweets van mensen die gevolgd worden door de gebruiker * De lijst kan altijd uitgebreid worden met oudere tweets door naar beneden te scrollen | | | |
| Zoeken door tweets | Gebruiker | 7 | Gebruiker kan een specifieke tweet terugzoeken |
| **Definition of done:**   * De gebruiker krijgt een invoerveld te zien * Invoerveld accepteert alleen tekst en andere unicode karakters * Er wordt gebruik gemaakt van input filtering (indien nodig, onderzoek) * De gebruiker krijgt een overzicht van alle tweets die de zoekterm bevatten (max 100) * Tweets van mensen de gebruiker volgt staan bovenaan, verder wordt er chronologisch gesorteerd | | | |
| Mentions inzien | Gebruiker | 8 | Gebruiker krijgt een overzicht van al zijn mentions |
| **Definition of done:**   * De gebruiker ziet de knop “alle mentions inzien” * De gebruiker ziet de laatste 100 keer dat hij genoemd is in een tweet * De lijst van alle tweets wordt chronologisch gesorteerd | | | |
| Reageren op tweet | Gebruiker | 9 | Gebruiker kan reageren op een tweet |
| **Definition of done:**   * De gebruiker ziet een “reageren op tweet” knop (Kan ook een pijltje terug zijn of iets dergelijks) * De tweet krijgt automatisch de gebruikersnaam van de eigenaar van de oorspronkelijke tweet * De tweet wordt verder afgehandeld als een normale tweet * De eigenaar van de oorspronkelijke tweet krijgt een melding dat hij een reactie heeft ontvangen | | | |
| Inloggen | Gebruiker | 10 | Gebruiker kan inloggen |
| **Definition of done:**   * De gebruiker kan inloggen met zijn geregistreerde account d.m.v. email en wachtwoord. * De gebruiker krijgt een melding te zien als er verkeerde credentials ingevoerd zijn. * Bij succesvol inloggen wordt de gebruiker doorverwezen naar de homepagina * De ingelogde gebruiker krijgt een (bearer) token toegewezen die gebruikt kan worden om zichzelf te verifiëren bij alle aankomende api calls | | | |
| Uitloggen | Gebruiker | 11 | Gebruiker kan uitloggen |
| **Definition of done:**   * De gebruiker kan uitloggen middels de uitlog knop * De toegewezen sessie token wordt verwijderd * De inlogknop is nu opnieuw zichtbaar | | | |
| Profiel aanpassen | Gebruiker | 12 | Gebruiker kan profiel aanpassen |
| **Definition of done:**   * De gebruiker dubbelklikt op het attribuut wat aangepast moet worden * De gebruiker krijgt een invoerveld en vult de juiste gegevens in * Invoerveld accepteert alleen tekst en andere unicode karakters * Er wordt gebruik gemaakt van input filtering (indien nodig, onderzoek) * Een gebruiker kan alleen zijn eigen profiel aanpassen | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Profiel inzien | Gebruiker | | 13 | | Gebruiker kan een profiel inzien |
| **Definition of done:**   * De gebruiker klikt op de naam van een gebruiker en wordt naar het bijbehorende profiel geleidt * Het profiel geeft de volgende persoonlijke informatie weer: gebruikersnaam, woonplaats, biografie en website * Verder worden de tweets, volgers en volgende van de gebruiker weergegeven * Op deze pagina zal de “gebruiker volgen” knop zichtbaar zijn | | | | | |
| Registreren | | Gebruiker | | 14 | Gebruiker kan een account aanmaken |
| **Definition of done:**   * Klant kan zelfstandig een account registreren. * Klant moet zijn persoonsgegevens kunnen invullen. * Er wordt een melding getoond als het ingevoerde wachtwoord niet voldoet aan de eisen. * Alles gegevens zijn op een veilige manier opgeslagen * De gebruikersnaam en het e-mailadres mogen maar een keer voorkomen in de database * Wanneer de gebruiker een al bestaand e-mailadres of gebruikersnaam gebruikt krijgt hij hiervan een melding | | | | | |

# CI/CD Link

<https://dev.azure.com/382763/Kwetter>

<https://hub.docker.com/repository/docker/kevints98/profileservice>

Dit project zou “Openbaar” moeten staan. Mochten jullie er niet in kunnen kan ik jullie ook even uitnodigen.

# Github link

<https://github.com/jeBoyKevin/kwetter>

# Sprint 1 retrospective

Ik heb deze tweede sprint flinke stappen gemaakt op het gebied van opzet van de backend en het implementeren van de eerste user stories. Ook heb ik nagedacht over het gebruik van Docker en heb ik een idee hoe ik messaging toe wil gaan passen binnen dit project. Ik moet wel opletten dat ik meer op de learning outcomes let en minder op de functionele eisen van het project.

# Sprint 2 retrospective

Ik heb deze derde sprint vooral gewerkt aan de leerdoelen. Zo ben ik opgeschoten met het leerdoel over cloud services door gebruik te maken van de Azure Portal voor mijn databases. Ook ben ik al bijna klaar met het automatisch pushen van de laatste versie van een microservice naar de Dockerhub en deze dan met behulp van Kuburnetes te schalen. Ook heb ik gewerkt aan authenticatie en verificatie door met behulp van JWT en bearer tokens elke belangrijke request eerst te verifieren.

Ik heb deze sprint als zeer nuttig ervaren en heb ook twee keer met Hans gesproken en een maal met Kiavash. Door meer feedback te vragen krijg ik beter inzicht in hoe ik er voor sta met mijn project.

# Sprint 3 planning

Op de Trello heb ik een planning gemaakt van taken die ik in sprint 2 uit wil gaan voeren.

Verder wil ik mij volgende sprint gaan richten op de leerdoelen Cloud Services en Security by Design. Voor het eerste leerdoel ga ik onderzoek naar hoe ik messaging en logging het beste toe kan passen binnen mijn project.

Voor security by design wil ik de owasp top 10 afgaan en opzoek gaan naar eventuele fouten in mijn project tot op dit punt, maar ook informatie krijgen over waar ik rekening mee moet houden bij het verder werken.

# Sprint 3 retrospective

Ik heb de afgelopen sprint Devops volledig werken inclusief deployment naar docker en resources scalen met behulp van kubernetes. Ook heb ik messaging toegepast om via de notificatieservice een gebruiker te laten weten dat er iets gebeurd als "een nieuwe volger" of je nieuwe tweet heeft x likes. Ik heb deze leerdoelen aangetoond na eerst onderzoek te doen.

Ik ben zeer tevreden over wat ik de afgelopen sprint bereikt heb, ik heb eerst veel onderzoek gedaan naar wat de juiste manier van werken is en hoe ik bepaalde technieken het beste kan toe passen. Pas na deze onderzoeken ben ik begonnen met programmeren en deze manier van werken past goed bij mij. Hans en Kiavash waren ook zeer tevreden afgelopen week in onze gesprekken.

# Sprint 4 planning

Op de Trello heb ik een planning gemaakt van taken die ik in sprint 4 uit wil gaan voeren.

Verder wil ik mij volgende sprint gaan richten op de leerdoelen Security by Design en scalable architectures.

Voor security by design wil ik de owasp top 10 afgaan en opzoek gaan naar eventuele fouten in mijn project tot op dit punt, maar ook informatie krijgen over waar ik rekening mee moet houden bij het verder werken.

Voor scalable architectures heb ik van Hans te horen gekregen dat ik dit leerdoel het beste aan kan tonen door een extra service te maken en als ik deze zonder code aan te passen binnen andere services werkend krijg, ik dit eigenlijk al heb laten zien.

# Sprint 4 retrospective

Deze sprint heb ik een aantal grote stappen gezet met mijn PDR. Ik had hier namelijk de feedback op gekregen dat deze veel uitgebreider moest. Deze feedback zal ik in sprint 5 ook nog moeten verwerken, maar ik ben van mening dat er dan een mooi eindproduct staat. De applicatie zelf is zo goed als af, wanneer ik volgende sprint nog wat testen kan schrijven en de applicatie volledig online draai met behulp van Kubernetes heb ik een mooi eindproduct neergezet

# Sprint 5 planning

Sprint 5 zal in het teken staan van afronden. Ik wil nog wat meer testen schrijven en de applicatie laten draaien in Kubernetes. Ook zal ik deze sprint een demo geven aan de docenten en de laatste feedback hiervan toepassen

# Sprint 5 retrospective

Ik heb deze sprint de applicatie bijna volledig afgerond. Hij is publiekelijk te bereiken middels de volgende URL: <http://kwetterreact.azurewebsites.net/>

De microservices draaien met behulp van Kubernetes en Azure online en zijn ook schaalbaar. Hier ben ik zeer tevreden over. Ik vind het jammer dat ik er niet meer aan toe ben gekomen meer en beter te testen. Dit is zeker een aandachtspuntje voor een volgende project.