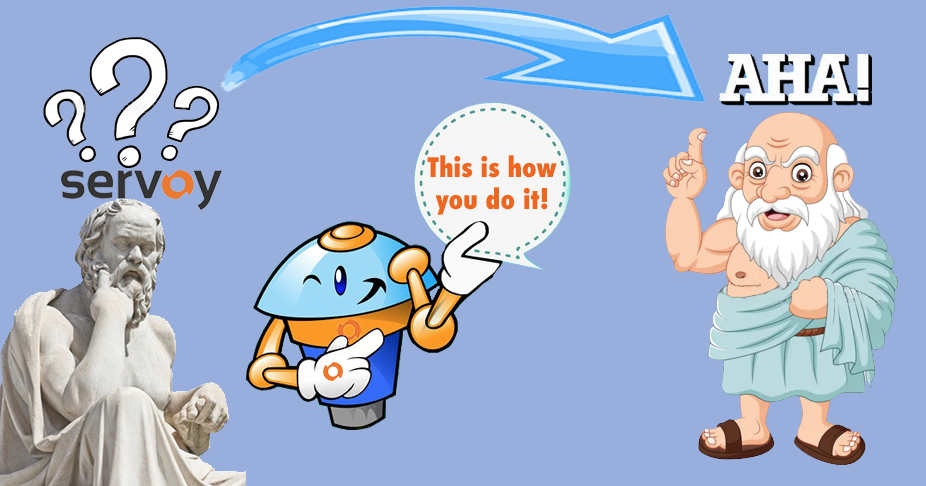
Scriptie

“Chatbot implementatie bij Servoy”



**Bedrijfsnaam** Servoy

**Auteur** Kostis Thanos

**Afstudeerbegeleider** Gerald Stap

**Stagebegeleider** Ron van der Burg

Scriptie

“Chatbot implementatie bij Servoy”



**Auteur** K. Thanos

**Studentnummer** 500739944

**Telefoonnummer** 0619157950

**Plaats** Amsterdam  
**Datum** 12-11-2019

**Onderwijsinstelling** Hogeschool van Amsterdam

**Opleiding & Studierichting** HBO-ICT : Software Engineering

**Begeleidend Docent** Gerald Stap

**Bedrijf** Servoy, Software Development, Fred. Roeskestraat 97

Amsterdam, 020 229 1150

**Bedrijfsbegeleider** Ron van der Burg

**Stageperiode** november 2019 - juni 2019

**Voorwoord**

Vanaf november 2019 t/m juni 2020 heb ik bij Servoy mijn afstudeerstage kunnen lopen om zo mijn studie Software Engineering op een mooie manier af te sluiten. Gedurende deze periode heb ik onderzoek gedaan naar de functionaliteiten die passen bij het platform van Servoy en heb deze vervolgens ook geïmplementeerd.

Ik wil graag de volgende mensen bedanken voor hun begeleiding en assistentie tijdens mijn onderzoek:  
  
Victor die mij de basistraining van Servoy gaf, waardoor ik net iets beter aan de slag kon met de Servoy IDE.

Manfred, Alex en René voor hun technische kennis op het gebied van uiteraard Servoy zelf maar ook andere gebieden zoals databases en REST.  
  
Natuurlijk Ron van der Burg voor zijn begeleiding, zijn perspectief op het grote plaatje en zijn technische inzicht.

En ook de rest van het team, waar ik toch een klein half jaar werkervaring mee heb opgedaan.

**De inhoudsopgave**

[**Samenvatting**](#_qavadhxvc71b)**…….….….….….…….….….….….…….….….….….…….….….….….……..….. 3**

[**1. Inleiding**](#_oh3gn2nb0b56)**….….….….….…….….….….….…….….….….…….….….….….…….….….….…. 4**

[**2. Context van de opdracht**](#_loe1po9vw4bt)**….….….….….…….….….….….…….….….….….…….….….…. 6**

[**3. Onderzoeksmethode beschrijving en – verantwoording**](#_k54as0z88cv2)**….….….….….…….….….…. 9**

[**4. Plan van aanpak**](#_cv4o26qas8fl)**….….….….….…….….….….….…….….….….….…….….….….….……. 11**

[**5. Evolutie van de Chatbot**](#_87937j55rh9b)**….….….….….…….….….….….…….….….….….…….….….…. 15**

[**6. Huidige omgeving**](#_3629u718v0qq)**….….….….….…….….….….….…….….….….….…….….….….….…. 17**

[**7. Theoretische Achtergrond**](#_fiau8ky4uk6p)**….….….….….…….….….….….…….….….….….…….….…. 18**

[**8. Uitwerking Hoofd- en Deelvragen**](#_rvisdpch8juw)**….….….….….…….….….….….…….….….….….…... 22**

[8.1 Algemene chatbot functionaliteiten die bestaan](#_9z3v5dqj7usg)……...……...……...……...……...… 22

[8.2 Wat zijn de problemen die nu worden ondervonden door de gebruikers bij het zoeken naar informatie?](#_olpgeeaf6d4j) ……...……...……...……...……...……...……...……...……...……...…….26

[8.3 Keuze Van Functionaliteiten](#_aro8skc88lzv) ……...……...……...……...……...……...……...……...…28

[8.4 Frameworks en AI-engines beschikbaar in de markt](#_b6p8ctqnwf5) ……...……...……...……...…….30

[8.5 Welk framework is het meest compatibel met Servoy?](#_k595qkk52yk) ……...……...……...……..…32

[**9. Ontwerp en realisatie van het product**](#_hc9gc3lgnn1c)**….….….….….…….….….….….…….….….….… 37**

[9.1 Curatie Applicatie](#_4p1nkhmhn17n)……...……...……...……...……...……...……...……...……...………. 37

9.2 Watson met Elasticsearch...……......……......……......……......……......……......……38  
[9.3 Werking van DeepPavlov](#_vbunqx39fvlq) ……...……...……...……...……...……...……...……...……..42

[9.4 De Chatbot](#_zcr6efjwl6c6) ……...……...……...……...……...……...……...……...……...……...……....46

[9.5 Performance tests](#_osb4n8gmcjc8) ……...……...……...……...……...……...……...……...……...……...48

[**10. Watson Implementatieproblemen**](#_19s2pw5l1j0i)**.….….….….….…….….….….….…….….….….….…**[**49**](#_19s2pw5l1j0i)

[**11. Resultaten**](#_743hbbce5smd) **….….….….….…….….….….….…….….….….….…….….….….….…….….…50**

[**12. Evaluatie**](#_8wgmr2obxxn7) **….….….….….…….….….….….…….….….….….…….….….….….…….….….51**

[**13. Conclusies**](#_gcmkhyizbych) **….….….….….……...….….….….…….….….….….…….….….….….…….….53**

[**14. Aanbevelingen**](#_fzv0o1oxytev) **….….….….….…….….….….….…….….….….….…….….….….….…….54**

[**15. Bronnenlijst**](#_goa5d62rm3y8) **….….….….….…….….….….….…….….….….….…….….….….….…….….56**

[**16. BegrippenLijst**](#_7ossi41qz9nz) **….….….….….…….….….….….…….….….….….…….….….….….…….59**

[**17. Bijlagen**](#_bf46hza8hug7) **….….…....….….…….….….….….…….….….….….…….….….….….…….….….61**

# **Samenvatting**

Het bedrijf Servoy gebruikt een eigen ontwikkelaarsplatform waarmee klanten snel hun applicaties kunnen bouwen met relatief weinig code. Hiervoor worden eigen keywords en klassen gebruikt waar de klant mee moet leren werken. Het probleem wat veel klanten aangeven is dat ze in de documentatie en het forum moeilijk antwoorden kunnen vinden op vragen en problemen die ze ondervinden tijdens het gebruik van Servoy.  
  
De probleemstelling van het onderzoek luidt dan ook als volgt:   
“Hoe kan een nieuwe chatbotfunctie zo goed mogelijk worden geïmplementeerd voor het

programma Servoy, zodat de gebruikers de informatie sneller kunnen vinden?”  
  
Om die reden is de opdracht gegeven om een chatbot te creëren die dit probleem kan verhelpen. Hiervoor wordt er eerst kwalitatief onderzoek gedaan in de vorm van deskresearch. Het literatuuronderzoek is verricht om inzicht te krijgen in de onderwerpen van: Natural Language Processing, Neural Networks, zoekmachines en Chatbot Stacks. Maar vooral de soort functionaliteiten die een chatbot kan hebben. Vervolgens worden daaruit degene gekozen die het beste passen bij de behoeften van de gebruikers.  
  
Vervolgens wordt er gekeken naar de chatbot frameworks op de markt**.** Daaruit wordt één open source framework en één betaald framework gekozen. Deze twee worden deels bepaald door de overeenkomst met de gekozen functionaliteiten maar ook door interviews en dialoog met de medewerkers en professionals op het gebied van chatbots.  
  
Als laatst wordt er kwalitatief praktijkonderzoek verricht om de prestatie van de geïmplementeerde chatbot te toetsen. En wordt er algemeen getest, maar ook op gebieden van syntax en semantiek.   
  
Helaas valt het betaalde framework af door complicaties. De resultaten van het opensource framework zijn echter positief verrassend. De implementatie weet bijna alle testen succesvol te doorstaan. Ook vervult het bijna alle requirements en ziet er dus veelbelovend uit.  
  
Om door te gaan zal er sterkere hardware nodig zijn om de geavanceerde algoritmes te ondersteunen.

# **1. Inleiding**

Chatbots zie je tegenwoordig overal, van autowebsites tot energieleveranciers, internetproviders en nog veel meer, kortom ze zijn overal te vinden op het web. Deze bots zijn vooral de laatste 10 jaar heel populair geworden. Een chatbot bespaart namelijk manuren en kan bovendien ook altijd bereikbaar zijn t.o.v. een werknemer die 9 tot 5 werkt. Door grotendeels eenmalig te investeren in de chatbot heeft een bedrijf meteen een assistent die vierentwintig uur per dag klaarstaat voor alle klanten van een bedrijf.  
  
Een eigenschap van chatbots is dat kan ook data opslaan van de gesprekken en kan ook op eigen houtje conclusies trekken, afhankelijk van de complexiteit van de algoritmes die de bot gebruikt. Hiermee kan het bedrijf inzicht krijgen in de gebruikersbehoeften, effectiviteit en andere zaken die allemaal kunnen helpen bij het verbeteren van de effectiviteit van het bedrijf en de gebruikerservaring.  
  
Vaak heeft een bedrijf dat producten verkoopt ook zijn eigen forum waarop mensen vragen kunnen stellen en in gesprek kunnen gaan over de producten en diensten. Door deze communicatie begrijpen de gebruikers hun product beter en leren ze het optimaal te gebruiken.  
  
Gedurende de periode november tot en met juni 2020 heb ik in Amsterdam-Zuid stage gelopen bij het bedrijf Servoy. Ik kwam bij Servoy via de website van Stagemax. De stage heb ik gevolgd als student Software Engineering bij de Hogeschool van Amsterdam.  
Het doel van de eindstage was om te bewijzen dat ik klaar ben voor het bedrijfsleven in de ICT-sector en het leveren van een wenselijk product voor het bedrijf en het verbeteren van mijn competenties.

Het bedrijf Servoy heeft ook zijn eigen product, namelijk het Servoy platform, waarmee mensen snel en eenvoudig bedrijfsapplicaties kunnen opzetten. Servoy heeft daarbij ook zijn eigen forum, maar heeft aangegeven behoefte te hebben aan een chatbot. De chatbot is bedoeld om de vragen van de mensen te beantwoorden zodat die zelf niet lang informatie hoeven op te zoeken in het forum waardoor hun gebruiksgemak wordt verbeterd.

Dit project is opgesteld om op een onderbouwde en gecontroleerde manier de opdracht van het bedrijf uit te voeren om tot een oplossing te komen voor dit probleem wordt er eerst deskresearch gedaan. Daarna wordt fieldresearch gedaan om de haalbaarheid van de implementatie en de effectiviteit van de implementatie te testen.

**Leeswijzer**  
Hoofdstuk 1 is de inleiding van de scriptie.  
Hoofdstuk 2 beschrijft de context van de opdracht, de hoofd- en deelvragen en de op te leveren producten.  
Hoofdstuk 3 beschrijft de onderzoeksmethoden, het type onderzoek en het minimum viable product.  
Hoofdstuk 4 zet het onderzoeksplan uiteen.  
Hoofdstuk 5 geeft een korte geschiedenis van de chatbot en de evolutie daarvan.  
Hoofdstuk 6 geeft de huidige situatie van het bedrijf weer wat betreft de componenten waarmee gewerkt zal worden.  
Hoofdstuk 7 is een diepere theoretische achtergrond over de werking van chatbots.  
Hoofdstuk 8 geeft antwoord op de hoofd- en deelvragen. De gebruikersproblemen worden weergegeven en omgezet in requirements. Ook is er een analyse en keuze van de best passende functionaliteiten en best passende framework.  
Hoofdstuk 9 laat het ontwerp en het realisatie van het product zien.  
Hoofdstuk 10 gaat over de ondervonden problemen met het Watson framework.  
Hoofdstuk 11 laat de resultaten en prestaties van de gecreëerde bot zien.   
Hoofdstuk 12 omvat de evaluatie van de resultaten  
Hoofdstuk 13 bevat de conclusie van het onderzoek.  
Hoofdstuk 14 bevat de aanbevelingen voor het bedrijf.  
Hoofdstuk 15 is de bronnenlijst.

Hoofdstuk 16 is de begrippenlijst.  
Hoofdstuk 17 bevat de bijlagen.

# **2. Context van de opdracht**

* *In dit hoofdstuk wordt de context van de opdracht besproken. Hier worden onder andere de hoofd- en deelvragen genoemd. Ook wordt de impact die het zal hebben op de organisatie genoemd en de producten die moeten worden opgeleverd.*  
  Een bedrijf wil natuurlijk altijd zorgen dat een product optimaal gebruikt kan worden. Er zijn verschillende manieren om dit te realiseren, zo kan een bedrijf gebruikershandleidingen schrijven, of forums creëren waar de gebruikers met elkaar kunnen praten.

Het kan ook zo zijn dat deze functionaliteiten op een gegeven moment niet adequaat genoeg meer zijn voor de behoefte. Voor deze functionaliteiten is dan vervanging vereist.

**Het bedrijf Servoy**De opdracht waar dit rapport voor gemaakt is, is geformuleerd vanuit het bedrijf Servoy.Servoy werd in 2001 opgericht door Jan Aleman met zijn studenten en heeft zijn hoofdkantoren in de Verenigde Staten en Nederland met 4 extra kantoren wereldwijd en distributeurs en partners over de hele wereld. Servoy was tot tweemaal toe winnaar van de Deloitte Technology Fast 50 award, die de vijftig snelst groeiende technologiebedrijven en de meest innovatieve startups van Nederland. Het product Servoy levert een manier om cloudgebaseerde zakelijke applicaties met zo min mogelijk code overal en op elk apparaat te ontwikkelen en te implementeren.

Het Servoy-platform bestaat uit de basislaag en de geïntegreerde ontwikkelomgeving.

De basislaag, of kern, bevat client- en server technologieën en de kaders die worden gebruikt om gemeenschappelijke patronen te implementeren (“About Servoy”, z.d.).  
Patronen kunnen bijvoorbeeld multi tenancy, meertaligheid, beveiliging, navigatie zijn. Het zijn de bouwstenen die gebruikt worden om een nieuwe applicatie snel te assembleren. De basislaag integreert met bestaande technologieën. Het verbindt met databases, webservices en meer met behulp van de Servoy API. Het kan ook webservices beschikbaar stellen.  
  
Servoy biedt dus een meer geautomatiseerde manier om applicaties te ontwikkelen en heeft zijn eigen programmeertaal die werkt op javascript. Er worden vanuit Servoy trainingen gegeven om goed overweg te kunnen gaan met de IDE en zijn functies/Om de programmeertaal en de structuur van het platform te leren.  
 **Opdrachtomschrijving**

De omschrijving van de opdracht vanuit het bedrijf is als volgt:

Servoy is een software ontwikkelplatform waarmee complexe applicaties te bouwen zijn. Omdat het platform zich focust op complexe applicaties zijn featureset en mogelijkheden enorm. Het platform is gedocumenteerd op een publieke wiki en er is een digitaal forum waarin Servoy en haar klanten communiceren over gebruik van het platform.

Van nieuwe en bestaande klanten heeft Servoy al vaak de feedback gekregen dat informatie soms lastig te vinden is. Met name over hoe het platform te gebruiken en wanneer welke oplossing het beste is.

De oplossingsrichting hiervoor is het bouwen van een chatbot die gevoed wordt door de wiki alsmede de vraag-antwoord discussies op het forum. Deze chatbot zou dan beschikbaar moeten komen in de IDE.

Onderzoek zal onder andere gedaan worden in de vorm van: interviews en onderzoek op het internet naar implementatiemogelijkheden. De opdracht heeft een duur van 5 à 6 maanden.

**Doelstelling**

Uit de probleemomschrijving is de volgende doelstelling afgeleid: Het doel van het onderzoek is erachter te komen hoe de chatbot het beste geïmplementeerd kan worden zodat de gebruikers alleen bij nieuwe vragen naar het forum gaan.

**Analyse van Opdracht**

Het doel van het onderzoek is om er achter te komen wat de huidige problemen zijn zonder dat er een chatbot is en op welke manier er een chatbot functie kan worden geïmplementeerd die de huidige probleemsituatie kan verhelpen of verlichten. Om de chatbot te bouwen waar het bedrijf naar op zoek is, is het belangrijk om eerst de juiste vragen te stellen.  
  
**Probleemstelling**  
De probleemstelling is:   
“Hoe kan een nieuwe chatbotfunctie zo goed mogelijk worden geïmplementeerd voor het

programma Servoy, zodat de gebruikers de informatie sneller kunnen vinden?”  
  
**Hoofd- en Deelvragen**  
De hoofdvraag is:

* Welke functionaliteiten van autonome chatbots zorgen ervoor dat klanten van servoy zo snel mogelijk de juiste informatie kunnen vinden?

Hierbij zijn de volgende deelvragen opgesteld:

* Welke functies/functionaliteiten kan een chatbot hebben?
* Wat zijn de problemen die nu worden ondervonden door de gebruikers bij het zoeken naar informatie?
* Welke frameworks / AI-engines zijn er beschikbaar in de markt?
* Welk framework/bestaande AI-engine is het meest compatibel met Servoy?

**Producten**Aan het eind van de stageperiode zal er worden opgeleverd:

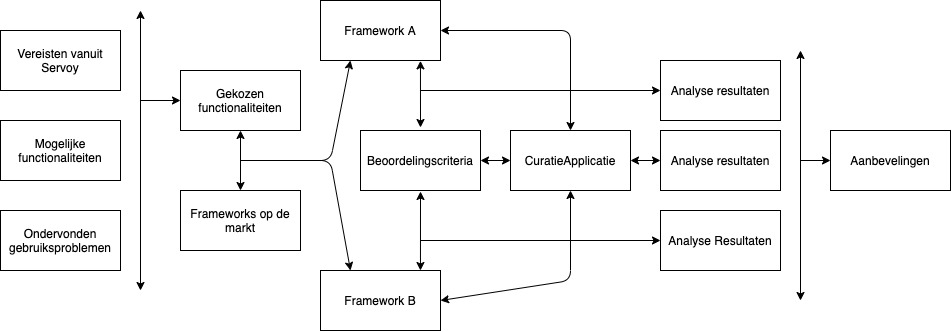
- Het plan van aanpak, dit bespreekt de methoden en technieken zien waarmee het project zal worden uitgevoerd en ook het type onderzoek.  
- De scriptie, het verslag van de afstudeerstage.  
- Het portfolio, hierin worden de competenties waaraan gewerkt is weergegeven en in welke specifieke vorm.  
- Het onderzoeksrapport, dat antwoord geeft op de hoofd- en deelvragen.   
- Een curatie applicatie, die bedoeld is voor de medewerkers van Servoy, om de data van het forum te filteren op relevante posts voor de chatbot.   
- Een chatbot die kan antwoorden op vragen die voorkomen in de documentatie en in het forum.  
- De eindpresentatie waarin de opdracht en de resultaten ervan worden laten zien.

**Impact op de organisatie**

Als de curatieapp en chatbot geïmplementeerd worden in de Servoy IDE, zal het de leerervaring van de gebruikers vergemakkelijken en dus zowel tijd voor de gebruikers als voor de medewerkers van het bedrijf besparen. Zo zullen er geen dubbele vragen meer gesteld worden op het forum wat de werknemers tijd bespaart. Ook komen gebruikers er meteen achter of er antwoord is op hun vraag of niet zonder eindeloos de forums of de documentatie te hoeven doorzoeken. Ook zal een chatbot de IDE er professioneler doen overkomen.

# **3. Onderzoeksmethode beschrijving en – verantwoording**

*In dit hoofdstuk worden de methoden en technieken beschreven die gebruikt zijn tijdens het maken en uitvoeren van het onderzoek. Dit wordt gedaan om de benaderingswijze in het onderzoek te verduidelijken. Eerst wordt er gesproken over de algemene technieken en daarna wordt er ingegaan op de specifiekere methoden.*  
   
**Onderzoeksmethode**  
Voor dit project is gekozen voor een literatuuronderzoek omdat het goed uitgevoerd worden binnen dit project doordat het zich richt op het ontwerpen van een prototype. Tijdens het literatuuronderzoek zal er eerst kwalitatief onderzoek plaatsvinden vooral d.m.v. secundaire gegevens om de beste functionaliteiten en frameworks te vinden. Hiervoor worden requirements opgesteld. Eerst zullen de bestaande chatbotfunctionaliteiten en hun eigenschappen in kaart worden gebracht.   
  
Vervolgens zullen deze worden vergeleken met de behoeftes van de gebruikers van het forum en de documentatie om zo de best passende functionaliteiten te vinden.   
  
Daarna wordt er gekeken naar de bestaande frameworks in de markt en wordt er gekeken welke functionaliteiten ze bezitten. Ook wordt hierbij de prijs, de snelheid en gebruik van geavanceerde technieken mee in acht genomen.  
  
Als laatst zal er kwalitatief praktijkonderzoek plaatsvinden om de prestatie van de geïmplementeerde chatbot te toetsen. Er zal algemeen getest worden, maar ook op gebieden van syntax en semantiek. Hieronder volgt het onderzoeksschema



**Technieken**  
Hieronder volgen de technieken om de opdracht uit te voeren.  
   
De volgende technieken zijn opgelegd door Servoy:  
- Een AI API die ervoor zal zorgen dat de chatbot taal kan herkennen en antwoord weet te geven op de vraag van de gebruiker.  
- Javascript i.c.m. Servoy om een deel van de opdracht mee te ontwikkelen.  
- SQL, want er zal worden gewerkt met database gegevens. Servoy zal de tooling verrichten wat betreft het ophalen en werken met sql gegevens.  
  
De volgende zijn vrij gekozen:  
- Servoy. De curatie applicatie wordt ontwikkeld met Servoy. Om bekend te worden met de bedrijfstaal die de collega’s kennen en waarin zij eventueel ook support kunnen bieden. Bovendien is Servoy low code database transacties bijvoorbeeld worden onder water verricht.  
  
Eventueel als er nog genoeg tijd is:  
- Java, om de backend van de chatbot te verbinden met de Servoy IDE als eclipse plugin aangezien de Servoy IDE op eclipse is gebaseerd.   
- Angularjs of Servoy om de front end van de chatbot te maken.  
  
**Minimum Viable Product**Het minimum viable product (MVP) is als volgt: Er moet minimaal een chatbot zijn die in staat is om mensen taal te begrijpen en antwoord kan geven op een vraag vanuit een text corpus. Met corpus wordt er hier bedoeld, een verzameling geschreven teksten, bijvoorbeeld een stuk geschreven over een bepaald onderwerp.

# **4. Plan van aanpak**

*In dit hoofdstuk wordt het plan van aanpak voor het onderzoek beschreven. Het vormt de basis voor het uitvoeren van het onderzoek en garandeert de gestructureerde aanpak van het onderzoek. Hierin worden onder andere het pr doelstelling, probleemstelling, deelvragen, het theoretisch kader, de kosten en de planning besproken.*  
**4-1 Doelstelling en probleemstelling**De probleemstelling is de volgende:  
De gebruikers zijn veel tijd kwijt aan het vinden van relevante antwoorden op hun vragen wat betreft het Servoyplatform. Hierdoor verliezen de medewerkers van Servoy ook tijd aan het beantwoorden van deze vragen.   
De doelstelling die hieruit leidt is als volgt: Er wordt gekeken of er een chatbot valt te maken die dit probleem te verkleinen.

**4-2 Onderwerp van onderzoek**Het onderwerp van het onderzoek is het vinden van de meest geschikte functionaliteiten van een chatbot voor het systeem van Servoy.

**4-3 Prototype**   
Het uiteindelijke product voor dit onderzoek is een chatbotprototype dat antwoord kan geven op de vragen van de gebruikers die op het forum al gesteld zijn en ook vragen wat betreft de documentatie weet te beantwoorden. Hiervoor zal onderzoek moeten worden gedaan naar het juiste type database, zoekmachine en chatbot API waarbij ook andere factoren moeten worden overwogen zoals prijs en onderhoudbaarheid voor de toekomst. Verder is de aanname dat de data vooraf bewerkt worden waardoor de chatbot er beter op kan inspelen, of dit zo is zal worden tijdens het onderzoek verder worden uitgediept. Dit zal worden gedaan om de medewerkers van Servoy, die zich bezighouden met het forum, de mogelijkheid te bieden om de data te cureren op relevantie. In dit geval betekent cureren het beoordelen van de posts per topic op waarde van relevantie.   
  
**4-4. Theoretisch kader**  
Het theoretisch kader betreft een groot aantal bronnen. Om de onderzoeksobjecten te kunnen onderzoeken moet er worden onderzocht hoe de data kan worden geïnterpreteerd door geavanceerde taalfunctionaliteiten. De functionaliteiten die onderzocht worden zijn op het gebied van herkenning van semantiek, synoniemen en de analyse van relevantie van onderdelen van de vraag, en de vraag zelf. Voor de analyse van de chatbotframeworks wordt een variatie van bronnen gebruikt uit alle velden. Onder andere wetenschappelijke artikelen, papers en verslagen van conferenties. Verder moet er kennis worden opgedaan over de chatbot-API door de documentatie door te nemen. Ook zal er JavaScript i.c.m. de Servoy uitbreidingen moeten worden geleerd.   
 **4-5. Onderzoeksinstrumenten**Voor het onderzoek zijn de volgende onderzoeksinstrumenten gebruikt:  
*Desk Research*Literatuuronderzoek zal worden gedaan naar:  
- De bestaande functionaliteiten van chatbots. Op het gebied van taalherkenning en het vinden van een juist antwoord uit een specifieke context.  
- Het aanbod en de eigenschappen van de API’s in de markt.  
*Field Research*Wat betreft Field Research zullen er een aantal dingen gedaan worden:   
- Een interview met een AI-bedrijf om advies van experts te krijgen.   
- Veel dialoog met collega’s om een idee te krijgen over de Servoy IDE en de specifieke taal die ze gebruiken.   
- Mail- en/of telefoon contact met experts.  
- Metingen van de effectiviteit van de gekozen functionaliteiten d.m.v. tests.  
- Ook worden de gebruikerservaringen met de nieuwe chatbot gedocumenteerd.

**4-6. Organisatie**

Betrokken bij het onderzoek zijn meerdere programmeurs van Servoy die te maken hebben met de klanten en vragen die worden gesteld op het forum. De afdeling waar ik werkzaam ben heeft rond de 25 medewerkers. De CEO is de leidinggevende en tegelijk de bedrijfsbegeleider. Er zal wekelijks naar hem worden teruggekoppeld over de status van de voortgang. De Servoy consultants staan beschikbaar om training te worden gegeven om bekendheid op te doen met de Servoy taal. Verder zal Servoy’s R&D departement in Roemenië ook benaderd worden aangezien databases die ik benader, beheert worden door een medewerker in Roemenië. Eventuele gevonden bugs worden ook doorgegeven aan Roemenië. De Stagebegeleider zal twee keer langskomen om de voortgang en de stand van zaken te bespreken.  
Er zal toegang worden verschaft tot de databases van het forum en van de wiki. Ook wordt er met de Servoy IDE gewerkt. De communicatie met de bedrijfsmedewerkers zal gaan via Slack of Zoom.   
  
Verder wordt er geprogrammeerd met een eigen laptop, namelijk een Macbook pro 2015 met 16GB Ram en soms ook op een Windows computer met 8GB RAM en een betere videokaart en processor.

**4-7. Planning**

De volgorde waarin ik heb gepland is vijfledig:  
1. Aan het begin van de stageperiode zo veel mogelijk beginnen met de interviews en gesprekken om het volledige probleem in kaart brengen..   
2. De exacte behoeftes van de gebruikers te inventariseren door te praten met de medewerkers die in contact staan met de gebruikers.  
3. Het zoeken van de bestaande chatbot functionaliteiten en de combinaties van functionaliteiten die worden aangeboden door de frameworks in de markt.  
4. Vervolgens het opstellen van advies over de implementatiekeuze(s).  
5. Als laatst de implementatie(s) zelf en het testen ervan.  **4-8. Agile met iteraties**Voor het project wordt er agile gewerkt met iteraties in de vorm van timeboxing.  
Ik heb een fasering gemaakt waar ik de uit te voeren taken heb geplaatst. Zodra de einddatum van een timebox bereikt wordt, beland het project in de volgende fase.

Tijdens de implementatie zal er elke weekmondeling worden teruggekoppeld aan de bedrijfsbegeleider. Bij een afwijking van de planning door snelle vooruitgang of ondervonden obstakels wordt de stand van zaken geëvalueerd en worden er eventuele wijzigingen  
aangebracht. Hieronder wordt de planning voor het project weergegeven  
*Notitie: Tijdens de corona-pandemie werd er vanaf maart elke dag gewerkt met een daily stand-up via zoom om 9.30 ‘s ochtends.*

| **Fase 1**:  13-11 t/m 30-11 | **Fase 2:** 01-12 t/m 15-01 | **Fase 3**:  16-01 t/m 16-04 | **Fase 4**: 17-04 t/m 12-06 |
| --- | --- | --- | --- |
| Projectcontext beschrijven | Uitzoeken van chatbottermen | Onderzoek naar bestaande frameworks afmaken | Het noteren van de resultaten en het testen. |
| Hoofd- en deelvragen opzetten | Opstellen van requirements en use cases | Uitkiezen van framework(s) | Evaluatie |
| Onderzoeksmethode en -model definiëren | Onderzoek naar bestaande chatbot functionaliteiten | Implementatie van framework(s) beginnen | Advies opmaken uit de resultaten, tests en evaluatie en het doen van aanbevelingen |
|  | Keuze van geschikte chatbot functionaliteiten |  | Implementatie van framework(s) afmaken |
|  | Onderzoek naar bestaande frameworks beginnen |  |  |

**4-9. Kosten**

De kosten worden bepaald door de keuze van een of meerdere chatbot frameworks. Ook kunnen extra resources zoals een virtuele server, of extra frameworks die gekoppeld moeten worden aan het chatbot framework, geld kosten. Ook zou specifiek advies van het bedrijf van de geselecteerde functionaliteit kunnen worden aangevraagd wat kosten met zich mee zou kunnen brengen. Hieronder wordt een kostentabel weergeven:

| Prijs van eventuele virtuele server | stelpost maandelijks €400 euro |
| --- | --- |
| Professioneel advies | stelpost €200 |
| Framework API | stelpost maandelijks €250 |
| extra resources/frameworks | stelpost maandelijks €100 |

**4-10. Haalbaarheid en risico’s**

Een risico is het uit de tijd lopen bij het zoeken naar de complexe aard van de verschillende AI-technieken, omdat er erg veel tijd in kan gaan zitten. Om dit te voorkomen is de planning hierboven gemaakt d.m.v. Timeboxing.  
  
Ook is er risico dat het framework te duur wordt wat betreft maandelijkse prijs. Zowel om hardware te huren om de API te kunnen runnen, als de API zelf. Daarom zullen de bedrijven die frameworks bieden die een potentiële keuze zijn, telefonisch en via e-mail worden benaderd met de vraag om extra te verschaffen wat betreft de kosten.  
  
Het kan zijn dat laptop waarmee gewerkt wordt niet genoeg computerkracht bevat om met grote data en met de gekozen technieken te werken. Als oplossing kan voor dit probleem een virtuele server worden gebruikt.  
  
Een ander risico zou zijn om de functionaliteiten niet duidelijk genoeg af te bakenen waardoor verwarring zou kunnen ontstaan in de te behalen doelen en gewenste producten. Daarom wordt hierop in fase 2 van de planning de grootste nadruk gelegd, zodat er genoeg tijd is om grondig onderzoek te kunnen doen.  
  
Het grootste risico is dat het management het niet eens is met mijn beslissingen en er niet achter staat. Dit zal worden voorkomen door tijdens de wekelijkse terugkoppeling steeds het gedachteproces bij de huidige taak aan te kaarten. Ook worden de vervolgstappen steeds toegelicht zodat de bedrijfsbegeleider tijdig een andere koers zou kunnen aanbevelen in het geval van meningsverschil.

# **5. Evolutie van de Chatbot**

*In dit hoofdstuk zal er gesproken worden over de theorie van chatbots om de lezer wijs te maken over de chatbot in het algemeen, een stukje geschiedenis en ook wat inzicht in de huidige stand van zaken.*

Vandaag de dag zijn de meeste mensen wel bekend met chatbots en is het een wijdverspreid fenomeen. Er wordt tegenwoordig veel gesproken over bots en er zijn veel misvattingen over. Het idee van bots bestaat al sinds de dertiger jaren, toen Alan Turing met het idee kwam over robots die communiceren als mensen. Hij ontwikkelde in 1936 de Turing test waarin er wordt gekeken of een machine zich kan voordoen als een mens. Over deze test wordt altijd nog gesproken wanneer men het heeft over kunstmatige intelligentie.  
(Akerkar, 2018)  
 **De chatbot**Een chatbot is een service mogelijk gemaakt door regels en soms kunstmatige intelligentie waarmee je meestal via een chat interface communiceert.Historisch gezien kunnen chatbots worden verdeeld in twee groepen: op logische regels gebaseerd en gegevens gestuurd. De eerste is afhankelijk van vooraf gedefinieerde opdrachten die de bot uitvoert als er een specifiek geval optreedt zoals bij het voorbeeld: “Hoe laat zijn jullie open?”. Elk van deze opdrachten moet worden geschreven door een chatbot-ontwikkelaar met behulp van reguliere expressies en gegevensanalyse van de gegeven tekst. De tweede soort daarentegen, datagestuurde chatbots, vertrouwen op machine learning-modellen die vooraf zijn vastgelegd op dialoog gegevens.

De eerste chatbot werd gemaakt in 1964 toen Joseph Weizenbaum, die toen werkzaam was bij MIT, een chatbot ontwikkelde die Eliza heette. Die gebruikte simpele regels van conversatie and herformuleerde het meeste van wat de gebruikers zeiden. Terwijl het liet zien dat het naïeve gebruikers kon laten denken dat ze het tegen een echte therapeut hadden, kon het systeem niet daadwerkelijk het probleem van de gebruiker begrijpen (Janarthanam & Venigalla, 2017).  
  
Kunstmatige intelligentie is sinds die tijd een stuk opgeschoten, neem als voorbeeld 1997 toen een computer van IBM de wereld schaakkampioen Kasparov versloeg. Ook chatbots zijn vooruit gegaan. Neem als voorbeeld Siri, de algemeen bekende chatbot assistent van Apple die in 2011 op de markt verscheen en die wekelijks in 2016 miljard verzoeken per week kreeg ([Rossignol](https://www.macrumors.com/author/joe-rossignol/), 2016).   
In de praktijk zijn bots in veel velden actief zoals: persoonlijke assistent, klantenservice, Conversational Commerce (het raakvlak tussen berichten-apps en winkelen.), Teamsamenwerking, nieuws, entertainment, reizen en nog veel meer (“Types of Bots: An Overview”, z.d.). Ook zijn er meer basale vormen zoals Q&A systemen. Voor het bouwen van de verschillende chatbots zijn ook verschillende benaderingen en hulpmiddelen. Elk van deze hiervoor genoemde bots kan een andere combinatie van functionaliteiten hebben die het beste past bij het gewenste product. Een goede combinatie van verschillende AI-vormen kan bijvoorbeeld een combinatie van natuurlijke taalverwerking, machine learning en semantisch begrip zijn. Hier wordt later nog op teruggekomen.  
  
**Chatbots vandaag**Om een idee te geven van wat de omvang van het dagelijks gebruik van chatbots is. Er zijn vandaag meer dan 300.000 Facebook Messenger-chatbots actief. Ook is het aantal zoekopdrachten met het woord chatbot in google sinds het begin van 2016 heel populair geworden (zie Bijlage 1)

De conversatietechnologieën voor gebruikersinterfaces zijn momenteel een van de meest populaire onderwerpen in de technologiebranche (Janarthanam & Venigalla, 2017). Er wordt gezegd dat bots de software-industrie op dezelfde manier zullen verstoren als de web- en mobiele revoluties (SnatchBot Team, 2019).

Een van de meest populaire voorbeelden van chatbots is Mitsuku die wordt gezien als werelds beste conversatie chatbot gemaakt door Steve Worswick. Mitsuku is een vijfvoudige Loebner Prize-winnaar. Dat is een prijs die uitsluitend wordt uitgereikt voor de meest menselijke chatbot.

Meestal kosten bots de eenmalige prijs om ze te laten ontwikkelen maar daar buitenom ook nog de maandelijkse lasten. Zo heeft microsoft bijvoorbeeld een vaste prijs per 1000 berichten naast de 10000 berichten die je gratis elke maand mag versturen (“Azure Bot Service pricing”, z.d.).

**Kritiek op chatbots/deep learning**Er is natuurlijk ook kritiek op chatbots: Ze zijn nog niet perfect en zullen volgens nooit de echte medewerkers van bijvoorbeeld de klantenservice vervangen. Chatbots zouden geen nuance hebben (“Chatbots: what are they good for?”, z.d.) en bots die te veel functies tegelijk willen uitoefenen, falen ook vaak (Khalid, F. 2016). En als je chatbot slechte resultaten geeft laat dat ook natuurlijk een slechte indruk achter van je bedrijf. Daarnaast duurt het volgens sommigen veel te lang voordat de chatbot “volwassen” wordt d.m.v. trainen.

# **6. Huidige omgeving**

*In dit hoofdstuk wordt er besproken wat de huidige stand van zaken in het bedrijf is wat betreft het forum, de documentatie en de Servoy taal.*   
  
  
**Hoe werkt het nu?**  
De servoy ontwikkelomgeving maakt gebruik van eigen plugins en keywords. Dus is het een uitbreiding op de programmeertaal javascript die tijd kost om te leren. Zo zijn er keywords zoals: *databasemanager, forms, foundset, controller* en nog veel meer die uniek zijn voor de Servoy IDE.  
  
Het forum is gebouwd in PHP en maakt gebruik van phpBB, een opensource softwarepakket, geschreven in PHP, bedoeld om een internetforum te ontwerpen. Ook worden er een aantal extra functies gebruikt zoals Google Analytics, Apache en Atom (bijlage 2). Op dit moment is er bij Servoy nog geen chatbot aanwezig.   
  
**De organisatie van het huidige forum**Op dit moment worden de vragen op het forum behandelt door een aantal medewerkers in het kantoor op de zuidas maar ook door het R&D departement in Roemenië.  
  
De gegevens van het forum staan opgeslagen in een PostgreSQL database met een groot aantal kolommen waarvan de meeste niet relevant zijn voor gebruik in de nieuwe chatbot. Er zijn in totaal 112.409 posts in deze database.   
  
**De organisatie van de documentatie**  
De documentatie staat online op de wiki website https://wiki.servoy.com en staat ook opgeslagen in een postgreSQL database. De relevante tabel in de database die de documentatiepagina’s bevat heeft een grootte van rond de 80.000 records die ook HTML bevatten.

# **7. Theoretische Achtergrond**

*In dit hoofdstuk worden de concepten die een chatbot kan gebruiken uitgelegd: Kunstmatige intelligentie, Machine learning, Deep learning en Neural Networks.*  
 **Vormen van Kunstmatige intelligentie**Kunstmatige intelligentie is het bredere concept van machines die taken kunnen uitvoeren op een manier die we als 'slim'/smart zouden beschouwen. Machine Learning en Deep learning zijn allebei termen die vallen onder het begrip kunstmatige intelligentie.  
  
Voor machines betekent AI momenteel het vermogen om beslissingen te nemen op basis van ervaringen uit het verleden. Dit is een concept dat nauw verwant is aan Machine Learning, maar de beslissingskracht van de AI is meestal het startpunt, in plaats van het proces dat met vallen en opstaan ​​is bereikt.  
Arthur Lee Samuel introduceerde de term “machine learning”. Machine learning (ML) kan worden gedefinieerd als : “computationele methoden die de ervaring gebruiken om de prestaties te verbeteren of om nauwkeurige voorspellingen te doen (Akerkar, 2018).   


Het belangrijkste verschil tussen ML en traditioneel geprogrammeerde algoritmen is de mogelijkheid om gegevens te verwerken zonder dat een software engineer uitgebreide instructies hoef te geven over hoe elk type data record moet worden behandeld. De machine zelf definieert deze regels (“Arthur Samuel”, z.d.).

**Machine Learning**  
Machine Learning is een huidige techniek/toepassing van AI gebaseerd op het idee dat we machines toegang kunnen geven tot gegevens en dat ze zelf moeten “leren” (Marr, 2016).   
  
Het is een methode om algoritmen te trainen zodat ze kunnen leren hoe ze beslissingen moeten nemen. Training in ML houdt in dat het algoritme veel gegevens ontvangt en meer informatie over de verwerkte informatie krijgt.

**Voorbeeld ML***“Je moet bijvoorbeeld voorspellen of klanten van je e-commerce winkel een aankoop zullen doen of niet. Om een ​​model te trainen in het doen van dit soort voorspellingen, 'voedt' men een algoritme met een gegevensset waarin verschillende gegevens over klantgedrag en de resultaten worden opgeslagen, bijvoorbeeld of klanten een aankoop hebben verlaten of voltooid. Door van deze historische gegevens te leren, kan een model voorspellingen doen over toekomstige gegevens”*  (Akerkar, 2018).

Er zijn veel verschillende algoritmen voor ML, allemaal met verschillende sterktes en geschikt voor verschillende soorten problemen. Sommige, zoals beslissingsbomen, zijn transparant, zodat een waarnemer het redeneerproces van de machine volledig kan begrijpen. Anderen, zoals natuurlijke netwerken, zijn een zwarte doos, wat betekent dat ze een antwoord produceren, maar het is vaak erg moeilijk om de redenering erachter te reproduceren. (programming collective intelligence - building smart web 2.0 applications, 2007)

**Deep learning**De term Deep Learning(DL) werd geïntroduceerd in kunstmatige neurale netwerken door Igor Aizenberg in 2000. DL is een subset van ML; het is eigenlijk gewoon een techniek om ML te realiseren. Met andere woorden, DL is de volgende evolutie van ML. DL-algoritmen zijn grofweg geïnspireerd op de informatieverwerking patronen in het menselijk brein.

Net zoals we onze hersenen gebruiken om patronen te identificeren en verschillende soorten informatie te classificeren, kunnen diepe leeralgoritmen worden geleerd om dezelfde taken voor machines uit te voeren. Het brein probeert meestal de informatie die het ontvangt te ontcijferen. Dit wordt bereikt door tekstartikelen te labelen en toe te wijzen in verschillende categorieën. Wanneer we nieuwe informatie ontvangen, proberen de hersenen deze te vergelijken met een bekend item voordat de mens er iets van begrijpt - wat hetzelfde concept is als DL-algoritmen.

Het vergelijken van DL versus ML kan helpen hun subtiele verschillen te begrijpen. Hoewel DL bijvoorbeeld automatisch de functies kan ontdekken die voor classificatie moeten worden gebruikt, vereist ML dat deze functies handmatig worden verstrekt. Bovendien heeft DL, in tegenstelling tot ML, geavanceerde machines en aanzienlijk grote hoeveelheden trainingsgegevens nodig om nauwkeurige resultaten te leveren (Garbade, 2018).

**DL technieken:**Hieronder worden twee voorbeelden van DL-technieken genoemd:

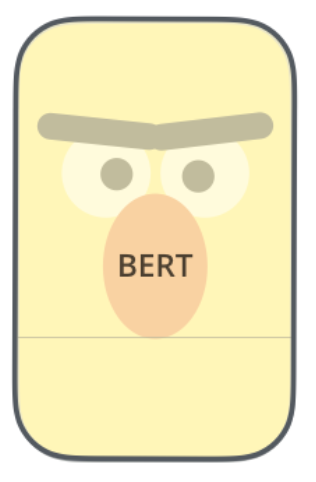
**Artificial neural networks**Neuraal netwerk of kunstmatig neuraal netwerk (ANN) lijkt veel op een brein en zijn een soort algoritmen die proberen de manier te imiteren waarop onze hersenen beslissingen nemen.Net zoals neuronen in de hersenen verantwoordelijk zijn voor al onze taken die vergelijkbaar zijn met die in ANN en ook bestaan ​​uit neuronen, is een 'neuron' in een neuraal netwerk een eenvoudige wiskundige functie die informatie vastlegt en organiseert volgens een architectuur van de geleverde dataset (Khatun, 2018).

**Natural Language Processing (NLP) en Natural Language Understanding (NLU)**NLP is een subgebied van kunstmatige intelligentie houdt zich bezig met de interactie tussen mensen en computers met behulp van taal. NLP biedt een chatbot de mogelijkheid om taal te herkennen die in een zin staat. Het wordt vooral gebruikt om grote hoeveelheden natuurlijke taalgegevens te analyseren, verwerken en gebruiken.  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
**Platforms en Conversational agents**   
Chatbotplatforms stellen gebruikers in staat om chatbots maken zonder enige programmeerervaring. door gebruik van simpele drag-and-drop functies. De platforms integreren vaak ook met grote communicatieplatforms zoals Facebook, Slack, en/of Twitter. (Artificial Solutions, 2020)   
  
Een conversational agent (CA) is een computersysteem dat bedoeld is om conversatie te houden met mensen. De CA’s gebruiken één of meer text, speech, grafische, gebaren, en andere modes voor communicatie in zowel het input als het output kanaal. 

# **8. Uitwerking Hoofd- en Deelvragen**

*In dit hoofdstuk worden de deelvragen per stuk onderzocht om zo antwoord te kunnen geven op de hoofdvraag. De functionaliteiten die bestaan en hun werking worden toegelicht. Ook worden er functionaliteiten gekozen en wordt er beredeneert waarom er voor deze functionaliteiten gekozen is. Verder wordt er gekeken naar de gebruikersproblemen en het aanbod van chatbots in de markt en de keuze daaruit.*

## **8.1 Algemene chatbot functionaliteiten die bestaan**

Veel bronnen op het internet omschrijven functionaliteiten die een bot kan hebben. Hier is vaak een bepaalde overlap, als we deze allemaal toevoegen is dit het complete overzicht.  
Er zijn veel functionaliteiten te vinden die een chatbot kan bevatten (“7 Characteristics of a Great Chatbot”, z.d.) (Chatbot Features to set you Apart, z.d.) zoals:  
  
- Intent classification/recognition  
- TF-IDF  
- Question Answering  
 Open Domain Question Answering(ODQA)  
 Closed Domain Question Answering(CDQA)  
- Ranker  
- Reader  
- Confidence score  
- NLU  
- Sentimentanalyse  
- Named entity Recognition(NER)  
- Word embedding  
- Bert  
- Elmo  
- Voice to Text  
- Training  
- Memory  
- Gated Flows  
- Entities  
- examples  
  
  
.  
  
  
  
 **Intent classification/recognition**Intent verwijst naar het doel dat de klant voor ogen heeft bij het typen van een vraag of opmerking.Intent recognition chatbots worden aangedreven door het oppikken van intenties tijden de conversatie. Een hoofddoel is om probabilistische losse gesprekken om te zetten in een gestructureerde indeling die de Chatbot begrijpt. Tegenwoordig verbeteren deze gespreks mogelijkheden met het effectieve gebruik van kunstmatige intelligentie, ML en andere intelligente technologieën (Anurag, 2020).  
 **Term frequency - inverse document frequency (TF-IDF)***Tf-idf* is een manier om te kijken hoe waardevol een woord is door te kijken naar hoe vaak een woord voorkomt in een document. Tf-idf komt vaak voor in search engines in scoring and ranking. Dit laat zien hoeveel informatie het potentieel kan leveren. Een hoog gewicht in tf-idf wordt bereikt door een hoge term frequentie in het betrokken document en een lage document frequentie van de term in de hele verzameling documenten; de gewichten hebben de neiging om gemeenschappelijke termen uit te filteren ([Algemene beschrijving van tf-idf],   
z.d.). De waarde van **idf en tf – idf** is altijd groter dan of gelijk aan 0 en maximaal 1.   
  


**ODQA**open-domein vraagbeantwoording (ODQA) is een systeem dat antwoord geeft op basis van een documentverzameling. Open-domeinsystemen verwerken vragen over alle onderwerpen die met de verzameling te maken hebben en is een uitdaging in de natuurlijke taalverwerking. Een succesvol ODQA-systeem moet namelijk in staat zijn om een of meer kennisbronnen effectief op te halen en te begrijpen, om zo een ​​juist antwoord te vinden (Wang et al., 2018). Chatbots passen ODQA toe om gebruikersverzoeken te beantwoorden. Er zijn verschillende benaderingen voor de architectuur van een ODQA-systeem. Bedrijfsgerichte Natural Language Processing (NLP) -oplossingen maken vaak gebruik van ODQA om vragen te beantwoorden op basis van interne bedrijfsdocumentatie.  
  
**CDQA**  
Met gesloten domein modellen daarentegen kunnen systemen voor het beantwoorden van gesloten-domeinvragen worden gebouwd bovenop een formeel model met de mogelijkheid om formele logica en redenering toe te passen.  
 **Ranker**De ranker functionaliteit omvat het ophalen van informatie, in het Engels Information Retrieval (IR).In dit geval bedoelen we met IR de wetenschappelijke discipline achter geavanceerde NLP zoekmachines. Rangorde van zoekopdrachten is een van de fundamentele problemen bij IR en is een belangrijk deel van het begrijpend lezen wat moderne chatbots kunnen.   
  
Een voorbeeld: Gegeven een vraag *q* en een verzameling van documenten *D*, is het probleem om de documenten in D te sorteren op overeenkomst met de vraag, volgens een bepaald criterium, zodat de bovenaan de "beste" resultaten staan wat betreft documenten. (Wang et al., 2018).   
  
**Reader**De reader extraheert het antwoord uit de aangeleverde passages. De Reader vergelijkt de vraag met de inhoud van elke pagina, om te kijken hoe volledig ze overeenkomen met de vraag en weergeeft dit in zogenoemde “probability scores”. Probability scores hebben meestal een nummer van 0 tot 1, waarbij 1 de hoogste mate van overeenkomst aanduidt. Rankers gebruiken de probability score overigens ook. De reader kan verschillende complexe taalmodellen gebruiken om dit te berekenen, zoals bijvoorbeeld MATH-LSTM en SGD/backprop (Wang et al., 2018)  
 **Confidence Score**  
Met de confidence score wordt er aangegeven hoe precies de chatbot kan bepalen wat er door de gebruiker gezegd wordt en hoe precies het antwoord is wat hij op de query vindt. Hoe werkt de confidence score  
**NLU**Kan het best gezien worden als een eerste stap richting NLP. Voordat een chatbot een taal kan processen moet hij de taal eerst begrijpen. Het gebruik van NLU kan variëren van simpele taken zoals het uitvoeren van kleine commands gebaseerd op tekstbegrip op een smal niveau, zoals het omleiden van een email naar de juiste persoon. Tot veel ingewikkeldere taken zoals het volle begrijpen van nieuws artikelen of zelfs dubbelzinnige poëzie (Monkeylearn, 2020). **Sentimentanalyse**Is de mogelijkheid om de stemming van de gebruiker te herkennen door aanwijzingen in de zinsstructuur en ook woorden te analyseren. Het is een laag bovenop NLU.  
  
**Named entity Recognition(NER)**  
NER is het herkennen van woorden op linguïstisch gebied of het gebied van semantiek. Zo kan NER bijvoorbeeld locaties, datums en entiteiten herkennen. Afhangende van de databank, kan NER heel onderwerp breed zijn of juist in één onderwerp heel specifieke herkenningen doen. Ook heb je Deep Learning NER wat veel preciezer is omdat het gebruikt maak van een techniek die word embedding wordt genoemd (Innoplexus, 2019)  
  
**Word Embedding**Word embeddings kunnen gebruikt worden om representaties van woorden te maken en leiden tot betere prestaties bij taken met NLP. Word embeddings maken het mogelijk woorden weer te geven door een reeks getallen. De nieuwste versies van word embeddings maken gebruik van neurale netwerken.  
  
FastTextFastText is een open-source, gratis, library, gemaakt door Facebook's AI Research (FAIR) lab, waarmee bots vectorrepresentaties kunnen creëren. Fasttext maakt gebruik van neurale netwerken. De vector voor een woord wordt gemaakt door de som van “karakter n-grams”. N-grams kunnen lettergrepen, letters of woorden zijn. Er zijn ook alternatieve modellen op FastText zoals CBOW, GloVe en Skip-gram (Wu & Yan, 2018)  
word2vector   
Ook word2vector kan een corpus van tekst gebruiken om een vectorruimte te produceren, meestal van enkele honderden dimensies, waarbij aan elk uniek woord in het corpus een overeenkomstige vector in de ruimte wordt toegewezen.  
  
**BERT**Bidirectional Encoder Representations from Transformers (BERT) werd in 2018 gecreëerd door Jacob Devlin and zijn collega’s bij Google. BERT is een vorm van pre-training en voorspelt richting twee kanten van een woord het volgende meest waarschijnlijk voorkomende woord. Hierdoor kan een chatbot een zoekopdracht beter begrijpen door. Het originele Engelstalige BERT-model gebruikte twee corpora (met corpora bedoelen we een verzameling [teksten](https://nl.wikipedia.org/wiki/Tekst)) in pre-training: BookCorpus en de Engelse Wikipedia (Nayak, 2019) (Alammar, 2018)  
  
**ELMo**ELMo is een diep gecontextualiseerde woord representatie die zowel complexe kenmerken van woordgebruik modelleert bijvoorbeeld syntactisch en semantiek, als hoe dit gebruik varieert binnen taalkundige contexten d.w.z zijn verschillende betekenissen van één woord.  
Net als BERT voorspelt ELMo het volgende woord in een sequentie van woorden. Het kan gebruikt worden als een component in andere modellen die met taal werken.  
(ELMo, z.d.).  
  
**Voice to Text**De functie die de spraak van de gebruiker kan omzetten in text  
**Training**  
De mogelijkheid om de chatbot te trainen. Learn from agent/customer live actions  
  
**Memory**  
Een groter geheugen ondersteund langere conversaties met de chatbot zonder dat hij de lijn van het verhaal verliest en daardoor dus coherent blijft.  
  
**Gated Flows**  
Vergen bepaalde antwoorden en acties van de gebruiker om zo de conversatie in vooropgestelde banen te kunnen leiden.  
  
**Entities**  
Chatbot-entiteiten zijn er om meer persoonlijke en nauwkeurige antwoorden te geven bij het zoeken van gebruikers. Entiteiten worden opgemaakt uit velden, gegevens of tekst en kunnen onderwerpen beschrijven als een tijd, plaats, persoon, item, nummer, enz. en worden gebruikt om waarden toe te voegen aan de zoekintentie.  
  
**Examples**  
Examples of voorbeelden kunnen worden gebruikt door chatbots om een antwoord op te baseren, hoe groter het aantal examples des te groter gebied kan de chatbot beantwoorden. **Deelconclusie**  
Kortom, er bestaat een groot scala aan functionaliteiten in de markt waarvan sommige gericht zijn op prestatiemaximalisatie, andere weer op het streven naar het menselijke aspect van de chatbot en ander weer op het maximaliseren van het gebruikscomfort. Sommige zorgen voor een vooraf bepaalde structuur en andere zorgen juist weer voor een vrijere. Veel functionaliteiten werken samen met elkaar en vormen deel van een andere om zo samen één samenhangend geheel te vormen.

## **8.2 Wat zijn de problemen die nu worden ondervonden door de gebruikers bij het zoeken naar informatie?**

De ondervonden problemen in de huidige situatie zijn opgenomen in de vorm van requirements. Ook zijn hierbij de wensen van de eindgebruikers, die zijn doorgegeven aan de medewerkers van Servoy, en ook die van de CEO opgenomen.   
  
Tijdens het zoeken naar de requirements werd er gebrainstormd om zoveel mogelijk velden te bedenken waarin de chatbot verbetering kan bieden en waar dus requirements over kunnen worden opgesteld. Over deze velden wordt vervolgens weer gesproken met de medewerkers van Servoy en het management om eventueel input toe te voegen.  
  
Zoals al eerder genoemd kost het de gebruikers veel tijd om het juiste antwoord op hun vraag te vinden, de documentatie is te groot en de zoekmachine is beperkt.  
  
Om nog tot extra professioneel inzicht te komen werd er analyse en advies gevraagd aan ervaren AI professionals met meer dan tien jaar expertise op het gebied van chatbots. Er werd een interview opgezet met de oprichters van AI-bedrijf Cognitive Affairs (voor een samenvatting zie bijlage 3): Reiner Bruns en Roland Hallebeek. Zij lieten de zwakke plekken van de huidige situatie zien, de zoekmachine was te langzaam en te onderontwikkeld. Ook was de data te ongeprepareerd voor een chatbot. Hun advies legde de nadruk op het verbeteren van de zoekmachine en een tf-idf implementatie. Dit advies werd meegenomen in het opstellen van de requirements en dus ook de uiteindelijke keuze van geschikte functionaliteiten. Hieronder staan de opgestelde requirements: **Functionele Requirements**

| **Requirement** | **Beschrijving** |
| --- | --- |
| **FReq 1.** | **De chatbot moet antwoord geven wanneer de conversatie gestart wordt.** |
| **FReq 2.** | **Chatbot kan antwoorden ophalen uit de FAQ en documentatie.** |
| **FReq 3.** | **Er moet een applicatie komen die de huidige forum data cureert op relevantie.** |
| **FReq 4.** | **De chatbot moet synoniemen kunnen begrijpen.** |
| **FReq 5.** | **De chatbot moet foute antwoorden kunnen verbeteren.** |
| **FReq 6.** | **De chatbot moet vragen met dezelfde betekenis, maar semantisch anders gesteld kunnen herkennen als identiek en ze dus ook identiek antwoord geven.** |
| **FReq 7.** | **De chatbot moet gebruikersvriendelijk zijn zodat de gebruiker zich op zijn gemak voelt. de bot moet in staat zijn om vriendelijke conversatie te voeren en de gebruiker er soepel door te leiden.** |
| **FReq 8.** | **De chatbot moet de optie geven om het antwoord te beoordelen en bij negatieve feedback de vraag doorsturen naar een medewerker.** |

**Non-Functionele requirements**

| **Requirement** | **Beschrijving** |
| --- | --- |
| **NfReq 1.** | **Chatbot moet maximum na 10 seconden een antwoord geven.** |
| **NfReq 2.** | **De chatbot moet 24/7 beschikbaar zijn.** |
| **NfReq 3.** | **De chatbot moet consistent zijn met antwoorden. Het gegeven antwoord moet consistent zijn met het onderwerp.** |
| **NfReq 4.** | **De chatbot moet updatebaar zijn met nieuwe toegevoegde FAQ’s en documentatie. Dit zorgt ervoor dat de chatbot met de tijd mee kan gaan.** |
| **NfReq 5.** | **De chatbot moet te runnen zijn op ieder apparaat wat gebruik kan maken van de Servoy IDE.** |
| **NfReq 6.** | **De bot is makkelijk te moduleren om te vormen om te passen op een ander datasysteem en/of platform.** |
| **NfReq 7.** | **Het system moet de 80.000 records van de Wiki aankunnen en de 120.000 posts van het forum** |

**Deelconclusie**De problemen die worden ondervonden zijn de onvindbaarheid van antwoorden en ook de tijd die het kost om ze te vinden. Ook is de zoekmachine te traag en vereist vervanging. Verder is de dataset te ongeprepareerd.

## **8.3 Keuze Van Functionaliteiten**

Hieronder wordt de keuze van de functionaliteiten voor de chatbot weergeven met daarbij een bijbehorende argumentatie.  
  
**NLP**NLP is essentieel bij de reader omdat de opdracht geavanceerde herkenning nodig heeft die de vraag niet letterlijk opzoekt in de database (in het geval van de wiki), maar ook een synoniem of dezelfde vraag anders gesteld moet herkennen.  
  
**ODQA**Het is nodig om uit een kennisbank het antwoord te vinden, eveneens als het lezen kan het ophalen van de informatie ook niet letterlijk gedaan worden (in het geval van de wiki). Zonder deze geavanceerde functionaliteit is het niet mogelijk om te weten wat het antwoord op een gestelde vraag is aangezien er geen koppelingen zullen zijn tussen vraag en antwoord, zoals bij de FAQ van het forum.  
  
**TF-IDF en Zoekmachine**   
Uit het interview (bijlage 3) wordt duidelijk dat een nieuwe zoekmachine nodig is. Hierbij wordt TF-IDF aangeraden en om één van de bekendste zoekmachines te kiezen die dit gebruiken, namelijk apache solr en elasticsearch. Dit advies is ter harte genomen omdat snel zoeken requirement NfReq 1. kan realiseren.  
  
  
  
**Ranker**De ranker is belangrijk omdat er een groot aantal documenten wordt gebruik bij de wiki. Daarom moet er eerst worden gekeken welke artikelen relevant zijn, zodat er duidelijk wordt welke artikelen zullen worden gelezen  
  
**Reader**Een reader is nodig om de documenten te lezen om zo het daadwerkelijk antwoord op de gebruikersvraag te vinden.   
  
**ELMO**  
can use it as a component in other models that need to handle language.  
ELMo is een belangrijke stap voor het pre-trainen van de artikelen in NLP. Dit is handig omdat we gebruik maken van grote datasets waar dit model van kan leren zonder NER te hoeven gebruiken (Alammar, 2018).  
  
**Bert**BERT is een model dat verschillende records heeft gebroken voor hoe goed modellen taalgebaseerde taken aankunnen (Alammar, 2018) en is daarom ook een goede keuze om te zorgen dat de chatbot een snelle prestatie zal hebben.  
  
**Word embedding**  
Word embedding zorgt voor een snellere prestatie van de chatbot in het algemeen en is dus een welkome functionaliteit.  
  
 **FastText**  
 Wu en Yan (2018) schrijven dat van de genoemde vectormodellen in 8.1, Skip-gram   
 en FastText op het gebied van Efficacy on downstream NLP tasks ongeveer even   
 goed zijn en beter dan de andere twee. Ook zeggen ze dat het trainen van FastText   
 efficiënter is dan de andere drie en dat het beter is met FastText Out-of-Vocabulary   
 (OOV) wat leidt tot affiniteit met talen die rijk aan morfologie zijn, zoals Duits. En   
 aangezien Nederlands op Duits lijkt is dit een voordeel voor de Nederlandse klanten   
 in het geval dat de chatbot later ook in het Nederlands gaat werken. Hierom wordt   
 FastText gekozen als geprefereerde embedding.

**Trainen**  
Vooral bij het forum is het trainen belangrijk, omdat de posts in het forum niet even zorgvuldig hoeven worden ingevuld als Servoy’s officiële documentatie en daardoor dus ook vatbaarder zijn voor fouten. Maar ook bij de wiki zal het mogelijk moeten zijn om de chatbot te trainen, omdat het vooralsnog onmogelijk is voor een chatbot om 100% goed antwoord te geven. Het regelmatige trainen zal er voor zorgen dat de chatbot met de tijd beter wordt en een volwassen stadium bereikt.  
  
**Confidence score**  
De confidence score is zeer welkom omdat het inzicht geeft in de resultaten van de chatbot API. Deze cijfers kunnen worden opgeslagen en kunnen bij adequate grootte geanalyseerd worden om eventuele punten van verbetering te vinden waarna de configuratie kan worden aangepast om de prestatie van de chatbot te optimaliseren.  
  
**Deelconclusie**De gekozen functionaliteiten zijn nlp, odqa, ranker, reader, word embedding, fasttext, elmo/bert, tf-idf, trainen en confidence score. Naar verwachting zullen deze efficiënt met elkaar kunnen aansluiten om een solide pakket te vormen en zo de ondervonden problemen op te lossen

## **8.4 Frameworks en AI-engines beschikbaar in de markt**

Op de markt zijn er meerdere bot frameworks te vinden. Bedrijven adverteren veel op het niveau van chatbots en veel websites worden gesponsord of staan in contact met bedrijven die chatbots aanbieden. Daarom is het slim om een zo groot mogelijke lijst aan bronnen op te nemen zodat er zo’n onpartijdig mogelijk beeld wordt geschetst. Uit deze grote lijst wordt het best passende framework gekozen (Mamgain, 2019)(Shah & Shah, 2019)(Patil, Marimuthu & Niranchana, 2017)(Davydova, 2017)(Krishnan, 2020)(Rehan, 2019). Hierbij wordt ook gekeken naar welke bots voor newbies zijn en welke voor ervaren programmeurs **(**Chang, 2017). En ook naar de chatbots van de grootste bedrijven (Lahoti, 2017).

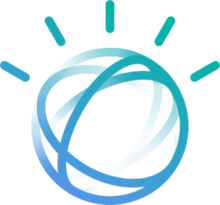
Er is een grote lijst aan chatbots opgesteld:  
BotUI, Cognitive affairs, Microsoft bot builder framework , IBM’s Watson Assistant, Google’s Dialog Flow, Rasa Stack, ANA.CHAT, Meya, Amazon Lex, Pandora Bots, Botpress, BotKit, Bottr, CeJS, Wit.ai, Chatterbot, Semantic Machines, Botman, [SAP Conversational AI](https://cai.tools.sap/), [GupShup](https://www.gupshup.io/developer/home), [Botsify](https://botsify.com/), [Flow XO](https://flowxo.com/), DeepPavlov.  
  
Deze frameworks supporten ook allemaal bepaalde talen om mee te programmeren, van Node.js bij Wit.ai, tot PHP bij Botman, tot meerdere gesupporte talen zoals bij Watson.  
  
Deze gebruiken allemaal verschillende talen, functionaliteiten en mogelijkheden. Daarom is het belangrijk om de eigenschappen van deze frameworks met elkaar te vergelijken om de beste keuze te maken.   
  
**Prijs**De prijzen van de frameworks variëren van gratis tot een éénmalig bedrag, een vast maandelijks bedrag of een maandelijks bedrag dat afhangt van het gebruik. De laatste komt het meeste voor. **Analytics**De duurdere frameworks bieden ook analytics aan, hiermee kun je gebruiksstatistieken analyseren. Ook de chatbot zelf valt te analyseren. Analytics kan bijdrage aan de winst van het bedrijf door de behoeftes van de markt en de gebruikers beter te herkennen. **Snelheid**De snelheid van het framework is vaak gecorreleerd aan de prijs die je betaald voor het gebruik van het framework. Het is hier belangrijk om rekening te houden met de hardware waar de software op gerund wordt. Zo kan het bedrijf wat het framework aanbiedt zelf de berekeningen uitvoeren met een supercomputer, dit staat garant voor een hoge snelheid. Het bedrijf wat het framework gebruikt kan de software ook op haar server runnen.  
  
**Gebruik van Geavanceerde technieken**Duurdere frameworks maken meestal gebruik van geavanceerdere technieken zoals ML en DL.

**Deelconclusie**Er zijn er groot aantal frameworks beschikbaar op de markt die verschillen van prijs, snelheid en andere features. Naar de volgende lijst van opgesteld:  
BotUI, Cognitive affairs, Microsoft bot builder framework , IBM’s Watson Assistant, Google’s Dialog Flow, Rasa Stack, ANA.CHAT, Meya, Amazon Lex, Pandora Bots, Botpress, BotKit, Bottr, CeJS, Wit.ai, Chatterbot, Semantic Machines, Botman, [SAP Conversational AI](https://cai.tools.sap/), [GupShup](https://www.gupshup.io/developer/home), [Botsify](https://botsify.com/), [Flow XO](https://flowxo.com/), DeepPavlov.

## 

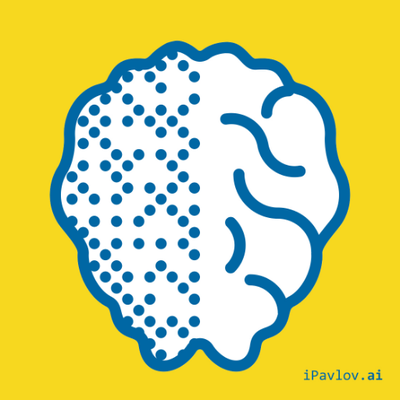
## **8.5 Welk framework is het meest compatibel met Servoy?**

De bronnen genoemd in het vorige kopje geven allemaal argumenten voor en tegen keuzes van bepaalde frameworks. De gekozen functionaliteiten komen het meeste voor in de volgende frameworks:  
  
Aangezien er gecodeerd zal worden en er dus geen sprake zal zijn van drag-and-drop frameworks of die waarvan de focus daar op ligt, vallen er al een aantal af zoals : Chatfuel, ManyChat, KISS, KITT AI, It’s Alive, Motion.AI, MobileMonkey**.**De CEO heeft aangegeven dat hij graag met grote namen in zee gaat: dus Google, IBM of Microsoft. Ook gaf hij aan graag lage kosten te behouden en sprak ook voorkeur uit voor een open-source framework zoals Tensorflow.  
  
Hieruit ontstond het idee om één betaald framework van de grote bedrijven te kiezen en één gratis framework te bekijken wat betreft implementatie. Op deze manier wordt er qua prijs een variëteit van de markt getest wat mede zal laten zien hoe belangrijk de prijsfactor is bij het maken van een kwaliteitschatbot.  
   
Dialogflow heeft geen ML gebasseerde entity recognition, terwijl Watson dat wel heeft. Dit maake Dialogflow langzamer (Azran, S. 2019).

  
**Keuze 1**  
De eerste keuze viel op watson om meerdere redenen:  
  
-Watson biedt meerdere talen om zich te laten benaderen namelijk:  
CURL, .NET, Go, Java, Node, Python, Ruby, Swift en Unity  
  
- Door het interview. Er werd sterk aangeraden om met Watson aan de slag te gaan omdat het een goede basis biedt voor iemand die nog geen ervaringen heeft met chatbots en dus goed te doen moet zijn.  
  
- Het past bij Servoy, zo bleek er na een gesprek met de sales representative van Servoy dat IBM al een partnerbedrijf van Servoy is.  
  
- Het is een zeer populaire platform, zo is “IBM Watson de eerste keuze als platform voor botbouw voor 61% van de bedrijven. (Mindbowser, 2017)  
  
- Het is een geavanceerd platform. Het is namelijk gebouwd op een neuraal netwerk van een miljard Wikipedia woorden en heeft 3 hoofdcomponenten: Intents, Entities en Dialog die het makkelijker maken om de chatbot in te richten(Davydova, 2017)  
  
IBM Watson biedt een reeks services waartoe de gebruiker toegang heeft via API-calls. Hierdoor is het eenvoudig om een van de geboden diensten in een app implementeren.   
Er valt te programmeren watson assistant dialog view, maar er is geen programmeerervaring nodig om simpele dialogen en minimale programmering in complexere dialogen te implementeren. Dus is er niet zoveel kennis over Watson nodig om te beginnen met de implementatie. Een laagdrempelig instapmodel, kan een grotere kans van succes bieden. Watson heeft ook een reeks starterkits waarmee een chatbot binnen enkele minuten kunnen worden geïmplementeerd ([Informatie over Watson], z.d.)(Watson Assistant, z.d.).  
  
Watson heeft NLC, wat een hogere nauwkeurigheid qua antwoord kan geven op minder trainingsgegevens dankzij machine learning. NLC-modellen maken gebruik van Vector Machines en Neural Networks en DL. Ook support het meerdere talen t.o.v. de gratis modellen. De prijzen van Watson’s NLC staan hieronder en lijken niet al te hoog te zijn.

**1 Natural Language Classifier (NLC) free per month.**

* 1000 API calls free per month
* 4 Training Events free per month
* $20.00 USD/Classifier per month
* $0.0035 USD/ API call
* $3.00 USD/ Training Event
* Train classifiers with text phrases, each with a maximum of 1024 characters
* Each .csv file used in training can contain up to 20,000 records.

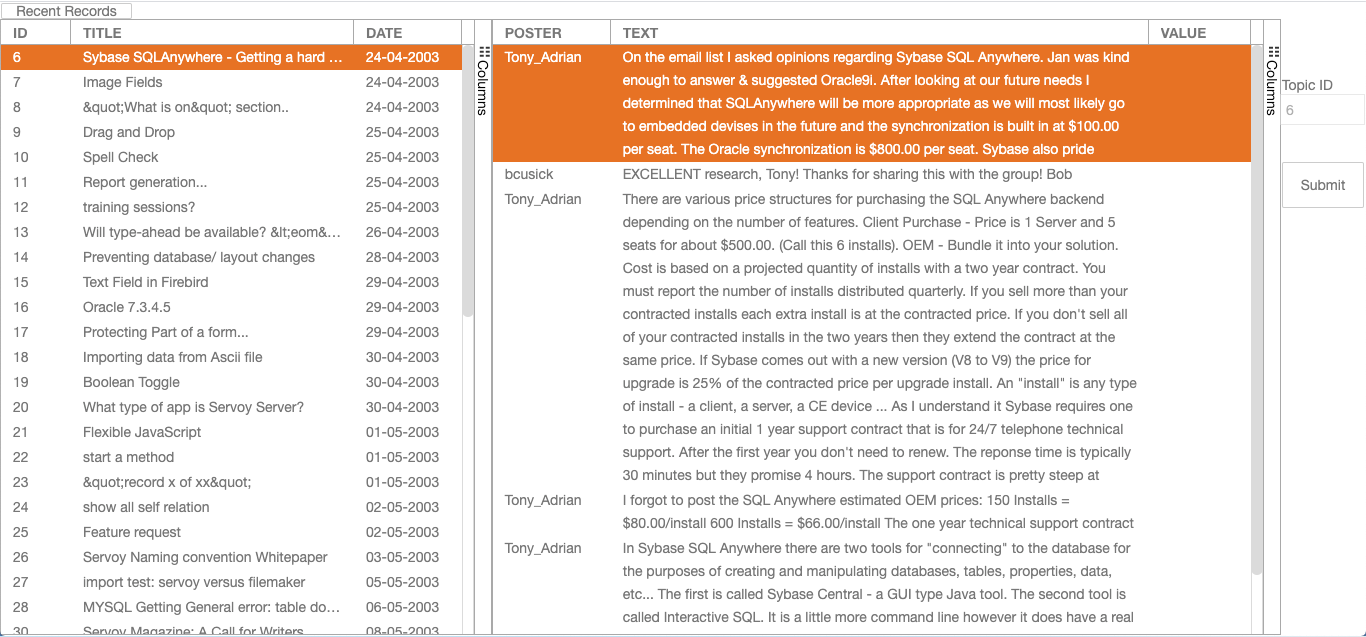
**Zoekmachine keuze**SOLR, ElasticSEARCH en Watson discovery zijn drie mogelijkheden bij het implementeren van een snellere zoekmachine. Een reden om voor deze moderne zoekmachines te kiezen is dat ze geen traditionele database structuur hebben maar de records opslaan in documentvorm.   
  
Tijdens een telefoongesprek met een succes consultant van Watson duidelijk dat Watson Discovery meer gericht is op grote bedrijven in het algemeen (~1000 medewerkers) en dus ook een hogere prijs betreft.   
  
Uit gesprekken met de medewerkers van Servoy bleek dat Elasticsearch meer voor middelgrote bedrijven is bedoeld. En er zijn op github projecten te vinden waar Watson Assistant wordt gecombineerd met Elasticsearch. Dus dit bewijst dat het compatibel is.  
  
Ook maakt Servoy al gebruik van Elasticsearch, dus is het makkelijk voor de medewerkers van Servoy om het parallel te implementeren aan de andere instanties van Elasticsearch die ze gebruiken. Dit draagt bij aan efficiënt management en bekendheid met het systeem.  
  
Daarom is er besloten om ElasticSearch te kiezen. In Elasticsearch is een index vergelijkbaar aan tabellen in een Relation Database Management System (RDBMS). Elke tabel is een collectie van rijen net zoals elke index een collectie van documenten is in Elasticsearch.   
  
 **  
Keuze 2**De 2e keuze viel op DeepPavlov om de volgende redenen:  
  
**MIPT**  
Het is een project van the Moscow Institute of Physics and Technology (MIPT) en dat is één van de vooraanstaande Russische universiteiten op het gebied van natuurkunde en technologie, wat dus een mate van expertise en kwaliteit met zich meebrengt. DeepPavlov wordt gebouwd en onderhouden door het Neural Networks en Deep Learning Lab van het Moscow Institute of Physics and Technology (MIPT). DeepPavlov is een winnende inzending voor de PoweredByTF 2.0 Challenge hackathon.  
  
**Open source**  
Het is open source en gratis, wat het financieel aantrekkelijk maakt als gekozen chatbot platform. Het maakt gebruik van de volgende modellen: **ODQA DEEPPAVLOV**Het ODQA-systeem van DeepPavlov (bijlage 4) heeft twee voor getrainde Wikipedia gebaseerde modellen. De eerste is gebaseerd op de Engelse Wikipedia-dump van 5.180.368 artikelen en de tweede, de Russische van (1.463.888 artikelen). De modellen behandelen Wikipedia als een verzameling artikelen en vertrouwen niet op de interne grafiek structuur. Het heeft een grote relatieve zekerheid (bijlage 5).   
  
Hetgene wat Pavlov heeft en wat het zo aantrekkelijk maakt is de ODQA. ODQA beantwoord welke vragen dan ook, gebaseerd op de documentencollectie die een groot bereik aan onderwerpen bevat. De ODQA taak combineert dus de twee uitdagingen van document retrieval, om de relevante artikelen/Q&A’s te vinden. En het begrijpen van tekst, het identificeren van de antwoord spanwijdte van het aantal gevonden artikelen/Q&A’s. (Konovalov, 2019)  
  
Een modulair ODQA-systeem bestaat uit twee componenten, de eerste (de ranker) moet de relevante artikelen in een database kunnen vinden (in dit geval Servoy’s documentatie), terwijl de tweede (de reader) een antwoord haalt uit een artikel of een kleine verzameling artikelen opgehaald door de ranker (Konovalov, 2019)  
  
**Gebruik van tf-idf**Deeppavlov maakt gebruik van een tf-idf ranker met vectorisatie.**Stanford Question Answering Dataset (SQUAD)**Vraag beantwoorden via SQuAD-dataset is een taak om een ​​antwoord op een vraag te vinden in een bepaalde context bijvoorbeeld een alinea van Wikipedia, waarbij het antwoord op elke vraag een segment van de context is.  
 ***ODQA Ranker***De standaard ranker implementatie neemt een reeks vragen als invoer en retourneert 25 documenttitels gesorteerd op relevantie. Maakt gebruik van een npz file waar de tf-idf matrix is opgeslagen.**Geavanceerde technologie**  
DeepPavlov wordt geleverd met een aantal vooraf gedefinieerde componenten die worden aangedreven door **TensorFlow** en **Keras** voor het oplossen van NLP-gerelateerde problemen, waaronder tekstclassificatie. Tensorflow was één van de frameworks die de CEO graag zou willen zien in de implementatie.  
  
In DeepPavlov definiëren configuratiebestanden de processing pipeline en bestaan ​​uit vier hoofdsecties: dataset\_reader, dataset\_iterator, chainer en train.  
  
**Grote variatie in support (van dataset types tot word embeddings)**  
DeepPavlov heeft een standaard compatibiliteit met variërende dataset types: wiki, json en txt. Ook kunnen configuratiebestanden worden aangepast om andere dataset types te lezen.  
Verder worden fasttext en word2vec allebei gesupport door DeepPavlov.   
  
**Eventuele verbinding met Elasticsearch supported**  
De support van Deeppavlov heeft aangegeven dat het eventueel mogelijk is om DeepPavlov te laten verbinden met Elasticsearch door dat zelf in de reader configuratie file te definiëren.

**Deelconclusie**Er worden twee frameworks gekozen om mee te werken, één betaalde en één open source, namelijk Watson en DeepPavlov. Dit komt doordat ze beiden voldoen aan een groot aantal gekozen functionaliteiten, met name in het veld van ML en DL.

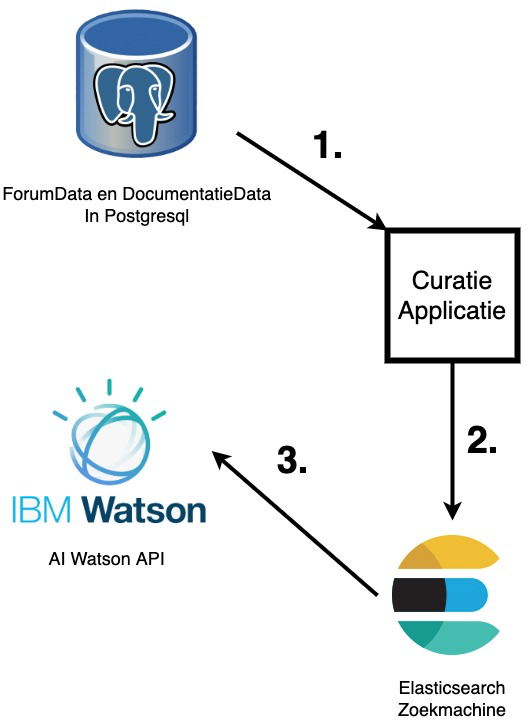
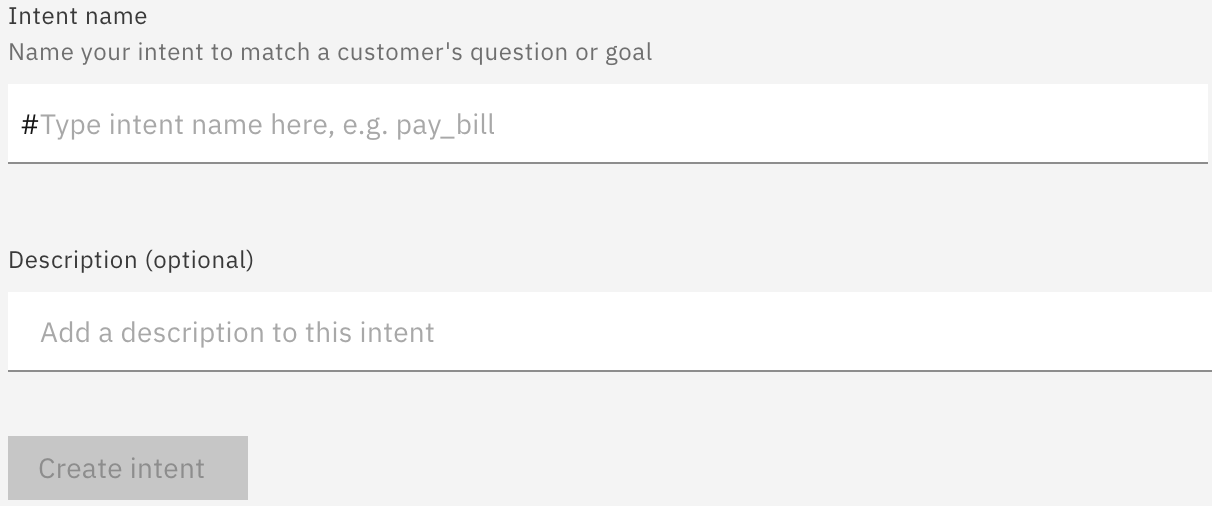
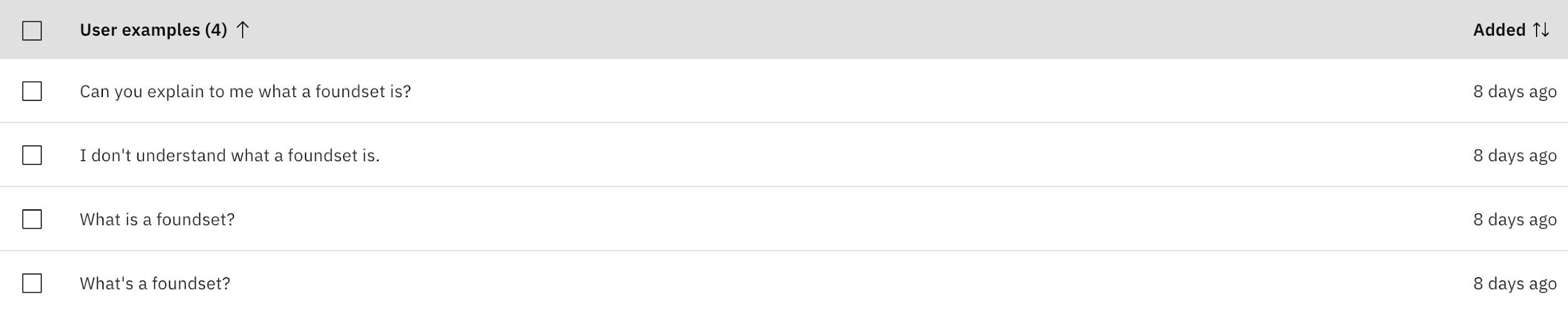
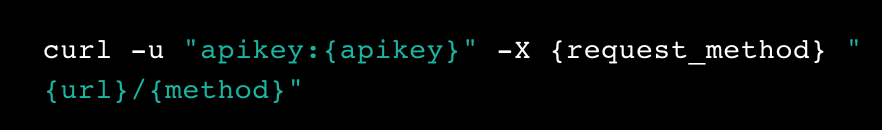
# **9. Ontwerp en realisatie van het product**

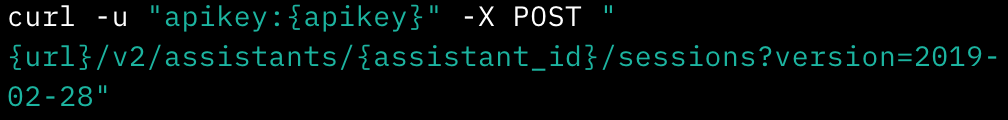
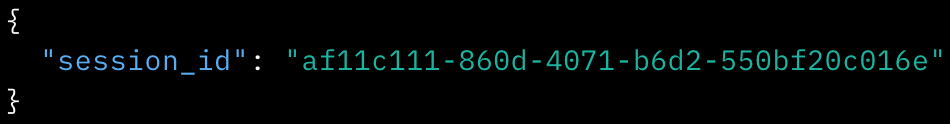
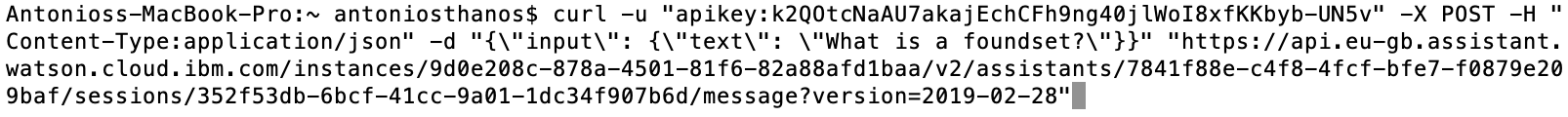
*In dit hoofdstuk worden de ontwerpen van de producten behandeld en hun realisatie. Zoals al eerder benoemd, zullen de forumdata en de wiki documentatie als datasets fungeren voor de chatbot. Maar omdat de data “grof” is moet de data worden geprepareerd, hiervoor is een curatieapplicatie ontworpen..* *Allereerst wordt het ontwerp van de curatieapplicatie besproken en de realisatie ervan. Vervolgens de poging tot realisatie van Watson i.c.m. elasticsearch, en als derde de poging met DeepPavlov. Ook worden er tests uitgevoerd op het gerealiseerde product.*

## **9.1 Curatie Applicatie**

**Het cureren van de huidige posts**Om de forumdata duidelijker te maken en te prepareren voor de chatbot is er een curatie applicatie, ontwikkeld in Javascript met de Servoy uitbreidingen, voor de medewerkers die de mogelijk biedt om alle forum posts te cureren op relevantie waarmee wordt aangegeven welke Q&A-paren relevant zijn. De gecureerde paren kunnen dan zowel naar de nieuwe zoekmachine elasticsearch gestuurd worden, als naar de tweede benadering met DeepPavlov. Hieronder wordt de applicatie weergeven, waarvan een grotere versie te zien is in bijlage 6.  
  
  
De linkerhelft van het scherm bevat de topics, wat het bijbehorende ID, de titel en de datum van creatie laat zien. Het rechterscherm laat de posts zien, met de naam van de poster, de inhoud van de post en de relevantie. Er kan een topic worden geselecteerd door de gebruiker waarvan er dan rechts de bijbehorende posts van dat topic worden weergeven. Het is aan de gebruiker om de relevantie te selecteren (Q, A of leeg als het irrelevant is) die standaard leeg wordt weergeven. Na alles te hebben ingevuld drukt de gebruiker op “submit” waarna de postparen worden opgeslagen. Op dat moment wordt de waarde checked\_relevance van het betreffende topic in de kopie database op “true” gezet. Hierdoor verdwijnt het topic uit de lijst.   
  
In het geval dat de medewerker toch nog iets wilt veranderen kan hij op het knopje “recent records” drukken wat de recentste records laadt, door alle topics op te halen met waar checked\_relevance op “true. Op dat moment kan er een herevaluatie van het topic worden uitgevoerd. Er worden standaard 200 records laten zien, waarna bij het naar onder scrollen er automatisch meer worden geladen door Servoy.  
  
**Dataopslag**Aanvankelijk werd er vanuit gegaan om via de hoofd-database met een read only account de gegevens op te sturen naar een temporary database in Servoy en ze vervolgens verder te sturen naar de gewenste locatie. Op deze manier hoefden er geen nieuwe databases te worden gecreëerd.  
  
Het idee van de temporary database bleek tijdens realisatie niet rendabel omdat het in de Servoy taal voor veel onomzeilbare complicaties veroorzaakte in de code. Zo ontstonden er veel moeilijkheden bij het aanroepen van de database records die in Servoy in de vorm van zogenoemde “foundsets” komen en ontstonden er problemen in te runnen loops. Het werd een onnodige puzzel die beter op viel te lossen door een kopie van de originele database te maken en daar mee te werken.  
  
De volgende velden zijn geselecteerd uit de originele records die worden opgeslagen in de nieuwe dataset:  
post\_id als primary key, topic\_id, forum\_id, poster\_id, post\_time, post\_text  
  
Ook werd er natuurlijk het nieuwe veld relevantie bijgevoegd maar ook het veld insert\_time, omdat de records moeten worden gerangschikt op insert time als ze opnieuw worden opgehaald in de curatieapplicatie.

## **9.2 Watson met Elasticsearch**

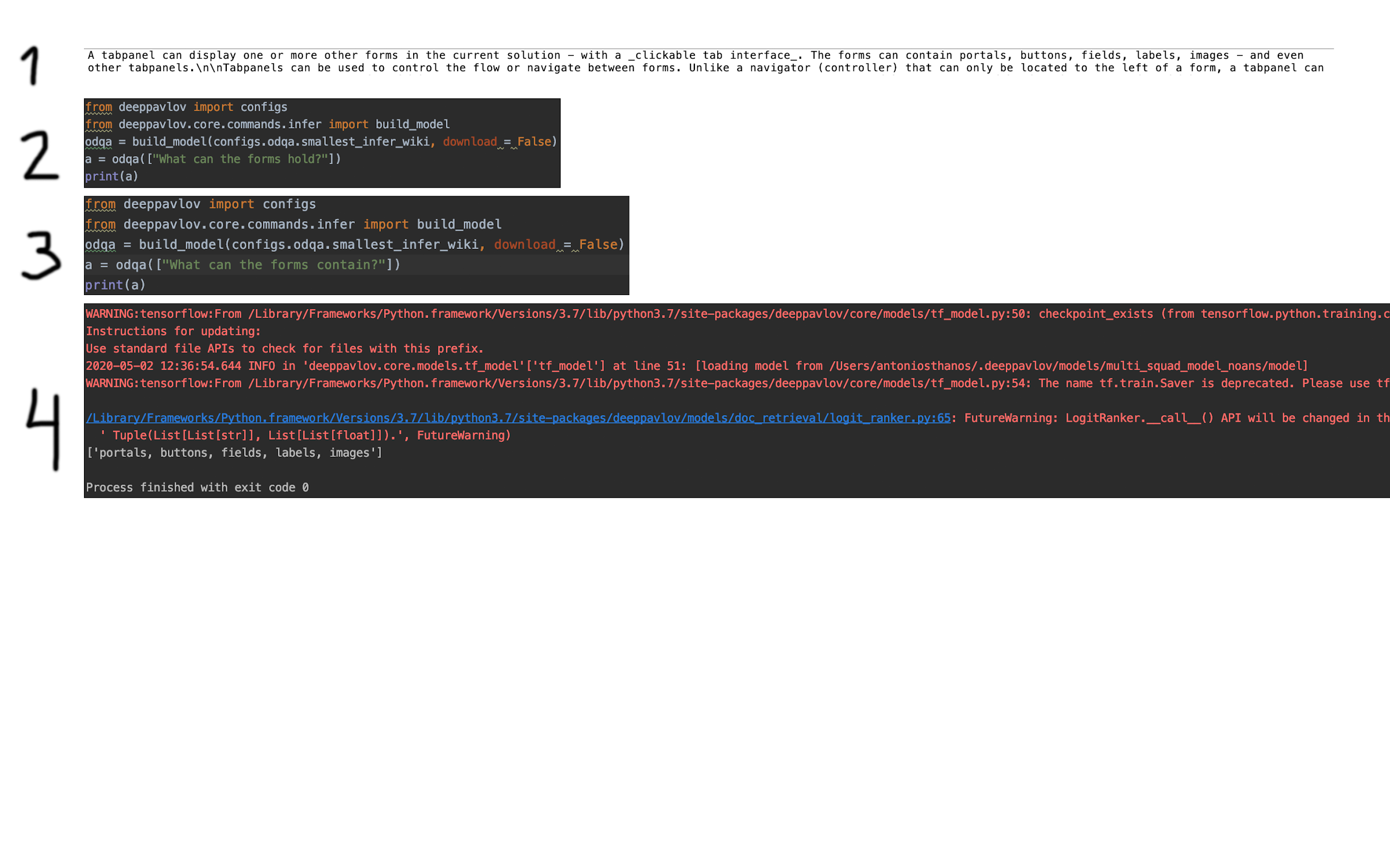
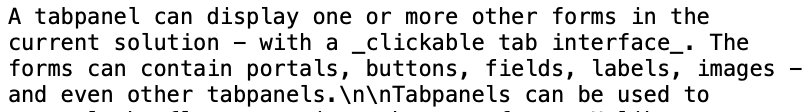
De volgende diagram weergeeft de structuur die bedacht is om de data vanuit de postgresql database via de curatieapplicatie in elasticsearch op te slaan en vervolgens te laten werken met Watson.   
****  
*Weergave structuur met Watson implementatie*  
  
**Elasticsearch installatie en dataopslag**Elasticsearch is op macOS met de package manager homebrew geïnstalleerd, hierna kan de service worden opgestart in de terminal met de commando’s. De opslag vanaf de curatie app naar elasticsearch wordt gedaan via Javascript i.c.m. Servoy. Er wordt gebruik gemaakt van een van Servoy’s eigen klassen, namelijk de “*http rest client”* klasse. In bijlage 7 wordt de implementatie hiervan weergegeven.   
  
  
  
**Watson via de user interface**Watson levert vanuit de websitepagina alle opties aan om de personal assistant te bouwen. Door zelf een IBM account aan te maken en online een watson assistant te creëren. Kan gebruik gemaakt worden van skills, dialog, examples, counterexamples, search skill enz.. om de chatbot vorm te geven.  
  
Daarna kunnen examples worden meegegeven zodat Watson die allemaal herkent als ze voorkomen:  
  
  
Het probleem om dit via de website te doen is alleen dat het handmatige invullen bij een grote hoeveelheid data erg veel tijd en werk gaat kosten. In het geval van Servoy is dit dus ook een probleem, omdat er namelijk een significant aantal posts zijn op het forum. het is niet praktisch en kost te veel tijd om voor al deze 100.000 posts entities, intents en ook nog meerdere examples te typen zodat een chatbot dat kan herkennen. Gelukkig zijn er ook er hiervoor ook andere mogelijkheden. Het automatiseren van deze hulpmiddelen is wat er moet gebeuren.   
  
Servoy is tegelijkertijd op zoek naar een manier om dus antwoorden op vragen van de gebruikers te geven die al zijn beantwoord op het forum zelfs als deze vragen met synoniemen of op een semantische manier gesteld worden. Dit zou met de hand nog veel meer tijd kosten.  
  
  
**Alternatief gebruik van Watson: curl**Omdat de standaard Watson interface dus geen nut had door het grote aantal records. Werd er gekozen om toch een andere manier die Watson support te gebruiken voor de verbinding tussen Servoy en Watson, namelijk curl. Het benaderen met curl wordt gedaan door gebruik te maken van de gegevens van de eigen watson assistant:  
- DeWatson Assistant ID  
- Watson Assistant API KEY   
- Assistant URL   
  
Als het intent er eenmaal is kan de intent via curl ingevoerd worden en herkend worden. Watson kan zelf begrijpen of de input lijkt op een van de voorbeelden (hoe meer voorbeelden hoe accurater).   
  
Dit wordt gedaan door eerst een sessie te creëren en vervolgens een input mee te geven, waarna Watson het juiste antwoord teruggeeft. Commando’s Met de volgende stijl van cUrl:  
****

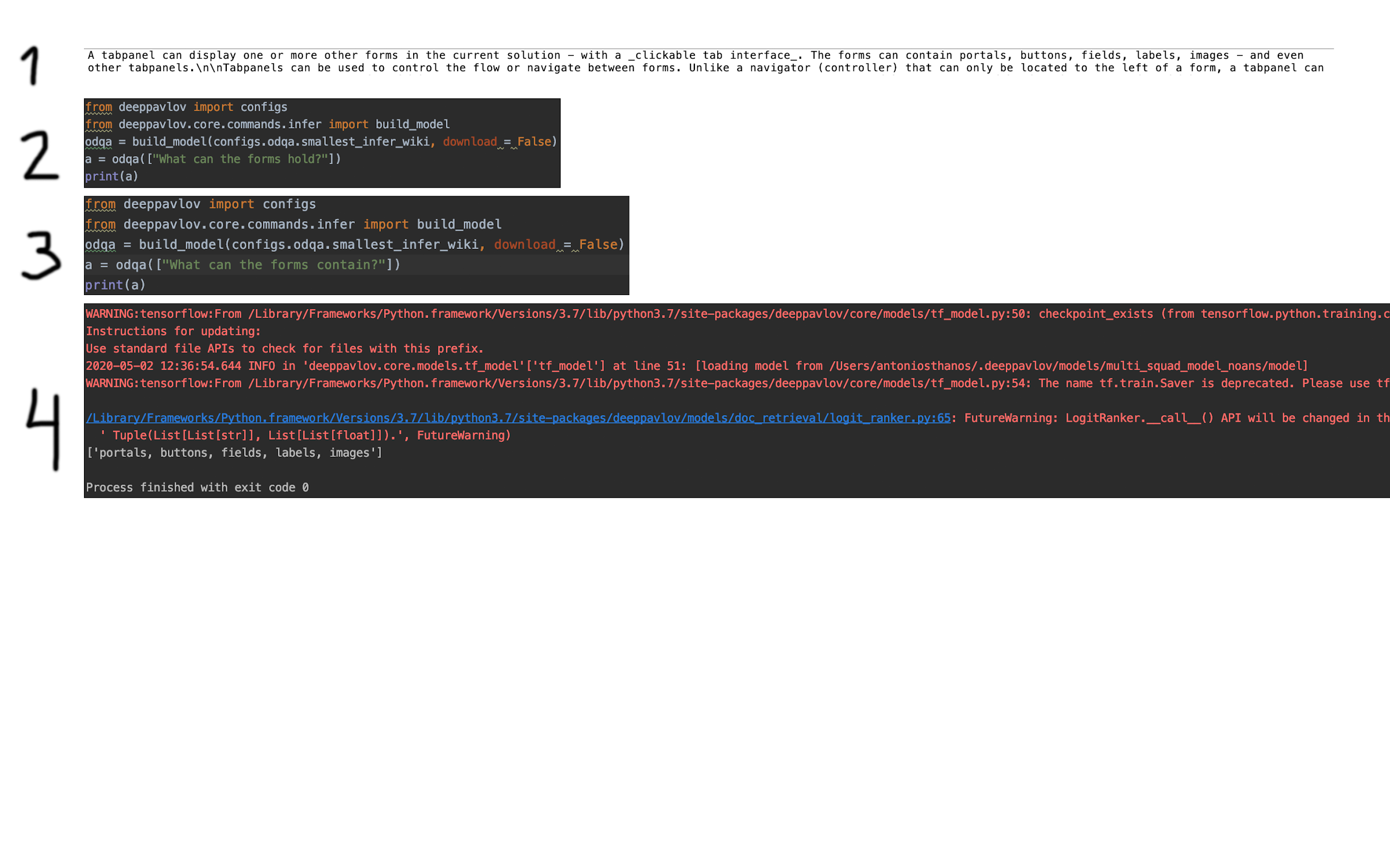
Met het volgende commando wordt een session gecreëerd:  
Waarna de gebruiker dan een sessie id krijgt in het volgende formaat :  
Als de gebruiker daarna dit sessie-id opneemt in een post request met daarin de vraag van de gebruiker:  
geeft watson vervolgens antwoord als het intent herkend wordt:  
  
  
Als de gebruiker dan zijn vraag intypt zoekt watson deze op met de intents die hij heeft en geeft terug of hij iets kan vinden of niet. De bedoeling is dat als hij dit niet vindt, hij dan een nieuwe request stuurt naar het kanaal waar open vragen worden behandeld door Servoy, zodat dit opgepakt kan worden.  
  
**Alternatief gebruik van Watson: logstash**Na logstash heel lang te hebben geprobeerd te gebruiken, was de conclusie dat het toch niet de moeite waard was. Er kwamen een aantal heel hardnekkige errors terug die bijna onmogelijk waren om te verhelpen. Dit wordt verder besproken in hoofdstuk 10.

## **9.3 Werking van DeepPavlov**

**Forum FAQ met Elasticsearch**  
Na verdere communicatie met de support van DeepPavlov via het forum bleek dat DeepPavlov geen standaard support biedt voor Elasticsearch. Maar dat dit misschien wel zou kunnen met een custom reader. Ook is het zo dat DeepPavlov van zichzelf al een snelle zoekmachine heeft.  
  
**Configuratie**  
Om de ODQA en FAQ met DeepPavlov werkende te krijgen wordt er met een aantal concepten en termen gewerkt die hieronder worden opgenoemd en worden uitgelegd.

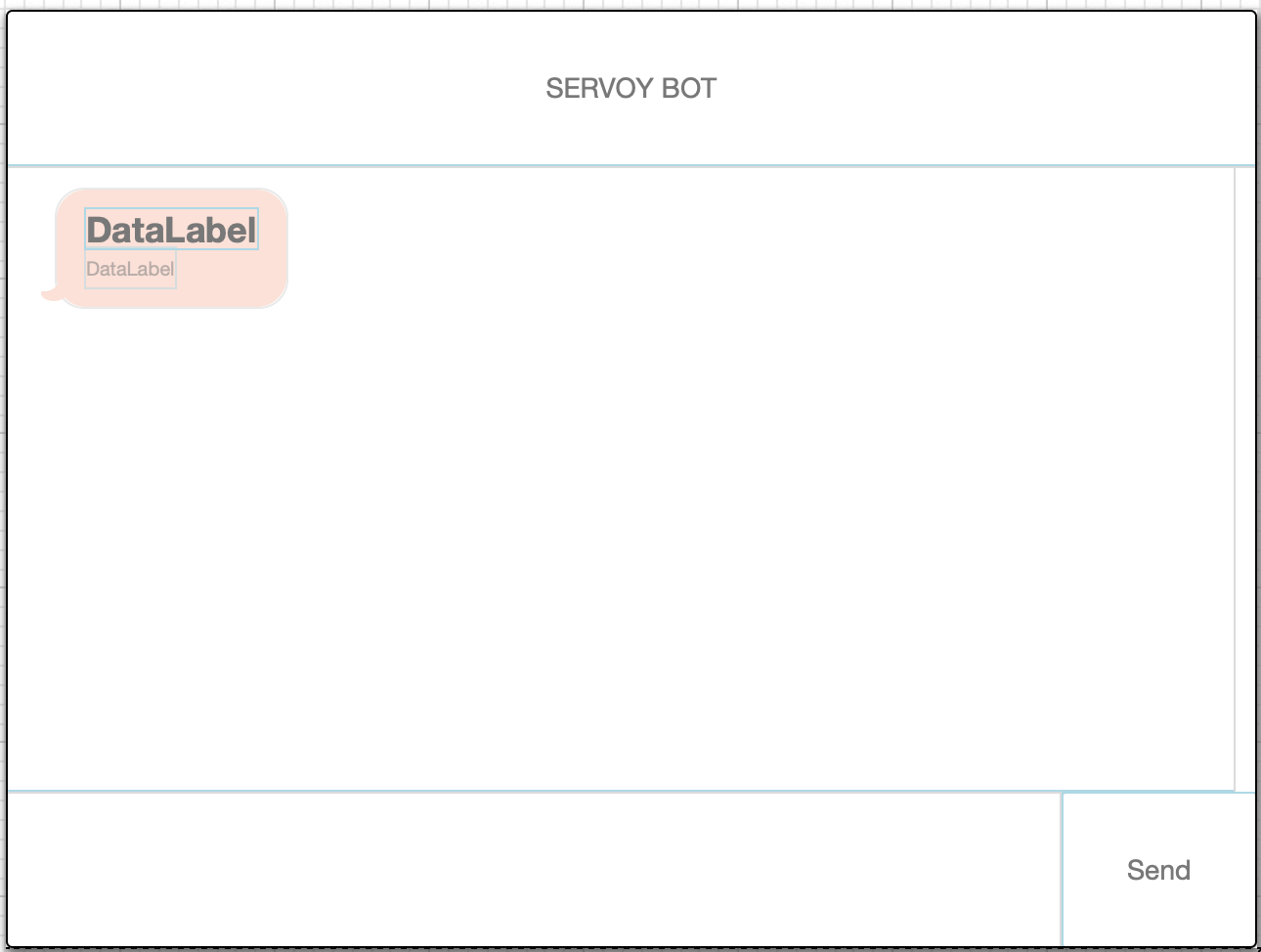
5 hoofd secties in de configuratie bestanden **Dataset\_reader** - Leest de dataset uit de data\_path directory. Leesbestanden zijn allemaal data\_types + extensie **Dataset\_iterator** - itereert over de bestanden uit de dataset. **Train** - Bevat de configuratie van het trainingsproces  
**metadata** -bevat extra informatie zoals downloads  
**chainer** - welke componenten gerund worden en in welke volgorde  
  
De ranker en reader configuratiebestanden voor de documentatie zijn te zien in bijlage 8 en 9.  
  
  
**ODQA Standaard Datasets**DeepPavlov’s ODQA kent drie standaard vormen van datasets waar de ranker mee kan werken, namelijk:  
wiki — specifiek een Wikipedia dump (db file, zoals bijvoorbeeld enwiki.db) geëxtraheerd door [wikiextractor](https://github.com/attardi/wikiextractor,),  
txt — een pad naar een folder met tekstbestanden,  
json — JSON files, die geformatteerd moeten zijn als een lijst met dictionaries die de “*title”* en “*doc”* sleutelwoorden bevatten (Konovalov, 2019).  
  
**FAQ Standaard Datasets**  
De FAQ heeft twee standaard datasets, namelijk:  
json - precies op dezelfde manier als de ODQA  
CSV - Een Comma Separated Values (**CSV**) plain text bestand die een lijst aan data bevat met kolommen.  
  
**Datasets gekozen  
.**txt files en CSV files. De forumdata kan worden toegevoegd aan een CSV en zo worden gebruikt voor de FAQ. In het CSV-bestand kunnen de Q&A-waarden van de curatie app ook makkelijk worden opgeslagen. Qua datasets leek het beste bestandstype text files om het feit dat de servoy niet alle informatie op wikipedia heeft, wel een servoy wiki maar dat is uiteraard niet hetzelfde.  
JSON klonk ingewikkelder dan puur textfiles te downloaden. Om de textfiles te download voerde ik het volgende functional sql commando uit via PostgreSQL:  
  
**Downloaden van textfiles uit de kopie database**Om de textfiles uit de database te krijgen en in een folder op de mac werd de volgende query gerund op de databasekopie. ****  
  
**Twee manieren van implementatie**  
Er waren twee manieren om dit te implementeren: of om alles zelf lokaal te installeren via de terminal of via docker. De voorkeur ging uit naar Docker omdat dat sneller zou zijn en dus handiger. De volgende moest met docker worden gerund:  
  
“*docker run -e CONFIG=ner/ner\_ontonotes\_bert -p 5000:5000\  
 -v /Users/antoniosthanos/Desktop/docker\_dp\_components\_volume/: root/.deeppavlov \  
 -v /Users/antoniosthanos/Desktop/Pavlov\ configuration\ files//:/venv \  
 deeppavlov/base-cpu:latest”*

**Docker errors**  
Helaas trad er bij het uitvoeren continu een error op wat betreft de *“spacey”* package. Die kon niet geïnstalleerd worden waardoor het proces gekillt werd door een error code. Ik probeerde zelf de spacey package apart te downloaden, wat wel lukte. Alsnog had de aparte installatie geen effect, hij verwijderde de package simpelweg en probeerde hem dan weer opnieuw te downloaden. Na overleg met een collega was de conclusie dat het docker script verouderd was waardoor hij niet goed functioneerde. Het zou veel werk worden om het zelf aan te passen.   
  
**Support niet optimaal**  
Ook bij DeepPavlov was de support niet optimaal, de deeppavlov forums worden vaak niet tijdig beantwoordt en het indienen van een contactformulier resulteert in geen gehoor. Toen er na drie weken eindelijk antwoord werd gegeven werd er gezegd: “dat deze docker error vaker voorkomt en er op dit moment nog geen oplossing voor is.  
  
**Manier 2: Runnen met Python via Pycharm**Door deze tegenvallers werd er daarom toch gekozen voor een manuele installatie. Voor de installatie via de terminal moest moest eerst een python virtual environment (venv) gecreëerd en geactiveerd worden (“Python virtual environments”. (z.d.)., 2014). Door “pip3 install deeppavlov” te runnen in de venv installeerde hij de packages. Ook kon het gedaan worden via Pycharm’s configuratie opties bij de project Interpreter, Deeppavlov te zoeken en het zo te installeren.  
 **Natural Language Toolkit (NLTK) -DATA**Bij de implementatie trad er een error op betreft de NLTK-DATA. NLTK is een toonaangevend platform voor het bouwen van Python-programma's om te werken met menselijke taalgegevens. Na onderzoek op het internet werd nog een stukje code (zie bijlage 10) toegevoegd. Hierna verscheen een scherm tijdens het runnen waarin de missende packages konden worden gedownload en geïnstalleerd (zie bijlage 11).  
  
**Error bij het lezen van bestanden**  
Verbazend was dat hij soms zelfs bij een kleine hoeveelheid files van 5, soms wel goed en soms niet goed ging. De volgende error trad op: *“python UnicodeDecodeError: 'utf-8' codec can't decode byte 0xd9 in position 99: invalid continuation byte”*. Na veel onderzoek bleek dit te komen door het openen van een antwoord met de text editor en het saven na een kleine bewerking voor test redenen. Hierdoor veranderde namelijk de encoding. Door de csv files voorlopig via het internet te bewerken wordt de encoding niet aangepast en wordt de file wel gelezen.  
  
  
  
**Werkende WIKI-ODQA**  
Door met de volgende twee stukken code (volledige implementatie is te zien in bijlage 12), de DeepPavlov implementatie te runnen:  
  
Waarin er dus twee vragen staan met synoniemen worden doorgegeven. Waarvan het antwoord als volgt vermeld staat in de text file:  
  
  
  
Weet DeepPavlov in beide gevallen het juiste antwoord antwoord te vinden:

  
  
Dit bewijst dat DeepPavlov werkt.  
   
  
**Gebruikte computer te zwak voor grote hoeveelheid bestanden**Tijdens het uitvoeren van het trainen werd het duidelijk dat een macbook pro 2015 niet bij verre na niet genoeg hardware capaciteit heeft om met 80.000 textfiles te werken. Zo duurde het uitgerekend 10 uur om al deze files maar één keer te kunnen trainen. Daarom is de conclusie getrokken dat de chatbot op een sterkere server moet worden gerund.

**Virtuele Machine**Servoy maakt al gebruik van Amazon Web Services (AWS). Het ODQA en FAQ model kunnen hierop geïmplementeerd worden. Doordat de modellen vrij complex zijn, is het ook wel logisch dat er veel rekenkracht voor nodig is. Volgens DeepPavlov hebben beide modellen ongeveer 24 GB aan RAM nodig. Het is mogelijk om ze uit te voeren op een 16 GB-computer, maar de swap size moet minimaal 8 GB zijn.  
  
**Passend BesturingsSysteem**  
Er werd gebruik gemaakt van zowel een windows computer als een mac om deeppavlov draaiende te krijgen. Helaas lukte dit alleen maar op de mac. Om zeker te zijn dat het probleem niet aan de computer lag, werd er besloten om de Virtual Machine met windows op te zetten. Pycharm werd erop geïnstalleerd, python en pip werden toegevoegd aan path variables. pip3 geüpgraded. *“pip3 install deeppavlov”* gedaan. extra libraries geïnstalleerd met *“python -m deeppavlov install deeppavlov/configs/odqa/en\_odqa\_infer\_wiki.json”*  
Helaas gaven beide windows installaties de hardnekkige error:   
*“error database\_name is already in use”*Hierdoor werd besloten om het uiteindelijk op Linux te proberen aangezien het probleem zich alleen met windows voordoet.  
  
  
  
  
  
  
**Opzetten op Linux**  
Om het nu verder werkende te krijgen op de linux virtual machine (want het werkt niet op windows) werd er een website aangeraden door de Devops collega (Tecmint, 2017):  
  
1. $ sudo yum install tigervnc-server  
2. $ vncpasswd  
3. # cp /lib/systemd/system/vncserver@.service /etc/systemd/system/vncserver@:1.service  
4. # vi /etc/systemd/system/vncserver@\:1.service  
  
Door de het service bestand aan te passen met de variabelen van de huidige computer. en ook VNC te installeren op de mac werd het mogelijk om te verbinden met de server via het adres.   
Helaas was het scherm zwart en was het niet helemaal goed geconfigureerd. Helaas was er niet meer genoeg tijd over om de server werkende te krijgen.

## **9.4 De Chatbot**

**Chatbot Flow**  
Op het einde is het de bedoeling dat de klant aangeeft of het antwoord wat hij terug krijg voldoende was of niet. Als het antwoord niet goed is behoort de klant zijn vraag automatisch te worden doorgestuurd naar een echte medewerker van Servoy. Een schets hiervan is te zien in bijlage 13  
 **Servoy en REST API**Voor nu wordt het chatbotprototype gebouwd met Servoy en REST wat ook gesupport wordt door DeepPavlov.Om de REST API te gebruiken voor het prototype moet telkens bij het toevoegen van nieuwe data moet de ranker het model eerst opnieuw getraind worden via de python code. Daarna valt de REST API zowel met het forum als de documentatie te runnen. Beide geven antwoorden correct terug en ook een lijst met probability scores (zie bijlage 14 voor een voorbeeld) zoals verwacht. Dit bewijst dat de REST API correct functioneert.   
  
**In de toekomst met Angular JS**  
Later is het de bedoeling om de frontend van de chatbot in Angularjs te schrijven en de backend in Java. De python code valt wel te verbinden met java via een connector zodat hij uiteindelijk als java eclipse plugin in de developer zal passen.  
  
**Front end**De front end van de applicatie wordt eerst weer in Servoy ontwikkeld en bij eventuele tijdsoverschot zou een geavanceerdere implementatie met Angular JS kunnen worden uitgevoerd.  
  
**Werking van de backend van de Chatbot**1. Het bericht van de gebruiker wordt door de chatbot opgeslagen  
2. Het bericht wordt via een REST Call naar localhost gestuurd waar een DeepPavlov model wordt aangeroepen, of ODQA of FAQ.  
3. Uit de responsbody word het antwoord geëxtraheerd  
4. Het antwoord wordt teruggegeven aan de gebruiker   
  
De volledige code is te zien in bijlage 15 en 16.

**Het design van het logo**Het volgende logo is gedesigned om te gebruiken voor in de IDE en/of voor de website van Servoy:

## **9.5 Performance tests**

Door de chatbot te testen met een aantal vragen uit een vooropgestelde csv file met Q&A-paren en deze vragen ook op alternatieve manieren te stellen, kan de bot getoetst worden. Er wordt gekeken in hoeverre de chatbot in staat is om correct antwoord te geven en waar de (voorlopige) limieten zich bevinden (Faber Novel, 2020).  
  
Er is besloten om op bepaalde categorieën te testen. De categorieën die getest worden met staan hieronder vermeld. Een lijst van alle testgevallen is te zien in bijlage 17.  
 **1 Algemeen testen**In de algemene test worden de standaardgevallen getest die andere soort tests op een specifieke eigenschap worden getest, puur om te fungeren als controlegroep. **2 Domeinspecifiek testen**De tweede fase betreft het domeinspecifiek testen. Dit wordt gedaan om te zien tot welke complexiteit de functionaliteiten van het framework van de bot weten te gaan. **Eigenschap 1 Synoniemen** Door synoniemen te gebruiken wordt er gekeken of de chatbot “slim” is en of de   
 gekozen functionaliteiten correct functioneren.  
 **Eigenschap 2 Semantisch Alternatief gestelde vraag** Semantisch variërende zinnen bedoelen qua betekenis hetzelfde, alleen de   
 locatie van de woorden in de zin is veranderd waardoor de zin, met eventueel   
 ook een nihil aantal extra woorden. Hierdoor valt het niet onder eigenschap 3.  
**3** **Stress/Limiet testen**De derde fase is het testen van de limieten van onze chatbot. Voor de eerste twee stappen gingen we uit van zinvolle zinnen. Deze laatste stap zal ons laten zien wat hoe bestendig de chatbot is zodra het benaderd wordt met het onbekende en onverwachte.   
  
**Eigenschap 1: Aantal woorden**  
 Door extra woorden die niet zozeer met de vraag te maken hebben, wordt er als het   
 ware een ruis gecreëerd die als afleiding fungeert. Hierbij is het interessant om te   
 zien hoeveel overbodige woorden de chatbot aankan voordat hij nonsens teruggeeft.   
 In de tegenovergestelde richting, door te weinig woorden op te schrijven gaat het   
 juist om de sleutelwoorden. Hierbij is het interessant om te zien en hoe goed hij die   
 weet te koppelen aan het juiste Q&A paar.

# **10. Watson Implementatieproblemen**

*Dit hoofdstuk gaat over de ondervonden problemen tijdens de implementatie van Watson die het niet toestonden om verder te gaan met de implementatie van het Watson framework i.v.m. mede door de opgelopen errors en tijdsdruk.*Helaas ontstonden er bij de implementatie van het Watson Framework een aantal complicaties die hieronder besproken zullen worden.

**Watson met Curl ongeschikt**  
Allereerst werd er gekozen om watson via curl te benaderen omdat ook makkelijk te koppelen viel aan een Servoy Javascript applicatie. Bij implementatie werd ontdekt dat curl maar beperkte mogelijkheden heeft qua aanroepen en niet de volledige API kan benaderen. Zo is er geen manier is om intent te creëren alleen het basische terugkrijgen van een antwoord met een bepaald intent. Hierdoor was Curl ongeschikt om complexere taalherkenning met verrichten. Daarom werd er gekozen om het toch via Java te proberen aangezien dit ook te implementeren valt in de Servoy IDE als aparte eclipse plugin.  
  
**Lage Watson Support**Ook gaf de customer support aan tijdens het contact met IBM Watson dat er geen mogelijkheid was om support van een software engineer te krijgen aangezien Servoy qua omvang een te relatief klein bedrijf zou zijn. Daarnaast waren ze net overgestapt op een nieuw forum wat nog relatief leeg was terwijl het oude forum al onbereikbaar was geworden en was er op die manier dus geen advies te vinden.   
  
**Watson Assistant API**  
Ook met het volgen van de instructies om Watson Assistant V1 en V2 (IBM Cloud API Docs Watson Assistant v1, z.d.) bleek het achteraf dat voor het volledige functioneren, handmatige intent creatie, example creatie en synoniem creatie allemaal handmatig gedaan zouden moeten worden wat een hele grote hoeveelheid extra tijd zou gaan kosten.   
  
**Meerdere pakketten vereist**  
Ook werd het geleidelijk duidelijk dat er een steeds meer pakketten nodig zouden zijn om de chatbot met alle functionaliteiten te laten werken, hierdoor werd de prijs ook steeds onvoorspelbaarder. Zo was er voor NLP naast de NLC ook nog een aparte NLU, tone analyzer en anderen. Door al deze punten bij elkaar en omdat het timeboxlimiet naderde, werd er besloten om verder te gaan met DeepPavlov voor de chatbot implementatie.   
  
**Verwachtingen**  
De verwachtingen van een werkende Watson API met Elasticsearch logstash waren als volgt: Nadat er een vraag gesteld zou zijn, zou Watson allereerst de Elasticsearch zoekmachine aanroepen om uit de documentatie en het forum snel te kunnen vinden welke documenten of posts eventueel goed antwoord bevatten op de vraag. Hierna zou Watson met zijn eigen API het antwoord uit de tekst extraheren en teruggeven.

# **11. Resultaten**

*In dit hoofdstuk worden de resultaten van de tests besproken, ook ten opzichte van de requirements. Ook wordt er besproken of de functionaliteiten voldoen.*

De testresultaten zijn grotendeels positief en behaalden de requirements   
FReq’s 1, 2, 3, 4 en 6. Wat betreft FReq 8. is alleen het front-end gedeelte deels functionerend. Voor FReq 5 is bewijs gevonden op het forum.  
Door veel gebruikersvriendelijke FAQ-regels toe te voegen kan FReq 7 ook bereikt worden  
  
**Non functionele requirements**Van de non functionele requirements zijn de requirements:   
NfReq’s 2, 3 en 6. behaald. Ook NfReq 1. kan behaald worden op de macbook bij een niet al te grote dataset. Of het op de virtual machine werkt is nog niet bewezen.  
  
NfReq’s 4, 5, 7 zijn nog niet behaald. Voor NfReq 4 is bewijs gevonden door op het forum om confirmatie te vragen. Ook zal naar verwachting NfReq 7. werken zodra de virtuele machine is opgezet en DeepPavlov daarop draaiende is.Na het runnen van de test genoemd in 9.5 zijn de volgende resultaten opgemaakt**:**

| **Testgeval #** | **Soort testen** | **Eigenschap** | **Uitslag** |
| --- | --- | --- | --- |
| **1** | **Algemeen** | **n.v.t.** | Correct |
| **2** | **Algemeen** | **n.v.t.** | Correct |
| **3** | **Algemeen** | **n.v.t.** | Correct |
| **4** | **Domeinspecifiek** | **Synoniemen** | Correct |
| **5** | **Domeinspecifiek** | **Synoniemen** | Correct |
| **6** | **Domeinspecifiek** | **Synoniemen** | Correct |
| **7** | **Domeinspecifiek** | **Semantisch Alternatief** | Correct |
| **8** | **Domeinspecifiek** | **Semantisch Alternatief** | Correct |
| **9** | **Domeinspecifiek** | **Semantisch Alternatief** | Correct |
| **10** | **Limiet testen** | **Aantal woorden** | Correct |
| **11** | **Limiet testen** | **Aantal woorden** | Failed |
| **12** | **Limiet testen** | **twee vragen in één zin** | Correct |
| **13** | **Limiet testen** | **twee vragen in één zin** | Failed |

De conclusie is dat de meeste tests geslaagd zijn en ook moeilijke domeinspecifieke zoals synoniemherkenning en semantisch variërend.

# **12. Evaluatie**

*In dit hoofdstuk wordt de prestatie van het DeepPavlov framework, dat de gekozen functionaliteiten gebruikt, geëvalueerd.* **Snelle runtijd na het trainen**Als er nieuwe files bij de dataset komen moet chatbot eerst opnieuw worden getraind. Maar alleen het aanroepen geeft zeer snelle responstijden van onder de halve seconde. Bij een grotere dataset en een sterke computer blijft dit resultaat naar verwachting evenredig. **Precisie**De chatbot is bijvoorbeeld in staat om, uit honderd text files met naar schatting een totaal aantal aan 3000 woorden, het goede antwoord te extraheren met een lengte van 5 woorden.  
Ook verlopen de meeste testgevallen positief wat duidt op een positieve vervolg met de gekozen functionaliteiten van de DeepPavlov API. Ook werken de moeilijke taalkundige onderwerpen van synoniemherkenning en semantiekherkenning met de FAQ. **Nog geen Analytics aanwezig**Op dit moment is er nog onduidelijk hoe goed DeepPavlov werkt met grote datasets. Omdit goed te begrijpen zou ik willen aanraden om analytics toe te passen, om extra informatie te krijgen over de performance van de applicatie. er moet gebruikt worden gemaakt van analytics om te zien wat de gebruikers vragen aan deeppavlov en om te zien  
  
**Backend nog niet geautomatiseerd**Zorg ervoor dat veranderingen in FAQ automatisch worden doorgevoerd in de CSV file. En bij elke grote update van de documentatie de nieuwste versie wordt omgezet in text files en meegegeven aan de chatbot.Ook kunnen er nog meer simpele welkomstberichten, formaliteiten, keuzemenu’s en ook vragen voor feedback worden toegevoegd. Op dit moment werkt de gebruikersreview van het chatbot antwoord nog niet. Ook kan er nog geen medewerker worden ingeschakeld om antwoord te geven  
  
**Data nog niet 100% geprepareerd**Mocht de markup language toch een probleem worden verwijderen van markup language uit de textfiles zodat Deeppavlov beter kan lezen.  
  
**Front end nog niet geschikt voor Servoy IDE**Het maken van een passende front-end Angular die in de Servoy IDE gaat passen. Dit levert ook de mogelijkheid tot doorverkoop als embedded systeem aan Servoy’s klanten. **Nog geen multi-platform inzet geïmplementeerd.**Ook een advies zou zijn om de DeepPavlov chatbotstructuur te implementeren voor een chatbot die op Servoy’s website kan werken en zo potentieel nieuwe klanten 24/7 te woord kan staan. Misschien zou het zelfs gedaan voor Servoy’s facebook pagina.  
  
  
**Q&A classificatie nog niet duidelijk genoeg**Op dit moment zijn de Q & A nog niet duidelijk genoeg. Zo kan het zijn dat er in een topic twee posts samen de gehele vraag vormen, of twee antwoorden.

| | Post ID | Relevance | | --- | --- | | 1 | Q | | 2 | A | | 3 | A | | 4 | Q | | 5 | Q | | 6 | A |   ***Voorbeeld 1*** | | Post ID | Relevance | | --- | --- | | 1 | Q1 | | 2 | Q2 | | 3 | Q3 | | 4 | A2 | | 5 | A2 | | 6 | A1 | | 7 | A3 | | 8 | A1 |   ***Voorbeeld 2*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Tot nu toe wordt het cureren gedaan volgens voorbeeld 1. Toch kan het zijn dat er meerdere vragen worden gesteld in een topic of dat twee posts deel vormen van één specifieke vraag. Dat valt niet op te maken uit voorbeeld 1. Bij voorbeeld 2 kan dat wel doordat de Q’ en A’s   
  
worden er eerst drie vragen gesteld waarna er antwoorden worden gegeven. Maar zonder ID te geven aan de relevantie is het niet duidelijk bij welke vraag het antwoord hoort. Aangezien je niet altijd op kunt maken uit de chronologie welke Q bij welke A hoort.  
 **Trainen nog niet mogelijk**  
Het trainen van de antwoorden van de chatbot zal hoe dan ook moeten gebeuren. Als de chatbot een bepaald antwoord niet weet kan er een nieuwe regel worden toegevoegd aan de CSV file met de juist Q&A waarde zodat hij het antwoord in de toekomst wel weet te pakken.   
  
**Watson altijd later nog mogelijk**Mocht de DeepPavlov implementatie toch niet naar wens zijn, dan kan er altijd nog worden voortgeborduurd op de onafgemaakte Watson implementatie. De curatie applicatie werkt met Elasticsearch, dus dan is nog de verbinding tussen Elasticsearch, Watson en de curatieapplicatie nodig.

# **13. Conclusies**

*In het hoofdstuk wordt er kort antwoord gegeven op elk van de hoofd- en deelvragen.*Er bestaat een groot scala aan functionaliteiten in de markt, waarvan sommige gericht zijn op prestatiemaximalisatie, andere weer op het streven naar het menselijke aspect van de chatbot of op het maximaliseren van het gebruikscomfort. Sommige zorgen voor een dichtere structuur en andere juist weer voor een openere. Veel functionaliteiten werken samen met elkaar en vormen deel van andere om zo samen één samenhangend geheel te vormen.  
  
De problemen die worden ondervonden zijn de onvindbaarheid van antwoorden en ook de tijd die het kost om ze te vinden. Ook is de zoekmachine te traag en vereist vervanging. Verder is de dataset te ongeprepareerd.  
  
Om een implementatie te creëren die dit probleem kan oplossen is gekozen voor functionaliteiten NLP, ODQA, ranker, reader, word embedding, fasttext, elmo/bert, tf-idf, trainen en confidence score. Naar verwachting zullen deze efficiënt met elkaar kunnen aansluiten om een solide pakket te vormen en zo de ondervonden problemen op te lossen.  
  
Er zijn er groot aantal frameworks beschikbaar op de markt. De volgende lijst is opgesteld:  
BotUI, Cognitive affairs, Microsoft bot builder framework , IBM’s Watson Assistant, Google’s Dialog Flow, Rasa Stack, ANA.CHAT, Meya, Amazon Lex, Pandora Bots, Botpress, BotKit, Bottr, CeJS, Wit.ai, Chatterbot, Semantic Machines, Botman, [SAP Conversational AI](https://cai.tools.sap/), [GupShup](https://www.gupshup.io/developer/home), [Botsify](https://botsify.com/), [Flow XO](https://flowxo.com/), DeepPavlov.Er worden twee frameworks gekozen om mee te werken, één betaalde en één open source, namelijk Watson en DeepPavlov. Dit komt doordat ze beiden voldoen aan een groot aantal gekozen functionaliteiten, met name in het veld van ML en DL.   
  
Ondanks het feit dat watson betaald is en dus meer gebruiksgemak zou moeten bieden, waren er bij implementatie veel configuratie problemen bij de logstash implementatie die voortzetting onmogelijk maakten. Deeppavlov ondanks dat het open source-software is en ook met zijn eigen configuratie problemen kampt gaf wel een presenteerbaar resultaat wat de meeste testen heeft doorstaan.  
  
Zonder een sterke omgeving zal de chatbot zeer traag zijn. Daarom is er sterkere hardware nodig in de vorm van een virtuele server om te zorgen dat de functionaliteiten optimaal kunnen worden gebruikt. Er wordt verwacht dat hoe beter de computer, hoe sneller de chatbot functioneert.

# **14. Aanbevelingen**

*In dit hoofdstuk worden de aanbevelingen gegeven voor de verdere implementatie van de chatbot en extra nieuwe benaderingen die potentie hebben voor de toekomst van de chatbot en het bedrijf.* **➢ Deeppavlov i.c.m. Analytics**om goed te begrijpen over welke onderwerpen meer vragen zijn  
  
**➢ Automatisering**Zorg ervoor dat veranderingen in FAQ automatisch worden doorgevoerd in de CSV file. En bij elke grote update van de documentatie de nieuwste versie wordt omgezet in text files en meegegeven aan de chatbot. **➢ Inbouwen van standaardberichten**Simpele welkomstberichten, formaliteiten, Keuzemenu voor onderwerpen, vraag voor extra informatie, etc…  
  
**➢ Opnemen van gebruikersinteractie**er moet gebruikt worden gemaakt van analytics om te zien wat de gebruikers vragen aan deeppavlov en om te zien  
  
**➢ Vragen van feedback aan de gebruiker**Het Vragen van feedback aan de gebruiker aan het eind van een chatbot sessie om zo verbeterpunten te vinden.  
  
**➢ Java-eclipse plugin creëren voor de Servoy IDE**Het maken van een passende front-end van de chatbot met Angular. Hierbij ook mogelijke doorverkoop als embedded systeem in gedachte te houden. **➢ implementatie van het front-end deel**Het deel wat de gebruiker in staat stelt om zijn vraag te sturen naar een medewerker voor een correct antwoord indien de chatbot niet het juiste antwoord kon geven (zoals in bijlage 6 te zien is)  
 **➢ Verbreding van de inzet van de chatbot**Ook een advies zou zijn om de DeepPavlov chatbotstructuur te implementeren voor een chatbot die op Servoy’s website kan werken en zo potentieel nieuwe klanten 24/7 te woord kan staan. Misschien zou het zelfs gedaan voor Servoy’s facebook pagina.  
  
  
  
  
  
  
**➢ Q&A classificatie verduidelijken**Om verder te gaan met het de implementatie van de chatbot zou het advies zijn om de Q & A waardes beter te specificeren. Zodat de chatbot de Q&A paren duidelijk heeft.   
  
**➢ Eventueel Markup Language verwijderen**Mocht de markup language toch een probleem worden verwijderen van markup language uit de textfiles zodat Deeppavlov beter kan lezen

**➢ Inbouwen van de train mogelijk**  
Het trainen van de antwoorden van de chatbot zal hoe dan ook moeten gebeuren. Als de chatbot een bepaald antwoord niet weet kan er een nieuwe regel worden toegevoegd aan de CSV file met de juist Q&A waarde zodat hij het antwoord in de toekomst wel weet te pakken. **extra**

# **15. Bronnenlijst**

**Websites**    
About Servoy. (z.d.). Geraadpleegd van https://servoy.com/servoy/

* Servoy Documentation. (z.d.). Geraadpleegd van https://wiki.servoy.com/  
  DeepPavlov. (z.d.). Welcome to DeepPavlov’s Documentation! Geraadpleegd van http://docs.deeppavlov.ai/en/master/   
    
  Konovalov, V. (2019-02-04). Open-domain question answering with DeepPavlov. Geraadpleegd van https://medium.com/deeppavlov/open-domain-question-answering-with-deeppavlov-c665d2ee4d65  
    
  Watson Assistant. (z.d.). [Watson Assistant Informatie] Geraadpleegd van https://www.ibm.com/cloud/watson-assistant/  
    
  IBM Cloud API Docs Watson Assistant v1. (z.d.). [Watson API handleiding] Geraadpleegd van https://cloud.ibm.com/apidocs/assistant/assistant-v1?code=java#list-entity-value-synonyms

Gus Chat. 2017-12-13. 5 Steps Before Implementing a Chatbot in Your Company. Geraadpleegd van https://chatbotsmagazine.com/5-steps-before-implementing-a-chatbot-in-your-company-92cec726c3b1

Mugan, J. 2017-07-19. Chatbot: Theory and Practice. Geraadpleegd van -https://medium.com/intuitionmachine/chatbots-theory-and-practice-3274f7e6d648

Makadia, M. 2019-05-07. Complete Guide to Enterprise Chatbot Development. Geraadpleegd van -https://towardsdatascience.com/complete-guide-to-enterprise-chatbot-development-c377aa5e4ddc

Marr, B. 2016-12-06. What is The Difference Between Artificial Intelligence And Machine Learning. Geraadpleegd van -https://www.forbes.com/sites/bernardmarr/2016/12/06/what-is-the-difference-between-artificial-intelligence-and-machine-learning/#5fab0cd22742

Cezar, M. 2017-09-19. How to Install and Configure VNC Server in CentOS 7. Geraadpleegd van

https://www.tecmint.com/install-and-configure-vnc-server-in-centos-7/  
  
Nayak, P. 2019-10-25. Understanding searches better than ever before. Geraadpleegd van   
https://blog.google/products/search/search-language-understanding-bert/

Davydova, O. 2017-05-11. 25 Chatbot Platforms: A Comparative Table. Geraadpleegd van https://chatbotsjournal.com/25-chatbot-platforms-a-comparative-table-aeefc932eaff  
  
About MIPT. (z.d.). Geraadpleegd van https://mipt.ru/english/about/about-mipt/  
  
Types of Bots: An Overview. (z.d.). Geraadpleegd van http://botnerds.com/types-of-bots/  
  
Mindbowser. Chatbot Survey 2017. (z.d.). Geraadpleegd van http://www.club-cmmc.it/lettura/Chatbots\_Survey\_2017\_PDF.pdf  
  
Rehan, A. 2019-13-07. 9 Best Chatbot Development Frameworks to Build Powerful Bots. Geraadpleegd van https://geekflare.com/chatbot-development-frameworks/  
  
Artificial Solutions. Chatbots: The Definitive Guide (2020). (z.d.). Geraadpleegd van https://www.artificial-solutions.com/chatbots  
  
[Rossignol](https://www.macrumors.com/author/joe-rossignol/), J. 2016, 25-10. Apple Reports 4Q 2016 Results: $9B Profit on $46.9B Revenue, 45.5M iPhones. Geraadpleegd van https://www.macrumors.com/2016/10/25/earnings-4q-2016/  
  
Azure Bot Service pricing. (z.d.). Geraadpleegd van https://azure.microsoft.com/en-us/pricing/details/bot-service/  
  
Chatbots: what are they good for? (z.d.). Geraadpleegd van https://www.whoson.com/chatbots-ai/chatbots-what-are-they-good-for/  
  
Chatbot Features to set you Apart (z.d.). Geraadpleegd van https://www.ubisend.com/features  
  
Arthur Samuel (z.d.). Geraadpleegd van https://history-computer.com/ModernComputer/thinkers/Samuel.html

Khalid, F. 2016-29-05. Why do chatbots suck? Geraadpleegd van https://techcrunch.com/2016/05/29/why-do-chatbots-suck/  
  
Garbade, M. J. 2018-14-09. Clearing the Confusion: AI vs Machine Learning vs Deep Learning Differences. Gereedplaagd van https://towardsdatascience.com/clearing-the-confusion-ai-vs-machine-learning-vs-deep-learning-differences-fce69b21d5eb  
  
7 Characteristics of a Great Chatbot. (z.d.). Geraadpleegd van https://www.bluewolf.com/bluewolf-now/7-characteristics-great-chatbot  
  
Anurag. 2020-18-04. A Complete Guide on Chatbot Architecture. Geraadpleegd van https://www.newgenapps.com/blog/complete-guide-on-chatbot-architecture/  
  
Using Synonyms. (z.d.). Geraadpleegd van https://support.snatchbot.me/docs/using-synonyms

ELMo. (z.d.). Geraadpleegd van https://allennlp.org/elmo  
  
Alammar, J. 2018-03-12. The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning. Geraadpleegd van http://jalammar.github.io/illustrated-bert/

Lahoti, S. 2017-20-10. Top 4 chatbot development frameworks for developers. Geraadpleegd van https://hub.packtpub.com/top-4-chatbot-development-frameworks-developers/

[Algemene beschrijving van tf-idf]. (z.d.). Geraadpleegd van http://www.tfidf.com/  
  
[Vector representatie van document] (z.d.). Geraadpleegd van http://mlg.postech.ac.kr/research/nmf  
  
[Informatie over Watson] (z.d.). Geraadpleegd van https://watson-api-explorer.ng.bluemix.net/  
  
AthenaTech. 2019-21-10. AI, Machine Learning (ML) and Natural Language Processing (NLP). Geraadpleegd van https://athenatech.tech/f/ai-machine-learning-ml-and-natural-language-processing-nlp  
  
SnatchBot Team. 2019-22-12. The Future of Chatbots. Geraadpleegd van https://snatchbot.me/blog/120/future-of-chatbots  
  
Innoplexus. 2019-27-02. What is Named Entity Recognition? Geraadpleegd van https://medium.com/@Innoplexus/what-is-named-entity-recognition-7ed05beb7171#:~:text=Named%20Entity%20Recognition%20(NER)%20is,specific%20like%20a%20biomedical%20term.  
  
Monkeylearn. 2020-27-03. What Is Natural Language Understanding and How Does It Work? Geraadpleegd van https://monkeylearn.com/blog/natural-language-understanding/

Chang, C. 2017-31-01. Choosing your chatbot building platform: a guide for both bot newbies and experienced coders. Geraadpleegd van https://medium.com/@pdway53/decide-your-chatbot-builder-platforms-4067ac183add

Krishnan, K. 2020-01-01. 17 Chatbot Development Frameworks and Chatbot Platforms [2020]. Geraadpleegd van https://www.cedextech.com/blog/chatbot-development-frameworks

Shapiro, M. 2019-22-07. Build a chatbot for your mobile app. Geraadpleegd van https://developer.ibm.com/patterns/building-a-chatbot-with-kubernetes-watson-assistant-and-elastic-search/  
  
Mamgain, D. 2019-05-10-. Dialogflow vs Lex vs Watson vs Wit vs Azure Bot | Which Chatbot Service Platform To Use? Geraadpleegd van: https://www.kommunicate.io/blog/dialogflow-vs-lex-vs-watson-vs-wit-vs-azure-bot/  
  
Khatun, A. 2018-05-10. Something you need to know about Neural Network. Geraadpleegd van https://chatbotsmagazine.com/something-you-need-to-know-about-neural-network-2de9fd4e1ced  
  
Python virtual environments. (z.d.). 11-26-2014. Geraadpleegd van https://uoa-eresearch.github.io/eresearch-cookbook/recipe/2014/11/26/python-virtual-env/

Faber Novel. 2020-30-05. Chatbot testing in 2020 covering A/B, auto & manual testing. Geraadpleegd van https://research.aimultiple.com/chatbot-testing/  
   
  
Davydova, O. 2017-11-05. 25 Chatbot Platforms: A Comparative Table Data Monsters Data Monsters. Geraadpleegd van <https://chatbotsjournal.com/25-chatbot-platforms-a-comparative-table-aeefc932eaff>  
  
Azran, S. 2019-23-03. Google Dialogflow vs Microsoft LUIS vs IBM Watson Assistant — Conversational AI Comparison. Geraadpleegd van <https://medium.com/@samuel_22971/google-dialogflow-vs-microsoft-luis-vs-ibm-watson-assistant-conversational-ai-comparision-eb2d374f1413>  
  
**Boeken, Journals en Papers**Wang, S., Yu, M., Guo, X., Wang, Z., Klinger, T., Zhang, W., Chang, S., Tesauro, G., Zhou, B., Jiang, J. (2018). R3: Reinforced Ranker-Reader for Open-Domain Question Answering. Presented at the The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)

Castaño, A. P. (2018). *Practical Artificial Intelligence: Machine Learning, Bots, and Agent Solutions Using C#*. Apress.

Janarthanam, S., Venigalla, V. (2017) *Hands-On Chatbots and Conversational UI Development*.Packt Publishing.

Goodfellow, I., Bengio, Y., Courville, A. (2016). *Deep Learning*. Mit Press Ltd.

Shevat, A. (2017). *Designing Bots: Creating Conversational Experiences*. O'Reilly Media, Inc.

Akerkar, R. (2018). *Artificial Intelligence for Business* (1e editie). Springer  
  
Wu, W., & Yan, R. 2018. Deep Chit-Chat: Deep Learning for ChatBots.

Conference on empirical methods in natural language processing. Brussel, België.

Geraadpleegd van: <http://www.ruiyan.me/pubs/tutorial-emnlp18.pdf>  
  
Patil, A., Marimuthu, K., & Niranchana, R. (2017). Comparative study of cloud platforms to develop a Chatbot. International Journal of Engineering & Technology, 6(3), 57-61.

Shah, V., & Shah, (2019). A Comparison of Various Chatbot Frameworks. Cikitusi Journal for multidisciplinary research, Volume 6(Issue 4). Issn nr: 0975-6876 . Geraadpleegd van http://cikitusi.com

**Interview Cognitive Affairs**  
Interview met de CEO en CTO van Cognitive Affairs, Interview wat betreft chatbot behoeftes bedrijf Servoy/ mogelijke samenwerking met cognitive affairs, Amsterdam, 9 December. Samenvatting is te vinden in bijlage 3.

# **16. BegrippenLijst**

**Information Retrieval**Het ophalen van informatie (Information Retrieval), de wetenschappelijke / technische discipline achter zoekmachines  
  
**ODQA**ODQA is een open-domein vraagbeantwoording systeem dat antwoord geeft op basis van een documentverzameling. Open-domeinsystemen verwerken vragen over alle onderwerpen die met de verzameling te maken hebben.

**Intent**  
Het intent is de intentie van de gebruiker. De intentie van de gebruiker staat altijd in de zin die hij naar de chatbot stuurt. Als een gebruiker bijvoorbeeld 'toon me de trending horloges van deze maand' typt, is het de bedoeling van de gebruiker om een ​​lijst met horloges te die in de mode zijn op te halen. Intents krijgen een naam, vaak een werkwoord en een zelfstandig naamwoord, zoals "showTrending".

**Entity**  
Een entity is deel van een zin en dus deel van een intent. Als een gebruiker bijvoorbeeld 'toon me het financiële nieuws van gisteren' typt, staan er de entiteiten 'gisteren' als datum en 'financieel' als onderwerp. Entiteiten hebben categoriën, een voorbeeld zijn "dateTime" en "newsType".

**Examples**  
Examples of voorbeelden kunnen worden gebruikt door chatbots om een antwoord op te baseren, hoe groter het aantal examples des te groter gebied kan de chatbot beantwoorden.

**Counterexamples**Counterexamples (of tegenvoorbeelden) moeten echt dingen zijn die helemaal buiten de scope vallen, zoals 'wat voor weer is het' als je er bent om een ​​auto te kopen/verkopen, of 'welke aandelen moet ik kopen' als je er bent om iemand te leren koken.

**Confidence Score** - Is de zekerheid dat het antwoord wat een chatbot teruggeeft correct is. Deze liggen meestal in een waarde van 0 tot 1, oftewel tussen 0 en 100%.  
 **Deep Learning**Dit kan je zien als synoniemen die het systeem kan herkennen en weer kan terug kan koppelen aan het intent herkenning.  
 **NLP**  
NLP / natural language processing, is een subgebied van taalkunde, informatica, informatietechnologie en kunstmatige intelligentie dat zich bezighoudt met de interacties tussen computers en menselijke taal, met name hoe computers vallen te programmeren om grote hoeveelheden natuurlijke taalgegevens te verwerken en te analyseren.  
  
**NLC**Een vorm van NLP die in Watson gebruik wordt.

**Ranker**De ranker functionaliteit omvat het ophalen van informatie. De ranker kiest uit een verzameling van documenten de meest relevante voor een bepaalde vraag of query,

**Reader**Een reader extraheert een antwoord uit een gegeven tekst document, meestal op basis van een bepaald machine learning model. **Servoy**

Servoy is de naam van het bedrijf en ook de naam van platform wat ze gebruiken. Servoy is een uitbreiding op javascript, die garant staat voor low coding en waarmee relatief snel applicaties kunnen worden gebouwd.

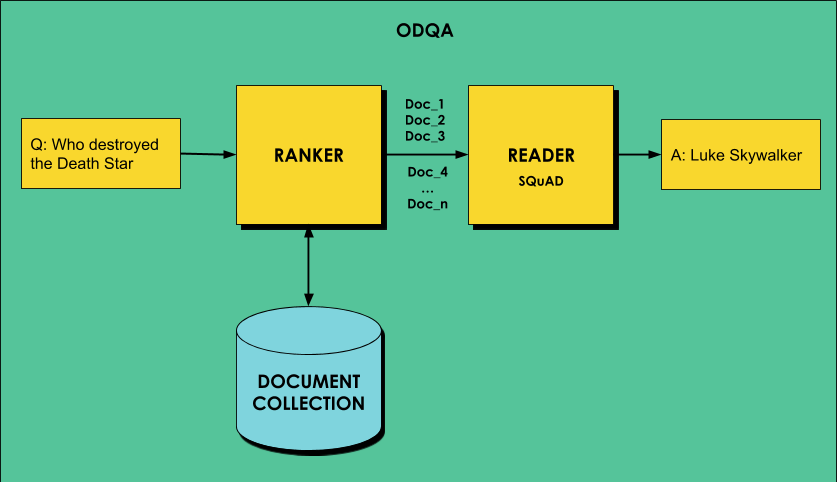
# **17. Bijlagen**

***Bijlage 1: De frequentie van het zoeken naar chatbot over time*** 

***Bijlage 2. Forum Website Technologie profiel.***

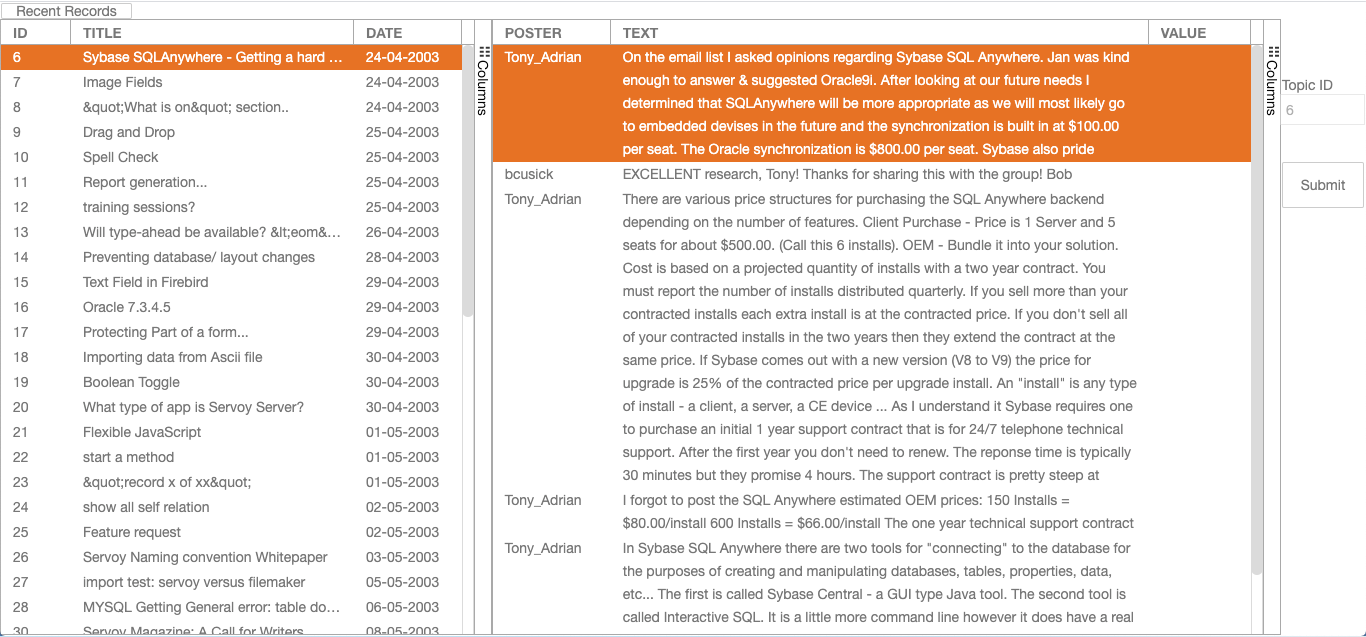
Na te hebben gesproken over de situatie met de mede oprichter en CEO van Cognitive Affairs Roland Hallebeek en de CTO Reiner Bruns, werden al snel een aantal dingen heel duidelijk. Volgens meneer Bruns moet je niet zomaar achteloos een chatbot naast je hoofd applicatie inzetten als je niet goed genoeg hebt nagedacht over de omgeving. Hij stipte aan dat het bijvoorbeeld belangrijk is om ook te zorgen dat de andere componenten waar de chatbot mee werkt optimaal bij de chatbot passen. Na het observeren van de huidige staat van het forum bij Servoy kwam meneer Bruns al snel met het advies om een betere zoekmachine te implementeren omdat de huidige niet optimaal functioneerde.   
  
Hiervoor waren een aantal reden, de prestatie van de huidige zoekmachine op het forum was aanzienlijk traag, het duurde namelijk ongeveer 2 seconden voordat het resultaat van de zoekopdracht geladen werd wat relatief erg langzaam is. Ook gaf de zoekmachine nauwelijks suggesties tijdens het zoeken, dit wordt vaak gedaan zodat de gebruiker alvast een idee heeft van wat er te vinden valt. Verder was het intent van de query’s ook nog niet duidelijk, in de zin dat de string die in de zoekmachine wordt ingetypt niet verder wordt geanalyseerd. Ook de dataset was niet duidelijk genoeg, in de zin dat je niet kan weten of er vragen en antwoorden zijn en welke dat zijn. Hierdoor zou een eventuele chatbot het moeilijk hebben om te bepalen waarmee hij antwoord moet geven.  
  
Verder vertelde de CTO ook dat er met een verbeterde zoekmachine een groot aantal extra opties komen die handig zijn voor het analyseren van de zoekstatistieken.Kortom, een groot aantal redenen en dus motivatie om van tevoren een betere zoekmachine te integreren.

***Bijlage 3. Een samenvatting van het interview***



***Bijlage 4 - Het DeepPavlov-gebaseerde ODQA Systeem architectuur***



***Bijlage 5 - Zekerheid van de methodes die het deeppavlov model gebruikt uitgedrukt in Leave-One-Out Cross-Validation (LOOCV). De dichter bij 1 de beter.***

**Bijlage 7 - Javascript code van de curatie applicatie**

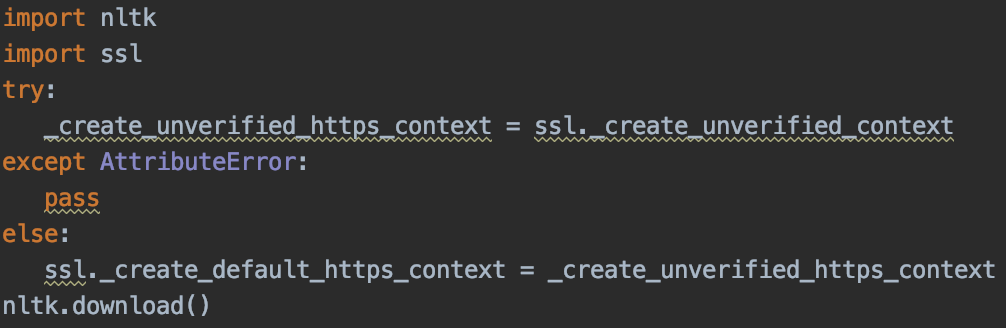


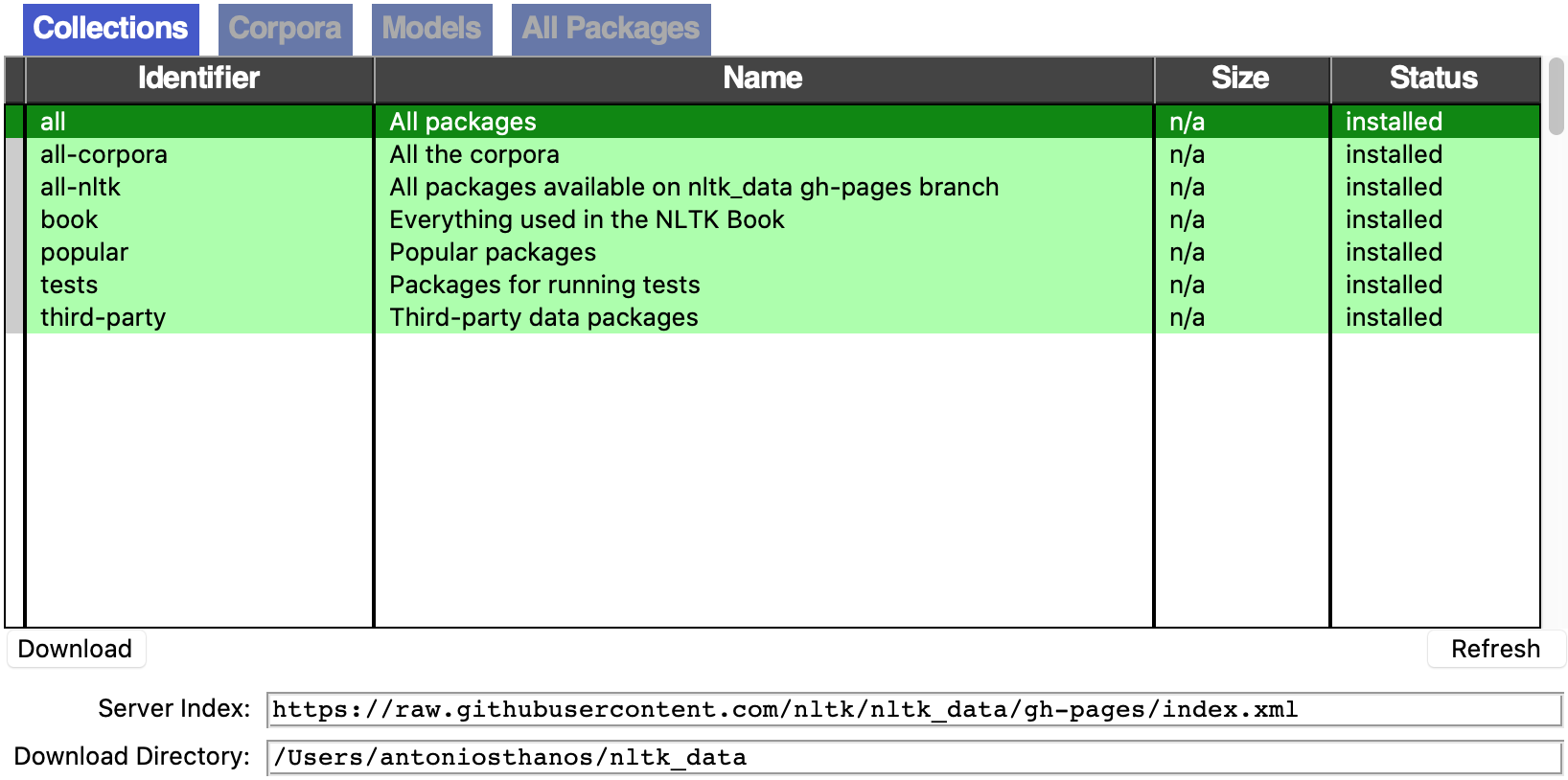
***Bijlage 8 -***

***Ranker file***

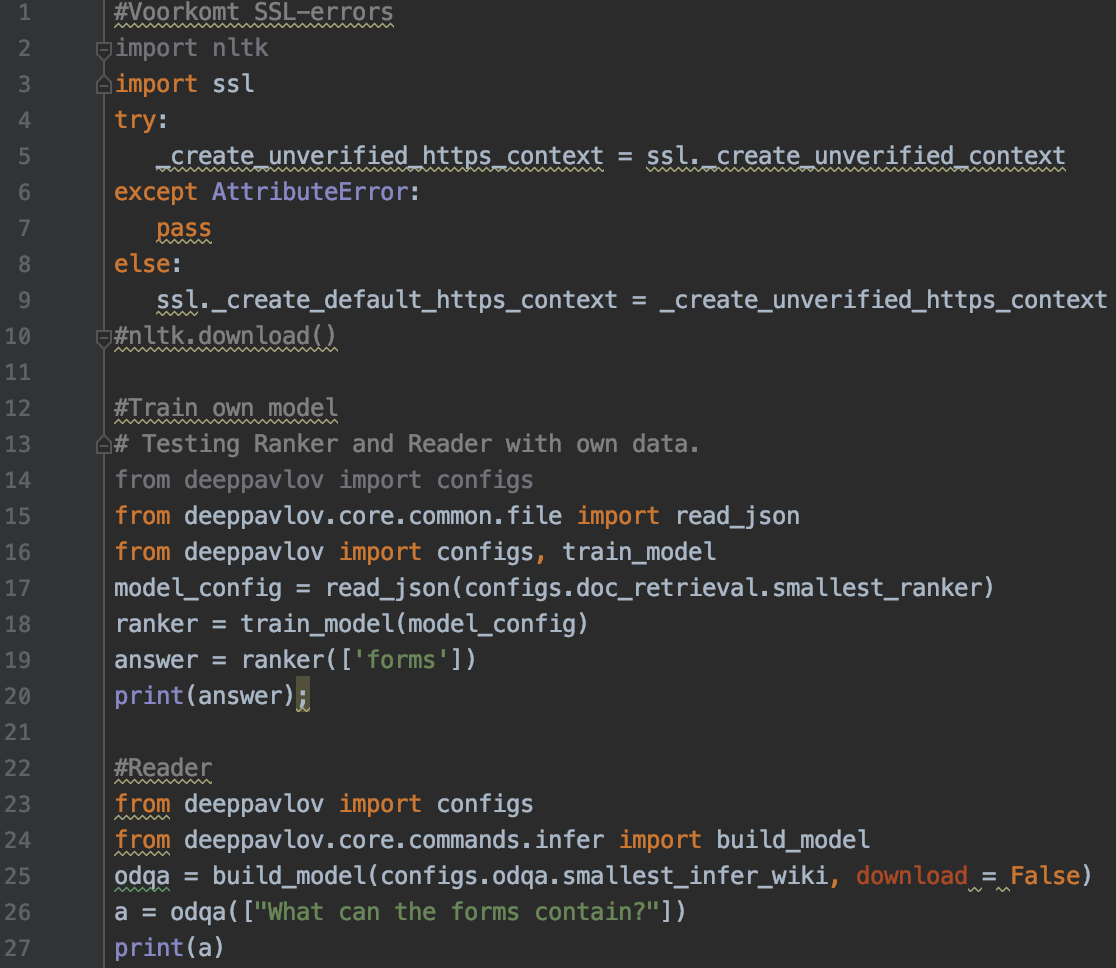
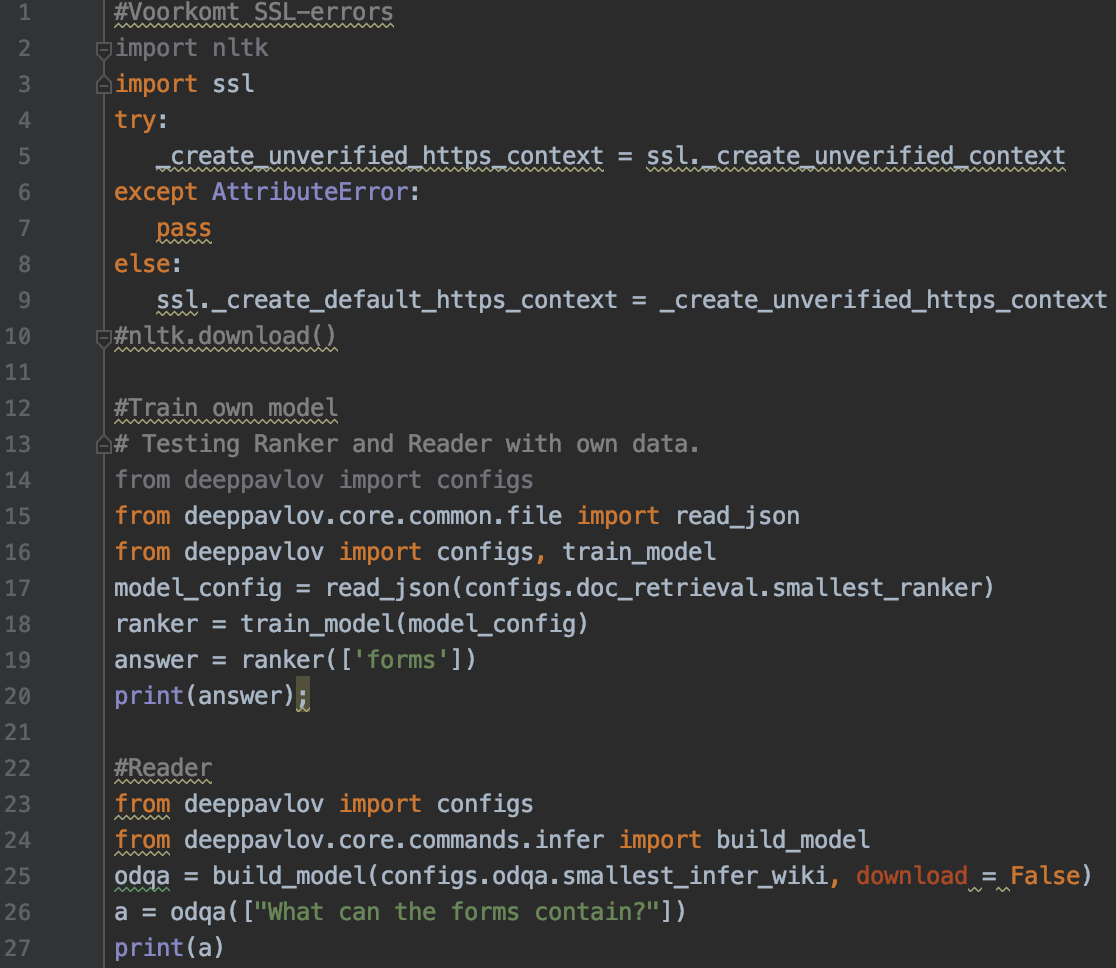


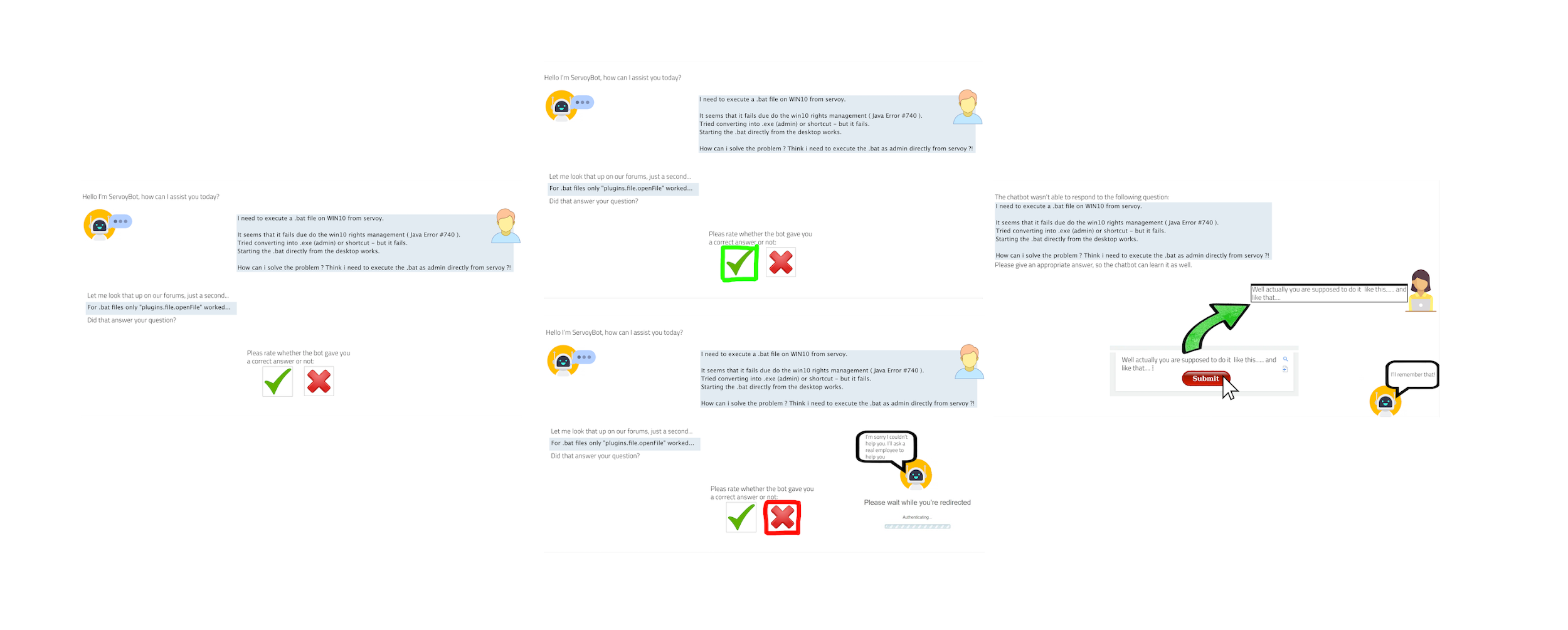
***Bijlage 9 - Reader file***

**Bijlage 10 - Code om de NLTK Downloader aan te roepen.**



**Bijlage 11 - NLTK downloader**

**Bijlage 12 - Python code implementatie Deeppavlov**



**Bijlage 13 - Chatbot Flow**

[ [

"they are calculated based on a set of values in a column",

[

0.0006284322165234036,

0.0008247691109322185,

0.18066502146594768,

0.779029682684125,

0.0008261025868301818,

0.0002867979042852699,  
 0.0021040487334457896,

0.0005403416967113814,

0.000977419526367211,

0.0003508835428068086

]]]

**Bijlage 14 - Output REST API met DeepPavlov**

****

**Bijlage 15 - Servoy Bot code met DeepPavlov REST**

****

**Bijlage 16 - CSS opmaak van chatbot  
  
1 Algemeen testen (controlegroep)**

| **Testgeval 1:** | |
| --- | --- |
| **Input** | “Where is the script editor located?” |
| **Resultaat** | “It appears in same place as Form Designer by default” |
| **Uitslag** | Correct |

| **Testgeval 2:** | |
| --- | --- |
| **Input** | “Can you help me?” |
| **Resultaat** | “Of course! If you could just write down your question I would be glad to help!” |
| **Uitslag** | Correct |

| **Testgeval 3:** | |
| --- | --- |
| **Input** | “How are aggregations calculated?” |
| **Resultaat** | “They are calculated based on a set of values in a column.” |
| **Uitslag** | Correct |

**2. Domeinspecifiek testen  
  
Eigenschap 1 Synoniemen**

| **Testgeval 4: Gebruik van synoniem (computed i.p.v. calculated)** | |
| --- | --- |
| **Input** | ”How are aggregations computed?” |
| **Verwacht** | “They are calculated based on a set of values in a column.” |
| **Resultaat** | “They are calculated based on a set of values in a column.” |
| **Uitslag** | Correct |

| **Testgeval 5: Gebruik van synoniem (characteristics i.p.v. features)** | |
| --- | --- |
| **Input** | ”What are some characteristics of a relation?” |
| **Verwacht** | “1. The tables can be linked by equality, but also by other operators (<,>,like, ... |
| **Resultaat** | “1. The tables can be linked by equality, but also by other operators (<,>,like, ... |
| **Uitslag** | Correct |

| **Testgeval 6: Gebruik van synoniem (association i.p.v. aggregation)** | |
| --- | --- |
| **Input** | ”"What is an association?"” |
| **Verwacht** | “An aggregation is a value derived from a set of records in a table.” |
| **Resultaat** | “An aggregation is a value derived from a set of records in a table.” |
| **Uitslag** | Correct |

**Eigenschap 2 Semantisch Alternatief**

| **Testgeval 7: Alternatief gestelde vraag** | |
| --- | --- |
| **Input** | “Where is the script editor view?” |
| **Verwacht** | “It appears in same place as Form Designer by default” |
| **Resultaat** | “It appears in same place as Form Designer by default” |
| **Uitslag** | Correct |

| **Testgeval 8: Compleet andere zin met dezelfde bedoeling** | |
| --- | --- |
| **Input** | ”I want to know about the way aggregations are calculated.” |
| **Verwacht** | “They are calculated based on a set of values in a column.” |
| **Resultaat** | “They are calculated based on a set of values in a column.” |
| **Uitslag** | Correct |

| **Testgeval 9: Gebruik van verkeerde zinsbouw** | |
| --- | --- |
| **Input** | ”aggregations are computed how exactly?” |
| **Verwacht** | “They are calculated based on a set of values in a column.” |
| **Resultaat** | “They are calculated based on a set of values in a column.” |
| **Uitslag** | Correct |

**3 Limiet testen  
  
Eigenschap 1: Aantal woorden**

| **Testgeval 10: 5 woorden extra in een originele vraag van 4 (what is an aggregation?)** | |
| --- | --- |
| **Input** | “What is a person that eats steak drinks water and buys aggregations?" |
| **Verwacht** | “An aggregation is a value derived from a set of records in a table.” |
| **Resultaat** | “An aggregation is a value derived from a set of records in a table.” |
| **Uitslag** | Correct |

| **Testgeval 11: 6 extra woordenaantal in een originele vraag van 4 (what is an aggregation?)** | |
| --- | --- |
| **Input** | ”Who is a person that eats steak and buys aggregations?" |
| **Verwacht** | “An aggregation is a value derived from a set of records in a table.” |
| **Resultaat** | “A Servoy Developer” |
| **Uitslag** | Failed |

**Eigenschap 2: twee vragen in één zin**

| **Testgeval 12: Gebruik van verkeerde zinsbouw** | |
| --- | --- |
| **Input** | ”What is an aggregation and whoever is Jan Blok” |
| **Verwacht** | “An aggregation is a value derived from a set of records in a table.” |
| **Resultaat** | “An aggregation is a value derived from a set of records in a table.” |
| **Uitslag** | Correct |

| **Testgeval 13: What is an aggregation met Een extra Q&A** | |
| --- | --- |
| **Input** | ”What is an aggregation and who is Jan Blok” |
| **Verwacht** | An aggregation is a value derived from a set of records in a table. |
| **Resultaat** | “A servoy developer” |
| **Uitslag** | Failed |

**Bijlage 17- Testgevallen DeepPavlov**