HOGESCHOOL PXL

Een Snellere Datastroom:

WebSockets Onder de Loep

Orens Jasper
Indestege Michelle
Lenaerts Roald
Level27, Hasselt

Woord vooraf

Voor u ligt de graduaatsproef "*Een snellere datastroom: websockets onder de loep*" Dit project is opgesteld om te voldoen aan de afstudeereisen van de opleiding *Programmeren* aan de ***PXL*** in Hasselt. Ik ben sinds februari begonnen aan het onderzoek met volle interesse.

Ik merkte op tijdens de studie, dat er enkele dingen wat moeilijker haalbaar zouden zijn. Ik had bijvoorbeeld opgegeven om het cashing mechanisme van websockets in mijn onderzoek op te nemen. Echter ondervond ik later dat WebSockets dit niet implementeren. Ik heb relatief veel ervaring opgedaan aan de studie en heb hier zeker en vast de meerwaarde van in gezien. Ik heb mijn stormen moeten trotseren om mijn project tot een goed einde te brengen. Maar al het harde werk tot laat in de nacht, heeft u deze scriptie gebracht.

Ik wil mijn PXL-coach Michelle Indestege bedanken voor de geweldige opvolging die ze heeft gedaan voor mij. Ik had u niet zelf gekozen als begeleider, maar ik ben zeker en vast blij dat u mij toegewezen is geweest. Mijn grootste dank gaat naar Roald Lenaerts mijn WPL-coach. Roald heeft tijdens het werkplek leren veel tijd gestoken in mij alles zo goed mogelijk aan te leren. En als hij iets niet wist, dan onderzocht hij dit zelf en verklaarde hij dit nadien aan mij.

Ik wil ook nog mijn dankwoord geven aan Juan Jacobs en Joran Dirkzwager van het platform team bij Level27. Het is mij een genoegen geweest om met zulke top programmeurs samen te werken. Ik heb zeer veel gehad van jullie hulp en kennis tijdens mijn traject bij Level27. De sfeer was er altijd aangenaam en jullie lieten mij er heel welkom voelen.

Tot slot wil ik dit project ook opdragen aan Erwin Peters, mijn persoonlijke held, mijn stiefvader, en mijn beste vriend. Voor de aanvang van mijn opleiding is hij helaas heen gegaan. Maar Erwin heeft mij de motivatie gegeven om altijd het beste overal van proberen te maken. En als iets niet meezat in mijn leven, dat ik dan moest zorgen voor de verandering. Dit heb ik toen ook gedaan door de opleiding te starten bij PXL. Dus bij deze mijn dank.

Ik hoop dat deze graduaatsproef u goed bevalt en ik wens u veel leesplezier toe.

Jasper Orens,

13 mei 2024, 3500 te Hasselt.

Inhoudsopgave

Inhoud

1 Bedrijfsvoorstelling

1.1 Situering van het bedrijf

Level27 is gevestigd in Hasselt, België en opereert in de wereld van Hosting op internationaal niveau. Het bedrijf werd opgericht in 2007. De afdeling waarin de graduaatsproef situeert is *Platform*. Deze afdeling is verantwoordelijk voor de interface en datastroom waar de klant constant mee in aanmerking komt. Zij zorgen voor verbeteringen aan het platform op de *Application Layer* van het *TCP/IP.*

1.2 Werkomgeving

Level27 is gespecialiseerd in diverse onlineactiviteiten. Deze variëren van webhosting, agency hosting, Managed services en Cloud-services. Het gebruik van een bekende methodologie wordt niet toegepast op het bedrijf. Echter wordt er iedere week een zelfreflectie ingevuld in een template. Deze zelfreflectie wordt ook wekelijks besproken met het afdelingshoofd. Iedere vrijdag presenteert één van de werknemers van Level27 een pitch voor om de loop van deze werknemer hun afdeling te tonen. Over het algemeen is het een zeer rustgevende werkplaats waar veel aandacht word besteed aan een goede sfeer tussen de collega's.

2 Projectvraag, onderzoeksacties en resultaten

2.1 Situering probleemstelling

De huidige websocket implementatie bij Level27 werkt niet volledig naar behoeven. Er moet een andere implementatie van WebSockets worden geprogrammeerd om data snelle van server naar client te versturen. Er is geen noodzaak om de implementatie te verbeteren van client naar server.

2.2 Projectvraag en deelvragen

- Wat zijn de meest voorkomende Web Socket implementaties in de werled om data te tonen van de server?
- Zijn er op dit moment alternatieven van WebSockets die eventueel beter kunnen werken?
- Hoe beïnvloed de keuze van het gekozen framework voor de WebSocket de prestaties van Full-Duplex communictie en welke framework is het meest geschikt voor optimalisatie tegen eind mei 2024?
- Is de nieuwe implementatie een verbetering op de huidige WebSocket die gebruikt wordt bij Level27?

2.3 (Onderzoeks-)acties

2.3.1 Wat zijn de meest voorkomende Web Socket implementaties in de wereld om data te tonen van de server?

Om beter begrip te verkrijgen van de werking van WebSockets heb ik drie diverse cursussen gedaan die dieper in gaan op WebSockets. Er is een Node.Js server/applicatie opgesteld om de noodzakelijke data mee teversturen voor alle socket implementaties en libraries op te laten werken. [8]. In de eerste fase is de node.js server zelf opgesteld en als tweede stap is de database hierin opgemaakt. De mockata in de databank maakt gebruik van de faker library [9]. Deze genereerd valse data die realistisch lijkt.

2.3.2 Zijn er op dit moment alternatieven van WebSockets die eventueel beter kunnen werken?

Om beter begrip

2.3.3 Hoe beïnvloed de keuze van het gekozen framework voor de WebSocket de prestaties van Full-Duplex communicatie en welke framework is het meest geschikt voor optimalisatie tegen eind mei 2024?

Om beter begrip

2.3.4 Is de nieuwe implementatie een verbetering op de huidige WebSocket die gebruikt wordt bij Level27?

Om beter begrip

2.4 Verzamelde resultaten

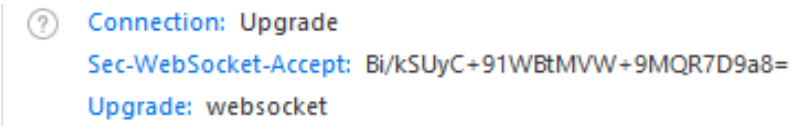
2.4.1 Web Socket

Algemene werking van WebSockets

Web Sockets maken gebruik van WS of WSS(secure vorm) in hun HTTP requests. De connectie blijft open tot de client een disconnect request verstuurd.

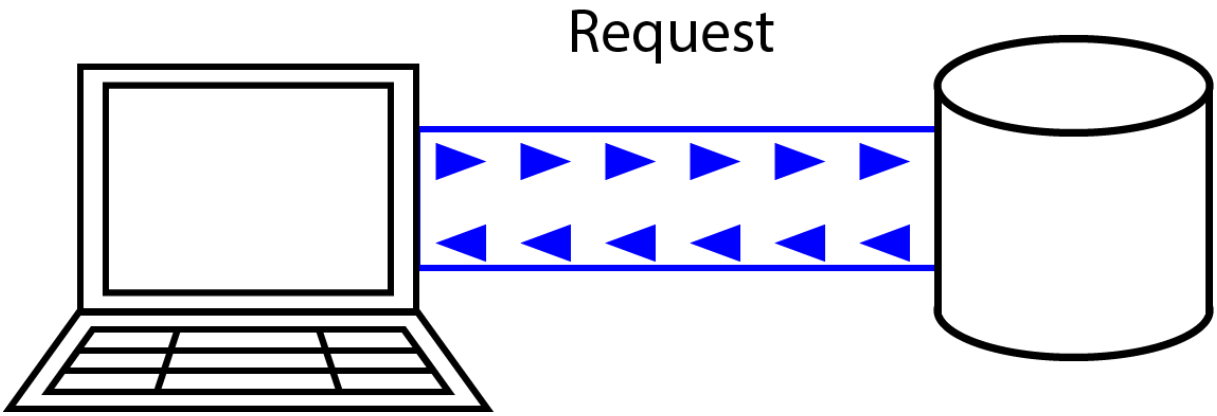
HandShake

De client verstuurd een Request met ‘upgrade’ als onderdeel van de header. Deze heeft als waarde ‘WebSocket’ en een veld ‘connection’ met als waarde ‘Upgrade’.



Screenshot van console in Firefox van een WebSocket implementatie

De client verstuurd dan een BASE64 encoded Key. Wanneer de Server dan een reactie geeft. Als deze bekwaam is voor WebSockets te aanvaarden. Gebruikt de Server het HTTP 101 response. (Switching Protocols). Nadien verstuurd de server een header met dezelfde waardes als de client initieel verstuurde. Namelijk ‘Upgrade: websocket’ en ‘Connection: Upgrade’. De server neemt dan de Key van de client, voegt een GUID hier aan toe en berekend hier een SHA-Hash op. Op deze manier weet de client dat de server het verzoek heeft aanvaard. Dit een noodzakelijke authenticatie voor ‘Man in the Middle’ attacks te voorkomen, wanneer een hacker zich kan voordoen als de server. Als deze acties uitgevoerd zijn is er Bidirectionele communicatie tussen de Server en Client.



Afbeelding door Jasper Orens

Use case

Chat applicatie:

Facebook Messenger maakt bijvoorbeeld gebruik van WebSockets om constant data te verzenden van client 1 naar server en naar client 2.

Social Media updates:

Push notificaties kunnen van social media kunnen gebruikt worden voor de gebruiker berichten te laten ontvangen wanneer er updates binnen komen (bv.: “Iemand heeft je getagged in een foto”, van Facebook).

Gok websites:

Om scores aan te tonen van Live wedstrijden moeten deze altijd gelijktijdig naar alle gebruikers worden ge-update. WebSockets zijn hier zeer geschikt voor.

IoT-toepassingen:

WebSockets kunnen worden gebruikt om communicatie met Internet of Things (IoT)-apparaten mogelijk te maken, waardoor apparaten zoals sensoren en slimme lampen real-time gegevens naar een server kunnen sturen.

2.4.2 Socket.IO

Wat is het?

Socket.IO is een library dat low-latency, bidirectioneel en event-based communicatie tussen client en server voorziet. De connectie kan opgesteld worden tussen verschillende low-level transports. Deze zijn:

- HTTP long-polling
- WebSocket
- WebTransport

Socket.IO kiest zelf het beste pad afhankelijk van

- De mogelijkheden van de browser
- Het netwerk (sommige netwerken blokeren WebSockets en/of WebTransport connecites

Deze library is geen implementatie van een WebSocket. Deze gebruikt wel WebSockets voor transport wanneer mogelijk, maar het voegt extra metadata toe in elk packet. Hierdoor kan een traditionele WebSocket geen connectie maken met Socket.IO omdat de WebSocket geen weg weet met deze extra data van Socket.IO. [7].

Gebruiksgemak en documentatie

De initiële setup voor socket.IO is relatief gemakkelijk. De basis setup is getest dankzij de duidelijke documentatie [10] Waarin het ook mogelijk is om de library te benutten via een virtual machine in de browser. De documentatie is zeer gebruiksvriendelijk, en éénmaal als de initiële setup afgerond is. Is het algemene gebruik overzichtelijk.

Performance en schaalbaarheid

De performance kan niet enkel verschillen met hoe de data verstuurd wordt, maar ook de browser heeft invloed op de totale performance. Over het algemeen verbruikt Firefox minder CPU dan Chrome. Dus wij zullen de performance hiermee testen.

Ondersteuning en community

Er is buiten de zeer goede documentatie weinig ondersteuning te vinden op hun platform. Er is echter wel een doorverwijzing naar stackoverflow.

Licentie en kosten

socket.IO is gratis en wordt verdeeld onder het MIT-licensie. [11]

Compatibiliteit en integratiemogelijkheden
Socket.IO is zeer compatibel en kan geïntegreerd worden met veel programmeertalen en platforms, waaronder JavaScript, Java, Python, en C++. Het ondersteunt zowel webbrowsers als mobiele apps en biedt functies zoals automatische reconnection en event broadcasting. Dit maakt het geschikt voor real-time webapplicaties zoals chats en interactieve collaboratieve tools. Voor meer technische details. [7, 11]

Voordelen
- Automatische fallback naar HTTP long-polling: Waar gewone WebSocket-implementaties afhankelijk zijn van de permanente beschikbaarheid van een WebSocket-verbinding, kan Socket.IO automatisch terugvallen op HTTP long-polling als WebSockets niet beschikbaar zijn.

- Eenvoudige API voor complexe functionaliteiten: Socket.IO vereenvoudigt de implementaties van complexe functies zoals broadcasting naar meerdere sockets en het afhandelen van reconnecties na verbindingsverlies, wat meer codering en configuratie zou vereisen bij direct gebruik van WebSockets.

Nadelen
- Overhead: Socket.IO voegt extra bytes toe aan elk bericht voor het beheren van zijn functionaliteiten zoals namespaces en rooms, wat resulteert in grotere bericht groottes vergeleken met een zuivere WebSocket-oplossing.

- Complexiteit: Voor projecten waarbij eenvoudige berichtuitwisseling voldoende is, kan Soikcet.IO overkill zijn vanwege de extra functionaliteiten en de ingebouwde ondersteuningsmechanismen die misschien niet nodig zijn.

2.4.3 WebSocket API (Native)
Wat is het?
text

Var socket = new WebSocket(url); socket.onopen = () => {}; socket.onmessage = (e) => {}; socket.onclose = (e) => {} socket.onerror = (e) => {}	- We maken een connectie met de server - De connectie is open - Wij ontvangen een bericht - De connectie is gesloten - Fout afhandeling

[1].

Gebruiksgemak en documentatie

Performance en schaalbaarheid

Ondersteuning en community

Licentie en kosten

Compatibiliteit en integratiemogelijkheden
text

Voordelen
tekst

Nadelen
text

2.4.4 Pusher

Wat is het?
text

Gebruiksgemak en documentatie
text

Ondersteunende functies en mogelijkheden
text

Performance en schaalbaarheid
text

Ondersteuning en community
text

Licentie en kosten
text

Compatibiliteit en integratiemogelijkheden
text

Voordelen en Nadelen
text

2.4.5 RxJS WebSocket

Wat is het?
text

Gebruiksgemak en documentatie
text

Ondersteunende functies en mogelijkheden
text

Performance en schaalbaarheid
text

Ondersteuning en community
text

Licentie en kosten
text

Compatibiliteit en integratiemogelijkheden
text

Voordelen en Nadelen
text

2.4.6 GraphQL Subscriptions

Wat is het?
text

Gebruiksgemak en documentatie
text

Ondersteunende functies en mogelijkheden
text

Performance en schaalbaarheid
text

Ondersteuning en community
text

Licentie en kosten
text

Compatibiliteit en integratiemogelijkheden
text

Voordelen en Nadelen
text

2.4.7 SockJS

Wat is het?
text

Gebruiksgemak en documentatie
text

Ondersteunende functies en mogelijkheden
text

Performance en schaalbaarheid
text

Ondersteuning en community
text

Licentie en kosten
text

Compatibiliteit en integratiemogelijkheden
text

Voordelen en Nadelen
text

2.4.8 WS / WSS

Wat is het?
text

Gebruiksgemak en documentatie
text

Ondersteunende functies en mogelijkheden
text

Performance en schaalbaarheid
text

Ondersteuning en community
text

Licentie en kosten
text

Compatibiliteit en integratiemogelijkheden
text

Voordelen en Nadelen
text

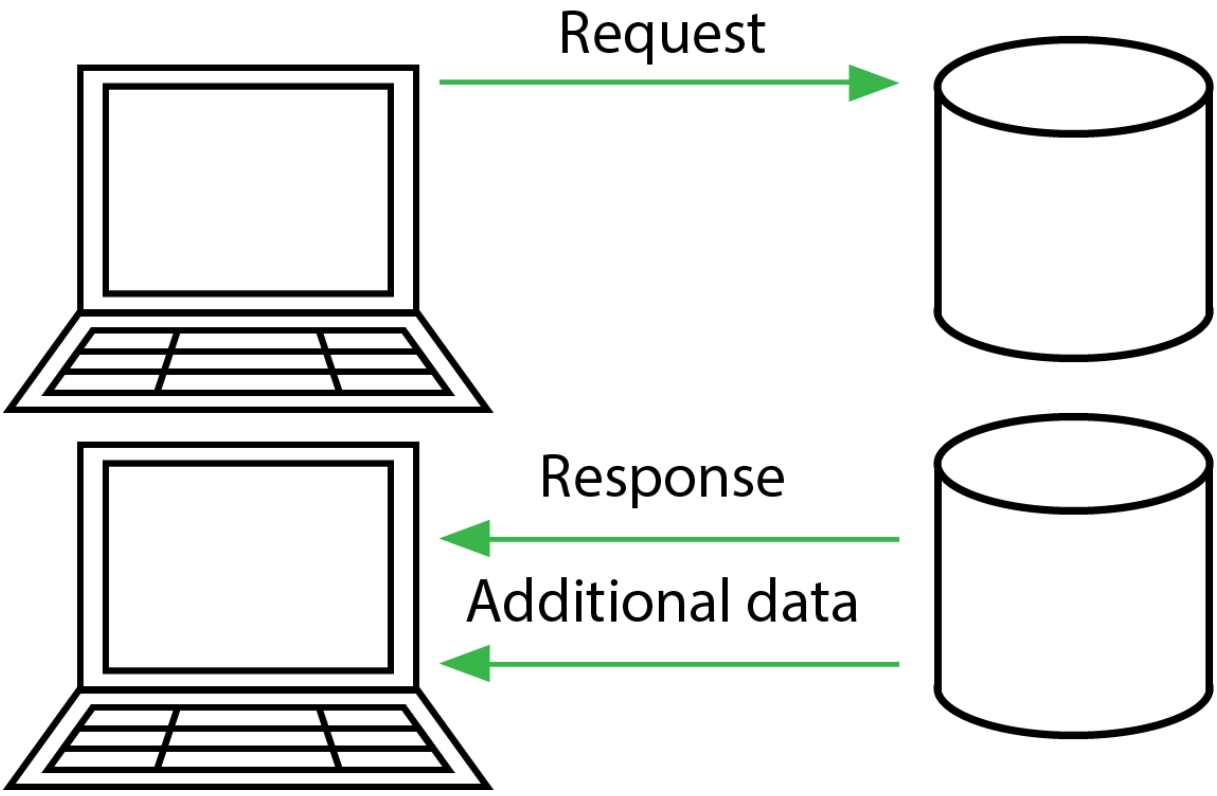
2.4.9 Alternatieven voor WebSockets

2.4.9.1 HTTP/2 Push

Er werd onderzocht naar HTTP/2 Push. Maar Google heeft besloten om deze functie te verwijderen. Verdere implementatie van deze functie is dus niet uitgewerkt binnen het project. Google heeft het aantal requests op Chrome Browsers bestudeerd dat push requests verzonden en ontvangen. Uit hun onderzoek bleek dat 1.25% van HTTP/2 websites gebruik maakten van deze functie. Bij verdere opvolging bleek dat dit aantal gezakt was naar 0.7%. Op basis van deze analyse besloot Google om deze functie niet verder uit te werken. [2]

2.4.9.1.1 Werking van HTTP2/Push

De client verstuurt een verzoek voor een bepaalde webpagina of bron, zoals HTML, De Server ontvangt het verzoek van de client en identificeert alle vereiste bronnen die nodig zijn om de gevraagde webpagina correct te renderen. In plaats van te wachten tot de client afzonderlijke verzoeken voor elke bron verzendt, kan de server proactief beslissen om bepaalde bronnen naar de client te duwen. Dit wordt gedaan door extra bronnen te ‘pushen’ naar de client over dezelfde verbinding zonder te wachten op een afzonderlijke verzoek van de client. Zodra de client de gepushte bronnen ontvang, kan hij deze lokaal opslaan in de cache. Wanneer de client later die bron nodig heeft, hoeft hij geen nieuwe verzoek naar de server te sturen, omdat de bron al beschikbaar is in de cache van de client. [3]

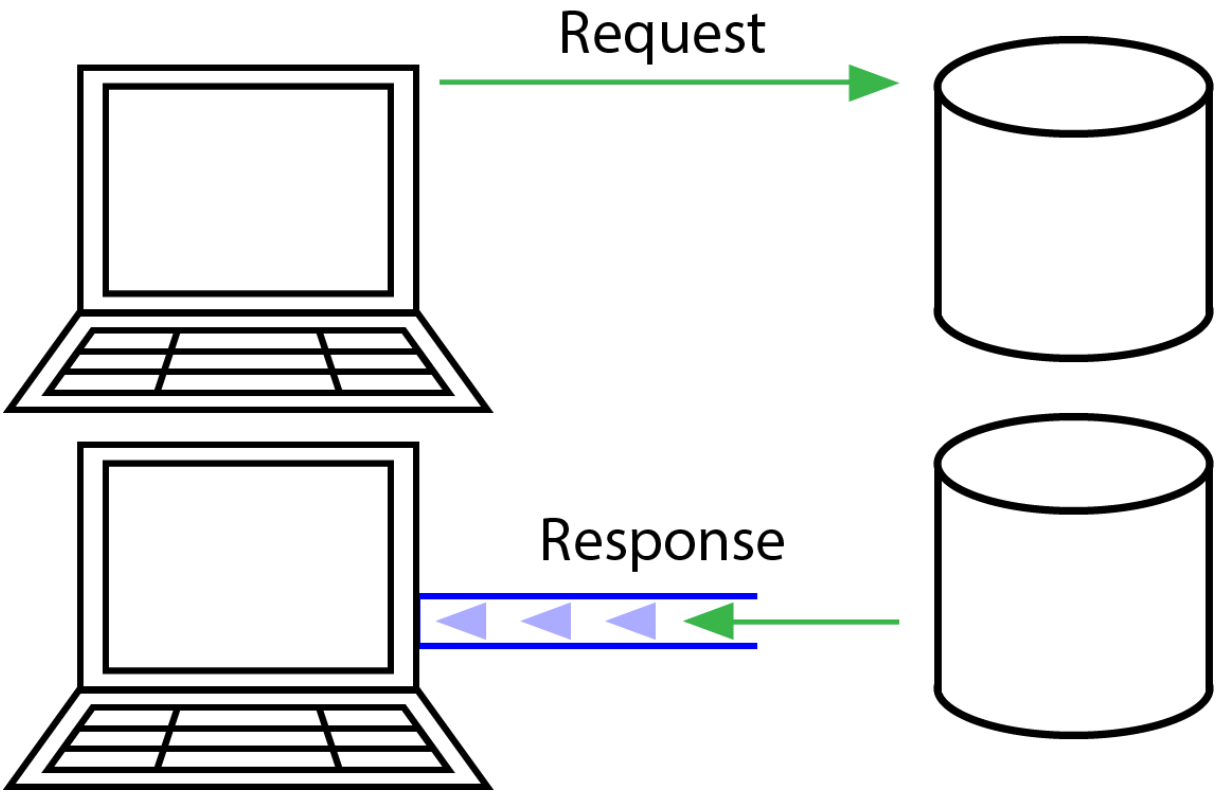


Afbeelding door Jasper Orens

2.4.9.2 Long Polling

2.4.9.2.1 Werking van Long Polling

Deze methode stuurt een verzoek naar de resource, als er geen antwoord komt op het verzoek van de client dan blijft de connectie met de resource open tot er wijzigingen gebeuren en data wordt overgemaakt van de resource naar de client. Via deze methode worden het aantal requests naar de server dus beperkt omdat de connectie langer openblijft en de client dus niet voortdurend requests moet sturen tot er een positief antwoord komt. Long Polling maakt gebruik van HTTP request in de formaten applicatoin/json, tekst/plain en de open connectie blijft 100 tot 300 seconden geldig.



Afbeelding door Jasper Orens

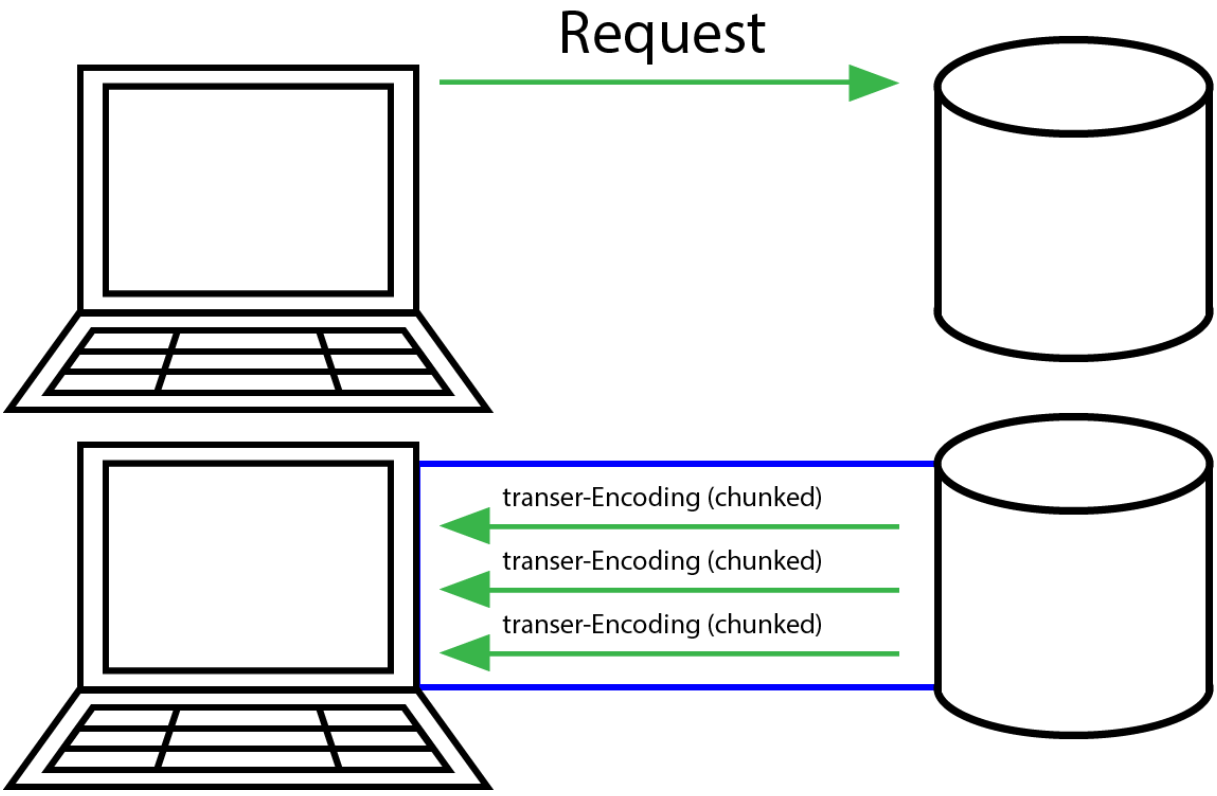
2.4.9.2.2 Use case

De beste benutting voor Long polling is voor chat applicaties waarin realtime updates noodzakelijk zijn. Specifiek chat communicatie tussen twee partijen. (Voor meerdere partijen zijn websockets de betere keuze.) Dit omdat het de connectie met de server openhoudt tot er nieuwe data (een bericht bv.) beschikbaar is. Zodra er nieuwe data aanwezig is op de server verstuurd deze de inhoud naar de client. De long polling open connectie wordt dan gesloten tot er van uit de client een nieuw polling request wordt verstuurd. [4]

2.4.9.3 Server-Sent Events (SSE)

2.4.9.3.1 Werking van SSE

Bij een SSE request wordt de aanvraag gedaan via een ‘event-stream’ in plaats van een .JSON-formaat of tekstformaat. Via deze weg weet de server dat er gewerkt wordt met een SSE-connectie. De server geeft als volgt meerdere antwoorden in de vorm van ‘event-stream’ in chunks. Dit zijn grootte blokken code die werden opgesplitst. Er wordt één request gestuurd voor een connectie te openen en de server beslist nadien welke data het wilt sturen naar de client. SSE maakt gebruik van HTTP requests enkel i het tekst/event-stream formaat. Data wordt verstuurd als Prefix. De connectie blijft open tot de client een disconnect request verstuurd.



Afbeelding door Jasper Orens

2.4.9.3.2 Use case

Server-Sent Events (SSE) worden gebruikt voor het live streamen van gegevens, vergelijkbaar met platforms zoals YouTube of Twitch voor video streams. Deze technologie maakt het mogelijk om real-time gegevens te verzenden van de server naar de client, waardoor een continue stroom van informatie wordt gecreëerd zonder dat de client herhaaldelijk om updates hoeft te vragen. SSE is vooral nuttig voor applicaties die afhankelijk zijn van live data, zoals sportupdates, financiële marktgegevens en meer. Het bijzondere van SSE is dat het een bidirectioneel communicatiekanaal biedt, hoewel het geen authentieke full-duplex communicatie is. In plaats daarvan stelt SSE servers in staat om gegevens in realtime naar clients te sturen, terwijl clients geen mogelijkheid hebben om direct naar de server te communiceren via hetzelfde SSE-kanaal. Deze eenrichtingsstroom van gegevens maakt het mogelijk voor clients om continu bijgewerkte informatie te ontvangen zonder de noodzaak van polling, waardoor de serverbelasting wordt verminderd en de efficiëntie van de communicatie wordt verbeterd.[5]

2.4.9.4 WebRTC (Web Real-Time Communication)

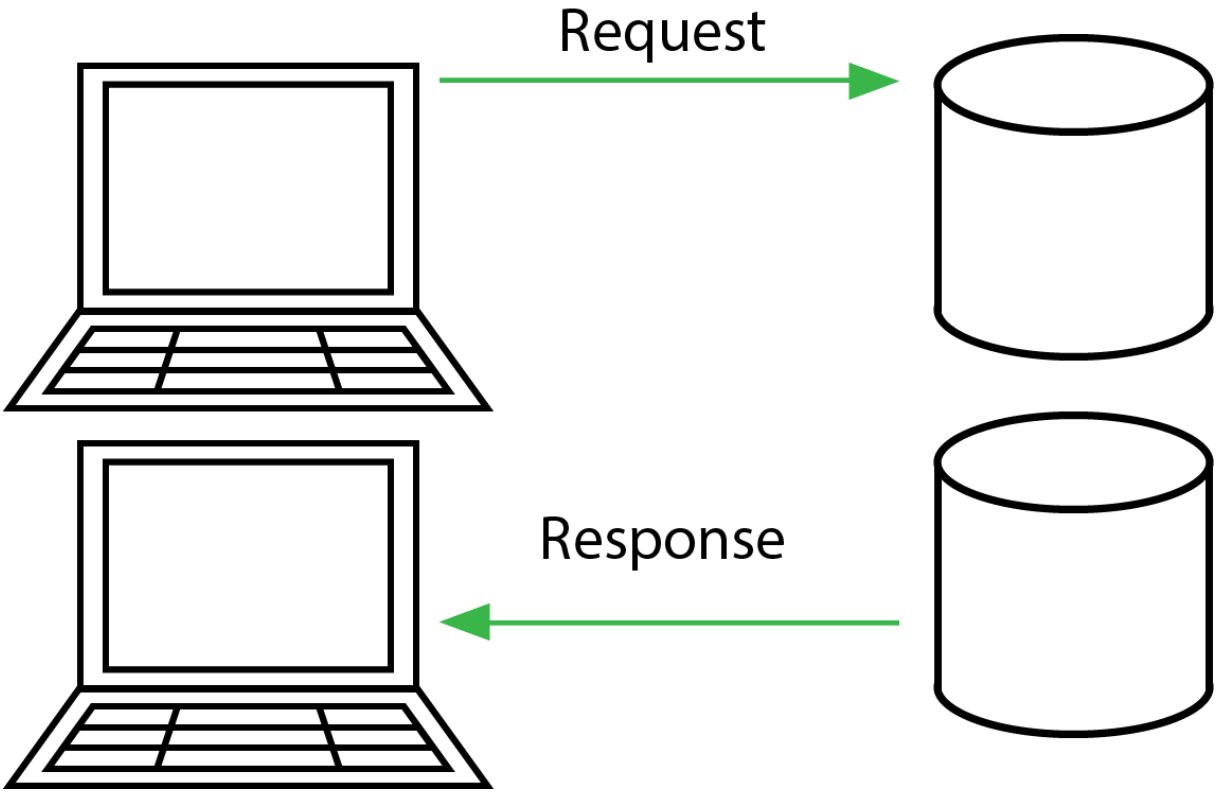
2.4.9.4.1 Werking van WebRTC

2.4.9.4.2 Use case

2.4.9.5 Short Polling

2.4.9.5.1 Werking van API

2.4.9.5.2 Use case



Afbeelding door Jasper Orens

3 Conclusies en aanbevelingen

In dit gedeelte geef je je conclusies die je vanuit je uitgevoerde onderzoek kan afleiden, weer. Je beantwoordt hier je onderzoeksvraag: je geeft op een synthetische en goed beargumenteerde manier een antwoord op je onderzoeksvraag en staat stil bij de belangrijkste vaststellingen van je onderzoek. Je geeft tips en aanbevelingen voor je werkplek naar voor schuiven die je vanuit je onderzoek aangereikt krijgt/gekregen hebt. Je kadert de meerwaarde hiervan.

3.1 Conclusies

Tekst tekst tekst

3.2 Aanbevelingen

Tekst tekst tekst

4 Persoonlijke reflecties en kritische kanttekeningen

*Je noteert hier hoe jij het werken aan je Graduaatsproef ervaren hebt a.d.h.v. het STARR reflectiemodel. Belangrijke persoonlijke leerinzichten en kritische kanttekeningen die je meeneemt vanuit het doorgemaakte proces kunnen hier hun plaats krijgen. Je beschrijft wat voor jou persoonlijk de meerwaarde van de uitwerking van je Graduaatsproef is; wat kan/wil je **voor jezelf** vanuit het doorlopen van je onderzoek als **leerinzichten** naar voor schuiven, koppel deze leerinzichten naar de **OLR** en **X-factor**.*

Daarnaast zijn ook jouw persoonlijke kritische kanttekeningen belangrijk om te vermelden: waar zit voor jou de ***meerwaarde en eventuele beperkingen van je onderzoek zoals jij dit doorlopen hebt****? Belangrijk hierbij is dat je in deze reflecties de link legt met hetgeen bovenstaand werd neergeschreven en dat je in je reflecties dus verwijst naar concrete informatie/acties … vanuit je Graduaatsproef.*

Tekst tekst tekst

5 Referentielijst

De referenties die je raadpleegde en die belangrijk zijn in de uitwerking van je Graduaatsproef dienen hier vermeld te worden. Je noteert dit op een consequente en correcte manier.

[1] ***Cursus network requests in JavaScript***: Network Requests in JavaScript by Christian Wenz (https://app.pluralsight.com/library/courses/javascript-network-requests/table-of-contents)

[2] https://developer.chrome.com/blog/removing-push

[3] https://www.ioriver.io/terms/http-2-server-push

[4] https://www.pubnub.com/guides/long-polling/

[5] https://medium.com/deliveryherotechhub/what-is-server-sent-events-sse-and-how-to-implement-it-904938bffd73

[6] https://webrtc.org/getting-started/firebase-rtc-codelab

[7] https://socket.io/docs/v4/

[8] https://www.w3schools.com/nodejs/nodejs_mysql_create_db.asp

[9] https://www.npmjs.com/package/@faker-js/faker

[10] Officiele Socket.IO github pagina: https://github.com/socketio/socket.io/blob/main/LICENSE

6 Bijlagen

Je voegt hier je genummerde bijlagen toe (bijv de enquête die je hanteerde, de uitprint van bepaalde resultaten, ….) die belangrijk zijn om het verslag op een goede manier te kunnen 'lezen' en 'interpreteren'.

Tekst tekst tekst