Vraag 1:

Zoek op het internet naar afbeeldingen op basis van de zoektermen http client server. Teken hieronder een schematische voorstelling van de communicatie tussen een client en een server. Voor computers gebruik je rechthoeken, voor berichten pijltjes en in het midden teken je een mysterieus wolkje dat 'het internet' voorstelt. Zorg ervoor dat de volgende termen erop aangeduid staan: client, server, request, response, internet, tekst en geef met getallen de volgorde van de berichten aan.

Antwoord:

Response (4)

Request (2)

internet

Server (3)

Client (5)

Client (1)

Vraag 2:

Duid de verschillende onderdelen van de volgende URL: https://www.bol.com/nl/p/hoe‐werkt‐dat‐ nou/9200000057347012/?country=BE&suggestionType=browse#product\_alternatives

Antwoord:

https:// -> Dit is het protocol dat gebruikt moet worden om de resource op te vragen.

[www.bol.com](http://www.bol.com) -> domeinnaam

/nl/p/hoe‐werkt‐dat‐ nou/9200000057347012/ -> Dit is de pagina die u wilt.

?country=BE&suggestionType=browse -> Dit zijn de parameters die worden meegegeven.

#product\_alternatives -> Welke regel op de pagina dat je wilt zijn.

Vraag 3:

Open de Chrome developer tools (F12), ga naar het Network tabblad en surf naar: https://www.htmldog.com/examples/headings1.html.

Vraag 4:

In het voorgaande voorbeeld heeft de browser niet alleen een request voor de headings1.html resource verstuurd. Welke resources heeft je browser nog meer opgevraagd? Hoe zie je dit? Kun je in het HTML document in het response body terugvinden waarom net die resources werden opgevraagd? Merk op dat browser plugins zoals anti‐virus programma's en ad‐blockers sommige requests kunnen tegenhouden en ook eigen requests kunnen versturen!

Antwoord: Je kan bij inspect naar network tab gaan en dan bij name (links) gaan zien welke documenten er zijn geladen.

Kun je in het HTML document in het response body terugvinden waarom net die resources werden opgevraagd?

Antwoord: Er is een image met een source (src=”link”) in en daardoor die image automatisch gaat gaan opvragen aan de server.

Vraag 5:

Surf naar https://www.vives.be. Welke andere soorten resources worden opgevraagd door het inladen van deze pagina?

Antwoord: Er zijn scripts, fonts, css, img en document opgevraagd.

Werden alle requests naar dezelfde server verstuurd? Om dit te zien kun je best een 'domain' kolom toevoegen aan de tabel. Rechtsklik hiervoor op de hoofding van de tabel en vink 'Domain' aan.

Antwoord: Neen, er zijn verschillende websites die werden gebruikt.

Het opvragen van 2 resources van een webpagina kan normaliter onafhankelijk van elkaar gebeuren, de browser kan dus een pagina sneller kunnen inladen door een volgend request te versturen, nog voor het response op een vorig request werd ontvangen.

Hoe kun je dit uit de timing informatie afleiden?

Antwoord: De volgorde die werd opgevraagd zie je van boven naar onder. Als je dan kijkt naar time kun je zien dat de timing bij de ene langer is dan bij andere.A screenshot of a computer

Description automatically generated with medium confidence

Lijkt het erop dat het aantal gelijktijdige ‘onafgewerkte’ requests beperkt is? (onafgewerkt, in de zin dat het request verstuurd is maar nog geen response werd ontvangen).

Antwoord: Er zijn er maar 13 met waiting for server.

A picture containing graphical user interface

Description automatically generated

Vraag 6:

Open het Netwerk tabblad van de Chrome developer tools, surf naar je favoriete webmail client, e.g., gmail, en log in.

Wacht tot 'alles' ingeladen is en zorg ervoor dat je niet met de muis over de eigenlijk webpagina gaat (beperk de bewegingen tot het network tabblad). Afhankelijk van welke webmail provider je gebruikt, zul je zien dat er na verloop van tijd requests bijkomen ook al verandert er niks aan de webpagina.

Waarvoor zouden die 'spontane' requests dienen?

Antwoord: Hij kijkt of er nieuwe mails toekomen.

Vraag 7:

Surf naar https://www.vives.be/dezepaginabestaatniet.hml. Merk op dat er wel degelijk een response teruggestuurd wordt alhoewel de pagina niet bestaat. Hoe komt dit?

Antwoord: Omdat de server een standaard pagina stuurt als er niks word gevonden.

Wat betekent de status code 404 in de response header?

Antwoord: Betekent page not found

Vraag 8:

Probeer nu eens https://www.ditdomeinbestaatnogniet.be/blabla.html. Wat is het verschil met de vorige opdracht?

Antwoord: De server heeft geen standaard pagina als de pagina niet word gevonden en toont daardoor niks.

Vraag 9:

Zoek onderstaande HTTP status codes op (bv. wikipedia) die in een response kunnen voorkomen en schrijf hun betekenis op. Omcirkel deze die je al eens bent tegengekomen bij het surfen. Schrijf ook een gepaste hoofding boven de kolommen.

Antwoord:

200 -> Means “succes” of “ok”

204 -> There is no content to send for this request, but the headers may be useful.

301 -> Resource has been moved permanently.

302 -> Request resource has been changed temporarily

303 -> Ceck out another URI

400 -> Bad request

401 -> unauthorized

404 -> The server can not find the requested resource.

500 -> The server has encountered a situation it does not know how to handle.

503 -> The server is not ready to handle the request.

Table

Description automatically generated

server error resp.

Client error resp.

Redirection resp.

Succes resp.

Vraag 10:

Zoek op het internet welke HTTP request methods er bestaan en schrijf ze neer. Waarvoor dienen de vaak gebruikte GET en POST methods?

Antwoord:

GET – Ontvang het document gespecificeerd door de URL.

HEAD – Ontvang alleen de headers van het op te vragen document.

POST – Zend gegevens naar de server.

PUT – Vervang het document op de server door de verzonden data.

DELETE – Verwijder het document.

Waar in een request staat aangegeven om welke request method het gaat, en hoe vind je dit terug in de Chrome developer tools (zie uitleg bij Opdracht 3)?

Antwoord: Bij network en status

Als je een url in de adresbalk van je browser typt en op enter drukt, wat voor request method gebruikt de browser dan om die resource op te vragen bij de server? Als je in een webpagina op een gewone hyperlink klikt, welke request method wordt er dan gebruikt?

Antwoord: Als je gewoon een url in de adresbalk typt en op enter duwt gebruik je een GET method. En als je op een hyperlink klikt doe je gewoon nog eens een GET.

Stel, iemand schrijft een webapplicatie om producten te beheren en realiseert de "wis productgegevens" functionaliteit door middel van een gewone hyperlink met 'wis' opschrift. Om een product te wissen moet een gebruiker dus gewoon op de 'wis' link klikken bij dit product. Op een bepaald moment komt bv. de google‐bot langs die (programmatorisch) alle links uitprobeert om te zien wat dit oplevert aan nieuwe pagina's om te indexeren. Of stel dat de browser een accelerator plugin bevat die proactief gelinkte pagina's inlaadt zodat de gebruiker niet hoeft te wachten bij het klikken op een link.

Wat zou er dan gebeuren met de productgegevens? Dit is trouwens geen hypothetisch geval, zie bv.

• https://thedailywtf.com/articles/The\_Spider\_of\_Doom  
• https://thedailywtf.com/articles/WellIntentioned‐Destruction  
• https://betanews.com/2005/05/06/google‐web‐accelerator‐draws‐concern/

Antwoord:

Dan zou de gegevens verwijderen en dat zou dus niet goed zijn. De link mag pas werken als je er op klikt.

Vraag 11:

Ga naar het network tabblad van de Chrome developer tools en surf naar https://www.vives.be. Bekijk de vele requests die het inladen van die ene pagina heeft veroorzaakt. Hoeveel requests waren er in totaal?

Antwoord: 47 requests

Vraag 12:

Zorg dat de Chrome developer tools actief zijn. Open dan de vives.be pagina, rechtsklik op de refresh knop en kies '*Empty cache and hard reload*'.

Hiermee dwingen we de browser om bij het herladen van de pagina z'n cache te negeren en alle resources bij de server op te vragen.

Hoeveel kilobytes of megabytes aan data werd er verstuurd om alle nodige resources in te laden?

Antwoord: 2.7 MB

Hoe lang duurde het vooraleer alle resources van de pagina waren ingeladen?

Antwoord: 5.77 seconds

Klik nu gewoon op de refresh knop.

Kijk nogmaals hoeveel data er werd verstuurd. Waarom is dit zoveel minder? Laadde de pagina sneller?

Antwoord: 894 kb in 5.43. Dit is rapper omdat er veel gegevens in cache zijn gestoken.

Waar kun je zien welke documenten daadwerkelijk verstuurd werden en welke niet? Waar vindt de browser dan de inhoud van de documenten die niet bij de server werden opgevraagd? Hoe weet de browser welke documenten best opgevraagd (moeten) worden en welke niet? M.a.w. hoe lang mag de browser een bepaalde resource als 'vers' te beschouwen? (Hint: kijk eens naar de response

headers).

Antwoord: Bij “size” kun je zien welke er in memory cashe zitten, welke in disk cashe en welke er worden opgevraagd door het aantal mb te zien staan. Als je op een document klikt en kijkt naar de response header dan kun je een expiring date terugvinden en dat is hoe lang die blijft opgeslaan in uw cashe.

Vraag 13:

Installeer in Chrome de *Ghostery* extension. Bezoek nu de volgende webpagina's:

Antwoord:

* <https://www.nieuwsblad.be/> (10)
* <https://www.cnn.com> (9)
* <https://www.vives.be> (8)
* <https://www.vrt.be/vrtnws/nl/> (8)

Schrijf bij elke webpagina hoeveel tracking scripts door Ghostery worden ontdekt. Welke soorten

tracking scripts je bent tegengekomen (beacon, advertising, analytics, etc.)?

Antwoord:

* Google analytics
* Google tag manager
* Tealium consent
* Gemius
* Double click
* Amazon advertisment
* Outbrain emplify
* Bounce exchange
* Dianomy
* Onetag
* Bing ads
* Addthis
* Facebook connect
* Linkedin analytics

Vraag 14:

Zorg ervoor dat je browser venster gemaximaliseerd is en surf naar https://www.starbucks.com. Maak nu je browser venster gaandeweg minder breed en let erop hoe de inhoud van de webpagina niet enkel opschuift maar ook daadwerkelijk verandert!

Deze techniek heet **responsive design** en komt later nog aan bod in de labo’s.

Doorgaans verschuiven elementen en/of verkleinen ze, maar op sommige breedtes is er echter een abrupte overgang naar een andere layout. Neem bij elke abrupte overgang een screenshot (bv. met het *Snipping Tool* in Windows). Hoeveel verschillende layouts tel je?

Antwoord: Ik tel 2 layouts

Graphical user interface

Description automatically generated

A picture containing text, screenshot

Description automatically generated