

Onderzoeksverslag

– HTTP –



Student:	A.W. Janisse
Studentnummer:	2213829
Instelling:	Fontys hogescholen te Eindhoven
Opleiding:	ICT & Technology
Docent:	Dhr. Erik Dortmans

Opdracht:	8, HTTP
Datum:	23 april 2015

Inleiding

In deze opdracht gaan we het HyperText Transfer Protocol (HTTP) verkennen. HTTP is een Internet applicatie protocol. Het is het werkpaard van het World Wide Web. HTTP wordt gebruikt door Browsers om webpagina's (en plaatjes die daarin staan) op te halen van Websites. Het wordt echter ook gebruikt om vanuit software programma's remote Webservices te benaderen. HTTP is een client/server protocol. Een HTTP server (http daemon) luister op een poort (normaliter poort 80) naar requests van clients. Elke request wordt beantwoord in de vorm van een respons. Een HTTP Client die een bepaald resource in een bepaald formaat (b.v. HTML, XML of JSON) wil ontvangen maakt een connectie naar de betreffende host/port, stuurt een http request, en wacht op de respons van de HTTP Server.

Taken

In deze opdracht wordt de uitwerking van de volgende taken beschreven:

1. HTTP handmatig. Handmatig via PuTTY met een webserver communiceren.
2. HTTP via cURL. cURL gebruiken om resources op te halen bij een HTTP server.
3. Communicatie met een REST Webservice.

Uitwerking taak 1

In deze taak wordt in Putty de volgende commando's gegeven:

GET /index.html HTTP/1.1

Host: www.example.com

Het resultaat heb ik in SublimeText geplakt om zodoende schermafbeeldingen te kunnen maken met daarin regelnummers. Het is dan eenvoudiger om het antwoord te formatteren en eventuele stukken code dicht te klappen. Het resultaat op de gegeven commando's is afgebeeld in afbeelding 1.

```
1 HTTP/1.1 200 OK
2 Accept-Ranges: bytes
3 Cache-Control: max-age=604800
4 Content-Type: text/html
5 Date: Thu, 16 Apr 2015 20:49:05 GMT
6 Etag: "359670651"
7 Expires: Thu, 23 Apr 2015 20:49:05 GMT
8 Last-Modified: Fri, 09 Aug 2013 23:54:35 GMT
9 Server: ECS (iad/18F0)
10 X-Cache: HIT
11 x-ec-custom-error: 1
12 Content-Length: 1270
13
14 <!doctype html>
15 <html>
16 <head>
53 </head>
54
55 <body>
56   <div>
57     <h1>Example Domain</h1>
58     <p>This domain is established to be used for illustrative examples in documents. You may use this
59       domain in examples without prior coordination or asking for permission.</p>
60     <p><a href="http://www.iana.org/domains/example">More information...</a></p>
61   </div>
62 </body>
63 </html>
64
```

Afbeelding 1

Betekenis van het resultaat:

Regel 1: De HTTP-Statuscode is in het volgende formaat gegeven:

HTTP/1.1 <status code> <omschrijving>

HTTP/1.1 geeft het protocol versie aan, in dit geval 1.1. De code 200 geeft aan dat de operatie succesvol was en de daarbij behorende omschrijving is OK. In tabel 1 is een overzicht van verschillende status codes en de betekenis gegeven.

Regel 2: The Accept-Ranges response-header field allows the server to indicate its acceptance of range requests for a resource¹

Regel 3: De *Cache-Control* header geeft aan hoe lang een bestand in de cache mag staan z onder na te gaan of er een nieuwe versie is. In dit geval is er 604800 seconden opgegeven wat overeenkomt met 7 dagen.

1 RFC 2616

- Regel 4:** De *Content-Type* header geeft aan wat het MIME-type is van het geretourneerde document. In dit geval wordt er dus text/html terug gegeven maar een ander voorbeeld zou image/gif kunnen zijn.
- Regel 5:** De datum waarop de inhoud is opgevraagd.
- Regel 6:** De *Etag* is een van de mechanismes die wordt gebruikt voor cache validatie. De *Etag* is een soort digitale vingerafdruk of checksum van een document waarmee bepaald kan worden of de inhoud is gewijzigd. Als de inhoud gewijzigd zal dit een andere code opleveren.
- Regel 7:** De *expires* header geeft aan wanneer de data in een document ongeldig wordt. Zolang de data niet verlopen is mag de data uit de cache worden gebruikt. Als we even terug rekenen dan zien we dat het document opgevraagd is op 16 april en dat het verloopt op 23 april. Dit zijn dus 7 dagen welke precies overeenkomen met de waarde in regel 3 (604800) .
- Regel 8:** De *Last-Modified* header geeft aan wanneer het document voor het laatst is veranderd. Deze informatie kan in een later stadium worden gebruikt bij een *GET* die met *If-Modified-Since* conditioneel is gemaakt.
- Regel 9:** De *Server* header geeft aan welke webserver er gebruikt is. In dit geval gaat het dus om ECS (Electronic Commerce Server).
- Regel 10:** Geeft aan dat er gebruik wordt gemaakt van een CDN² (Content Delivery Network). In grote lijnen betekend dit dat er een server zit tussen de server waar het document zich bevind en de eindgebruiker. Caching is bij dit model belangrijk en informatie hieromtrent word in deze header weergegeven.
- Regel 11:** De pre-fix 'x-' in de *x-ec-custom-error* header geeft aan dat het om een server specifieke header gaat. Hierbij moet worden opgemerkt dat deze notatie is afgeschaft. De 'ec-' zou aan kunnen geven dat het om de webserver ECS (Elastra Enterprise Cloud Server) gaat. De error code die dus wordt gegeven is dus specifiek voor de ze webserver.
- Regel 12:** De *Content-Length* header geeft aan hoe groot het document is. De grootte wordt opgegeven in bytes. De header bytes tellen niet mee.
- Regel 14 t/m 63:** Geeft de inhoud van het opgevraagde document weer.

2 http://en.wikipedia.org/wiki/Content_delivery_network

Statuscode	Betekenis
200	Operatie succesvol uitgevoerd.
204	Operatie succesvol uitgevoerd, cliënt krijgt geen nieuwe pagina.
301 / 302	URL van document is veranderd.
304	Cliënt deed een <i>If-Modified-Since</i> en het document is onveranderd.
401	Cliënt is niet geautoriseerd voor de gevraagde operatie.
403	Gevraagde operatie is verboden of autorisatie niet mogelijk.
404	Gevraagde document bestaat niet.
500	Interne fout opgetreden in de server.
503	Server is momenteel niet in staat om verzoek af te handelen, mogelijk door te hoge belasting.

Tabel 1: HTTP - Statuscodes

Het logische vervolg zou zijn om op de link 'More information...' te klikken. Deze actie uitgevoerd in Putty zal ik hierna beschrijven. De URL hiervoor is: <http://www.iana.org/domains/example>. Zie afbeelding 1 regel 60. Om dit te doen maak in Putty een nieuwe verbinding aan zoals ik dat eerder heb gedaan voor www.example.com maar gebruik nu www.iana.org als host. Het antwoord is weergegeven in afbeelding 2.

```

1  HTTP/1.1 302 Found
2  Server: Apache
3  Location: /domains/reserved
4  Content-Type: text/html; charset=iso-8859-1
5  Content-Length: 201
6  Accept-Ranges: bytes
7  Date: Sat, 18 Apr 2015 11:07:53 GMT
8  X-Varnish: 563073448
9  Age: 0
10 Via: 1.1 varnish
11 Connection: keep-alive
12
13 <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
14 <html>
15   <head>
16     <title>302 Found</title>
17   </head>
18   <body>
19     <h1>Found</h1>
20     <p>The document has moved <a href="/domains/reserved">here</a>.</p>
21   </body>
22 </html>
23

```

Afbeelding 2

In de ontvangen headers is te zien dat we te maken hebben met een re-direct (302). De URL voor deze re-direct is te vinden in de header *Location*. Een browser handelt deze re-direct direct af en zal uiteindelijk op de opgegeven locatie /domains/reserved uitkomen.

Ik zal dus in Putty nu de opdracht geven: GET /doamains/resreved HTTP/1.1. In afbeelding 3 is te zien dat ik nu op de uiteindelijk bestemming ben aangekomen. De status code is 200 en er zijn dus geen re-directs meer.

```
1 HTTP/1.1 200 OK
2 Server: Apache
3 Last-Modified: Fri, 24 Oct 2014 19:37:05 GMT
4 Vary: Accept-Encoding
5 Content-Type: text/html; charset=UTF-8
6 Transfer-Encoding: chunked
7 Date: Sat, 18 Apr 2015 11:21:36 GMT
8 X-Varnish: 563089379
9 Age: 0
10 Via: 1.1 varnish
11 Connection: keep-alive
12
13 0027e9
14 <!doctype html>
15 <html>
16 <head>
17 <title>IANA – IANA-managed Reserved Domains</title>
18
```

Afbeelding 3

Uitwerking taak 2

Als eerste zal ik beginnen om door middel van cURL de website www.example.com op te roepen. Afbeelding 4 toont hiervan het resultaat.

```
student@student-fontys:~/bart/programming/school/opdracht.8/code$ curl http://www.example.com
<!doctype html>
<html>
<head>
  <title>Example Domain</title>

  <meta charset="utf-8" />
  <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <style type="text/css">
    body {
      background-color: #f0f0f2;
      margin: 0;
      padding: 0;
      font-family: "Open Sans", "Helvetica Neue", Helvetica, Arial, sans-serif;
    }
    div {
      width: 600px;
      margin: 5em auto;
      padding: 50px;
      background-color: #fff;
      border-radius: 1em;
    }
    a:link, a:visited {
      color: #38488f;
      text-decoration: none;
    }
    @media (max-width: 700px) {
      body {
        background-color: #fff;
      }
      div {
        width: auto;
        margin: 0 auto;
        border-radius: 0;
        padding: 1em;
      }
    }
  </style>
</head>
<body>
<div>
  <h1>Example Domain</h1>
  <p>This domain is established to be used for illustrative examples in documents. You may use this
  domain in examples without prior coordination or asking for permission.</p>
  <p><a href="http://www.iana.org/domains/example">More information...</a></p>
</div>
</body>
</html>
student@student-fontys:~/bart/programming/school/opdracht.8/code$
```

Afbeelding 4

Logo downloaden

Om het logo van Google te downloaden heb ik eerst de header opgevraagd. Afbeelding 5 laat zien dat het *Content-Type* een image in het png formaat is.

```
student@student-fontys:~/bart/programming/school/opdracht.8/code$ curl --head https://www.google.nl/images/srpr/logo11w.png
HTTP/1.1 200 OK
Content-Type: image/png
Content-Length: 14022
Last-Modified: Wed, 09 Oct 2013 01:35:39 GMT
Date: Sat, 18 Apr 2015 13:45:25 GMT
Expires: Sat, 18 Apr 2015 13:45:25 GMT
Cache-Control: private, max-age=31536000
X-Content-Type-Options: nosniff
Server: sffe
X-XSS-Protection: 1; mode=block
Alternate-Protocol: 443:quic,p=1
```

Afbeelding 5

Om het logo te downloaden heb ik het commando gebruikt zoals te zien is in afbeelding 6. Hierin is ook het bestand logo.png te zien.

```
student@student-fontys:~/bart/programming/school/opdracht.8/code$ curl -o logo.png https://www.google.nl/images/srpr/logo11w.png
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 14022 100 14022 0 0 118k 0 --:--:-- --:--:-- --:--:-- 118k
student@student-fontys:~/bart/programming/school/opdracht.8/code$ ls
total 24K
-rw-rw-r-- 1 student student 14K apr 18 15:52 logo.png
-rw-rw-r-- 1 student student 261 apr 18 14:51 Makefile
-rw-rw-r-- 1 student student 653 apr 18 15:43 simple.c
student@student-fontys:~/bart/programming/school/opdracht.8/code$
```

Afbeelding 6

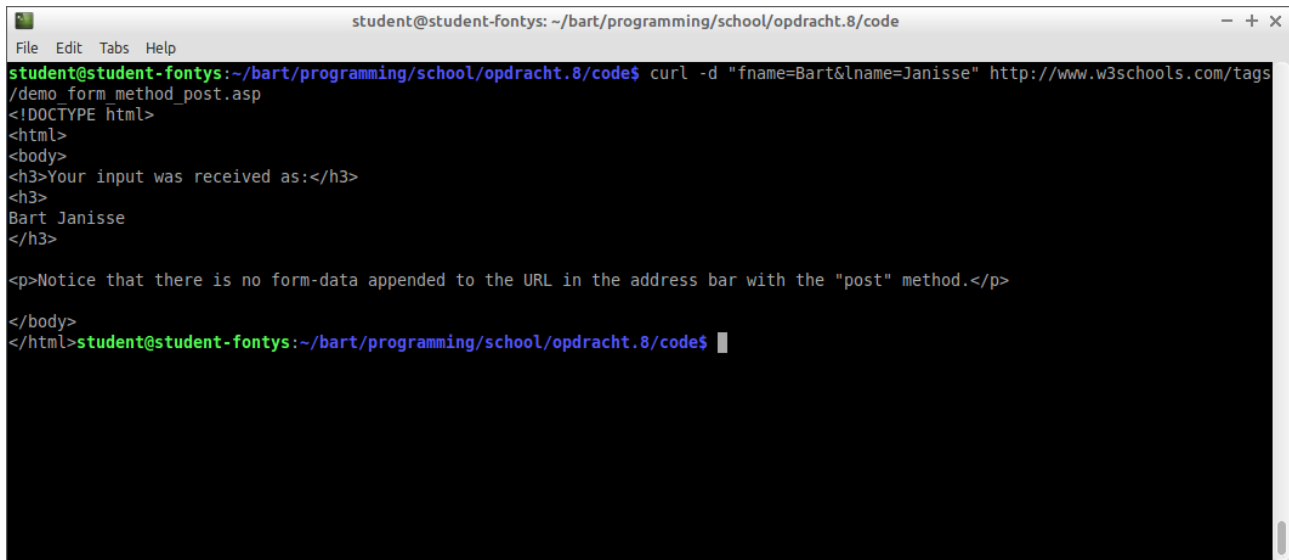
Afbeelding 7 toont dat het inderdaad gelukt is. Het logo is in de viewer zichtbaar.



Afbeelding 7

cURL POST

Om een post te doen gebruik ik de optie -d³. Hiermee kan ik de parameters meegeven. Afbeelding 8 toont het resultaat hiervan en zoals te zien is wordt mijn naam netjes weergegeven.

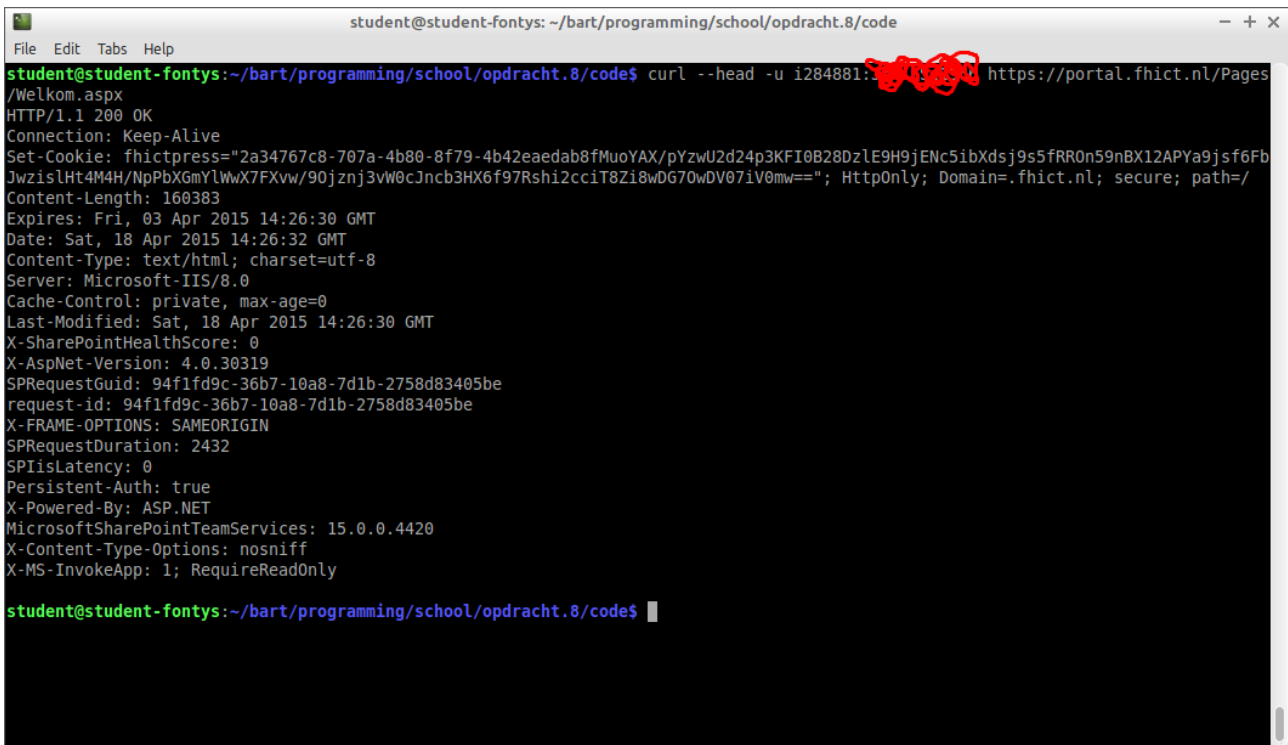
A terminal window titled 'student@student-fontys: ~/bart/programming/school/opdracht.8/code'. The prompt is 'student@student-fontys:~/bart/programming/school/opdracht.8/code\$'. The command entered is 'curl -d "fname=Bart&lname=Janisse" http://www.w3schools.com/tags/demo_form_method_post.asp'. The output is an HTML document. The first part is a header with <!DOCTYPE html>, <html>, and <body>. The main content is an <h3> tag containing 'Your input was received as:'. This is followed by another <h3> tag containing 'Bart Janisse'. Then there is a <p> tag with the text 'Notice that there is no form-data appended to the URL in the address bar with the "post" method.' The document ends with </body> and </html>. The terminal prompt is now 'student@student-fontys:~/bart/programming/school/opdracht.8/code\$'.

Afbeelding 8

3 (HTTP) Sends the specified data in a POST request to the HTTP server, in the same way that a browser does when a user has filled in an HTML form and presses the submit button. [man curl]

Fontys welkomspagina

OM de welkomspagina van Fontys te bereiken moet ik mijn username en password meegeven. Hiervoor kan ik de optie -u gebruiken. Het formaat voor deze optie is <username>:<password>. Tevens gebruik ik de --head optie om alleen de HTTP header zichtbaar te maken. In deze header is dus de statuscode 200 te zien. Het gehele commando is te zien in afbeelding 9.



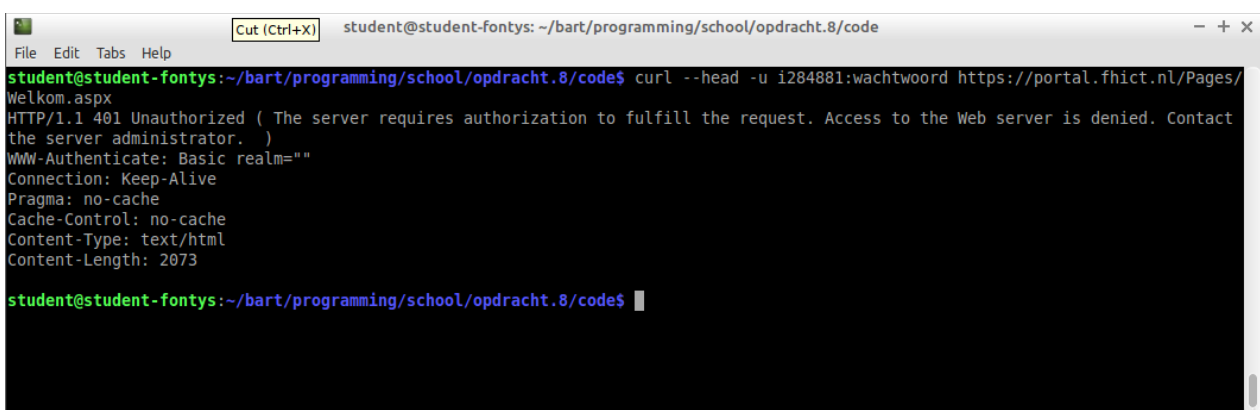
```
student@student-fontys: ~/bart/programming/school/opdracht.8/code
student@student-fontys:~/bart/programming/school/opdracht.8/code$ curl --head -u i284881: [redacted] https://portal.fhict.nl/Pages/Welkom.aspx
HTTP/1.1 200 OK
Connection: Keep-Alive
Set-Cookie: fhictpress="2a34767c8-707a-4b80-8f79-4b42eaedab8fMuoYAX/pYZwU2d24p3KFI0B28DzLE9H9jENC5ibXdsj9s5fRR0n59nBX12APYa9jsf6FbJwzislHt4M4H/NpPbXGmYLWwX7FXvw/90jznj3vW0cJncb3HX6f97Rshi2ccIT8Zi8wDG70wDV07iV0mw=="; HttpOnly; Domain=.fhict.nl; secure; path=/
Content-Length: 160383
Expires: Fri, 03 Apr 2015 14:26:30 GMT
Date: Sat, 18 Apr 2015 14:26:32 GMT
Content-Type: text/html; charset=utf-8
Server: Microsoft-IIS/8.0
Cache-Control: private, max-age=0
Last-Modified: Sat, 18 Apr 2015 14:26:30 GMT
X-ShaPeintHealthScore: 0
X-AspNet-Version: 4.0.30319
SPRequestGuid: 94f1fd9c-36b7-10a8-7d1b-2758d83405be
request-id: 94f1fd9c-36b7-10a8-7d1b-2758d83405be
X-FRAME-OPTIONS: SAMEORIGIN
SPRequestDuration: 2432
SPiisLatency: 0
Persistent-Auth: true
X-Powered-By: ASP.NET
MicrosoftSharePointTeamServices: 15.0.0.4420
X-Content-Type-Options: nosniff
X-MS-InvokeApp: 1; RequireReadOnly

student@student-fontys:~/bart/programming/school/opdracht.8/code$
```

Afbeelding 9

Extra

Ik kan het natuurlijk niet laten om te zien wat de response zou zijn met een fout wachtwoord. Afbeelding 10 toont dit. De status-code is nu dus een 401, cliënt is niet geautoriseerd.



```
student@student-fontys: ~/bart/programming/school/opdracht.8/code
student@student-fontys:~/bart/programming/school/opdracht.8/code$ curl --head -u i284881:wachtwoord https://portal.fhict.nl/Pages/Welkom.aspx
HTTP/1.1 401 Unauthorized ( The server requires authorization to fulfill the request. Access to the Web server is denied. Contact the server administrator. )
WWW-Authenticate: Basic realm=""
Connection: Keep-Alive
Pragma: no-cache
Cache-Control: no-cache
Content-Type: text/html
Content-Length: 2073

student@student-fontys:~/bart/programming/school/opdracht.8/code$
```

Afbeelding 10

Uitwerking taak 4

Voor deze taak mogen we zelf bepalen welke webservice we gaan gebruiken. Ik ga voor deze taak mijn Philips Hue⁴ gebruiken. Het Philips Hue systeem bestaat uit lampen en een bridge. De bridge communiceert met de lampen door middel van ZigBee en is aangesloten op mijn lokale netwerk. Via het netwerk is de RESTful API interface te benaderen. Hierna zal ik de Philips Hue bridge aanduiden als Hue.

Een volledige HTTP bestaat uit de volgende elementen:

URL	Het IP-adres van de Hue.
Body	Gedeelte van het bericht waarin beschreven is wat er gewijzigd moet worden en hoe. Het formaat hiervan is Json. Bijvoorbeeld {"on":false}
Commando	Het HTTP-commando kan een van de vier volgende zijn: GET Commando voor het opvragen van informatie over een resource. Bijvoorbeeld een lamp of timer. PUT Commando om een resource te wijzigen. Bijvoorbeeld lamp aan of uit. POST Commando om een nieuwe resource aan te maken DELETE Commando om een resource te wissen.
Response	De Hue geeft na het ontvangen van een bericht het resultaat hiervan in een Json formaat.

Voor dat er met de Hue gecommuniceerd kan worden moet er eerst een nieuwe gebruiker worden ingesteld. De gebruiker die ik heb aangemaakt heet 'linuxdeveloper'. Het aanmaken hiervan valt buiten het bestek van dit document.

Ter verduidelijking volgen twee voorbeelden, een om de status van een lamp op te vragen en een om een lamp uit te zetten.

Address	http://<bridge ip address>/api/linuxdeveloper/lights/1
Body	
Method	GET
Address	http://<bridge ip address>/api/linuxdeveloper/lights/1/state
Body	{"on":false}
Method	PUT

4 <http://www.developers.meethue.com/philips-hue-api>

Ik zal voor deze taak een applicatie maken die door middel van libcurl kan communiceren met mijn Hue. De applicatie heeft een eenvoudig menu waarmee de gebruiker uit de volgende mogelijkheden kan kiezen:

1. Het opvragen informatie van alle aanwezige devices (GET)
2. Het opvragen van de status van een device (GET)
3. Het aan en uit schakelen van een specifieke lamp (PUT)

Het opstarten van de applicatie moet gebeuren met het IP-adres en de username als parameter.

Als de applicatie gestart is wordt een menu getoond zoals afgebeeld in afbeelding 11 .

```
student@student-fontys:~/bart/programming/school/opdracht.8/code$ ./hue -i 192.168.123.75 -u linuxdeveloper
Menu
=====
[a]  get all devices
[g]  get device state
[s]  switch device on/off
[q]  quit
Enter choice: █
```

Afbeelding 11

Het opvragen van aanwezige devices kan met menu optie 'a'. Het resultaat hiervan staat in afbeelding 12. De devices worden getoond met hun id en hun naam.

```
Menu
=====
[a]  get all devices
[g]  get device state
[s]  switch device on/off
[q]  quit
Enter choice: a

1="Living colors"
2="Dimmable plug-in unit 1"
3="Dimmable plug-in unit 2"
4="Eettafel muurzijde"
5="Ronde lamp"
6="Vierkante lamp"
7="Eettafel kamerzijde"
8="Led strip"
9="Spot"
```

Afbeelding 12

Om van een specifiek device de status op te vragen kan menu optie 'g'. Als deze wordt gekozen moet er een device id worden opgegeven waarna de status van dit specifieke device wordt geprint. Afbeelding 13 laat hiervan het resultaat zien.

```
Menu
=====
[a]  get all devices
[g]  get device state
[s]  switch device on/off
[q]  quit
Enter choice: g

Enter id: 9

{"id"=9 , "name"="Spot", "on"=true, "bri"=36, "hue"=13398, "sat"=204

Menu
=====
[a]  get all devices
[g]  get device state
[s]  switch device on/off
[q]  quit
Enter choice: █
```

Afbeelding 13

Een device kan ook aan- of uitgeschakeld worden. Hiervoor is menu optie 's'. Als deze optie wordt gekozen kan na het opgeven van het device id ook worden opgegeven of het device aan of uit moet. Afbeelding 14 toont het uitschakelen en het resultaat hiervan.

```
Menu
=====
[a]  get all devices
[g]  get device state
[s]  switch device on/off
[q]  quit
Enter choice: s

Enter id: 9
Enter state [0, 1]: 0
[{"success":{"/lights/9/state/on":false}}]
Menu
=====
[a]  get all devices
[g]  get device state
[s]  switch device on/off
[q]  quit
Enter choice: g

Enter id: 9

{"id"=9 , "name"="Spot", "on"=false, "bri"=36, "hue"=13398, "sat"=204

Menu
=====
[a]  get all devices
[g]  get device state
[s]  switch device on/off
[q]  quit
Enter choice: █
```

Afbeelding 14

Verbeterpunten

De applicatie is op dit moment volledig in de file hue.c opgebouwd en voor een klein deel in hue_aux.c en hue_aux.h. Om het geheel beter te structureren zou het beter zijn om het geheel verder op te delen in verschillende files. Hiermee zou de code overzichtelijker worden maar ook beter herbruikbaar.