SPRC
Laboratorul #1 Comunicare Web HTTP

Versiunea 1

Responsabili: **Dorinel FILIP**

Obiectivele laboratorului

În urma parcurgerii laboratorului, studenții:

- Vor putea descrie protocolul HTTP și principalele noțiuni referitoare la comunicația prin HTTP;
- Vor cunoaște pricipalele metode de comunicare a parametrilor și de menținere a sesiunii folosite de aplicațiile Web dezvoltate folosind HTTP;
- Vor fi capabili să dezvolte o aplicație folosind primitivele unei biblioteci de tipul client Web.

Noțiuni teoretice

Pentru realizarea laboratorului sunt necesare noțiunile de bază despre funcționarea protocolului HTTP.

Asistanți cu atenție la prezentarea asistentului pentru a vă însuși principalele noțiuni.

Asiugrați-vă că v-ați lămurit cu referire la următoarele aspecte:

1. Care este formatul unei cereri HTTP (vezi Listing 1);
2. Care sunt principalele metode în HTTP și care este semantica recomandată pentru fiecare dintre ele (vezi RFC2616);
3. Care sunt principalele metode de a adăuga parametri unei cereri:
    - Encodați în adresă (vezi Listing 2);
    - Ca antete adiționale (vezi Listing 4);
    - În corpul cererii:
        - În format url-encoded (vezi Listing 3);
        - Ca alte formate (JSON, XML etc.) - vezi Listing 4;
        - În format mixt, folosind multipart (vezi Listing 5).
4. Ce sunt cookies și cum funcționează ele (vezi Listing 7);
5. Cum se pot folosi Cookies (sau alte mecanisme de stocare în browser) pentru a stabili o sesiune.

Pentru reactualizare/aprofundare vă recomandăm următoarele resurse:

- curs.upb.ro - Materialele de pe site-ul de curs.;
- IETF.org - RFC2616

Exemple de request-uri

În această secțiune listăm o serie ce cereri/răspunsuri HTTP care ar putea fi utile în exemplificarea noțiunilor necesare în cadrul laboratorului.

În cadrul exemplelor vom folosi:

- Paranteze ascuțite pentru a marca prezenta obligatorie a unui câmp;
- Paranteze drepte pentru a marca prezența opțională a unui câmp.

```
<METODA> <URL> <VERSIUNE>
[<HEADER1>: <VALOARE1>
<HEADER1>: <VALOARE1>
...
<HEADERn>: <VALOAREn>]
\r\n (linie goala)
[CORPUL CERERII]
```

Listing 1: Forma generală a unei cereri HTTP.

```
GET /?post_id=7&action=view HTTP/1.1
Host: example.com
\r\n (linie goala)
```

Listing 2: Cerere GET cu parametrii url-encoded.

```
POST /login HTTP/1.1
Host: example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 33

username=student&password=student
```

Listing 3: Cerere POST cu payload url-encoded.

```
POST /add-book HTTP/1.1
Authorization: Bearer <token>
Content-Type: application/json
User-Agent: PostmanRuntime/7.26.5
Host: example.com
Content-Length: 170

{
    "title": "Manual de Programarea Calculatoarelor",
    "authors": [
        "Prof.dr.ing. Florin Pop",
        "Ing.drd. Ion-Dorinel Filip"
    ],
    "pages": 500
}
```

Listing 4: Cerere POST cu payload JSON și headere suplimentare (User-Agent și Authorization).

```
POST /upload HTTP/1.1
Content-Length: 428
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryOmz20xyMCkE27rN7

------WebKitFormBoundaryOmz20xyMCkE27rN7
Content-Disposition: form-data; name="id"
```

```
Content-Type: text/plain

123e4567-e89b-12d3-a456-426655440000

------WebKitFormBoundaryOmz20xyMCkE27rN7
Content-Disposition: form-data; name="address"
Content-Type: application/json

{
  "street": "3, Garden St",
  "city": "Hillsbery, UT"
}

------WebKitFormBoundaryOmz20xyMCkE27rN7
Content-Disposition: form-data; name="profileImage "; filename="image1.png"
Content-Type: application/octet-steam

{...file content...}

------WebKitFormBoundaryOmz20xyMCkE27rN7--
```

Listing 5: Cerere POST cu payload multipart cu componente în diferite formate.

```
> GET / HTTP/1.1
> User-Agent: curl/7.29.0
> Host: ifconfig.co
> Accept: */*
>
< HTTP/1.1 200 OK
< Date: Wed, 14 Oct 2020 17:08:58 GMT
< Content-Type: text/plain; charset=utf-8
< Content-Length: 14
< Connection: keep-alive
< Set-Cookie: __cfduid=d26a2311ed4bcd8fe5555aab4b8db2b811602695338; expires=Fri, 13-
    Nov-20 17:08:58 GMT; path=/; domain=.ifconfig.co; HttpOnly; SameSite=Lax
<
141.85.241.99
```

Listing 6: Cerere GET cu raspuns 200OK.

```
====== Cererea 1
> GET / HTTP/1.1
> User-Agent: curl/7.29.0
> Host: ifconfig.co
> Accept: */*
>
< HTTP/1.1 200 OK
< Date: Wed, 14 Oct 2020 17:15:21 GMT
< Content-Type: text/plain; charset=utf-8
< Content-Length: 14
< Connection: keep-alive
< Set-Cookie: __cfduid=d492a7b8820e9514d828c932f04c403dd1602695721; expires=Fri, 13-
    Nov-20 17:15:21 GMT; path=/; domain=.ifconfig.co; HttpOnly; SameSite=Lax
<
141.85.241.99

====== Cererea 2
> GET / HTTP/1.1
> User-Agent: curl/7.29.0
```

```
> Host: ifconfig.co
> Accept: */*
> Cookie: __cfduid=d492a7b8820e9514d828c932f04c403dd1602695721
>
< HTTP/1.1 200 OK
< Date: Wed, 14 Oct 2020 17:15:29 GMT
< Content-Type: text/plain; charset=utf-8
< Content-Length: 14
< Connection: keep-alive
<
141.85.241.99
```

Listing 7: Doua cereri consecutive, cu Cookies.

Aplicații

Acest laborator dispune de checker online. Pentru a îl rezolva, trebuie să fiți conectați la Internet, pe durata realizării request-urilor. Rezolvarea se poate face în orice limbaj de programare doriți, folosind orice bibliotecă. O variantă sugerată este folosirea bibliotecii requests pentru Python 3.x: `https://requests.kennethreitz.org/en/master/user/quickstart/`.

Pentru fiecare task rezolvat corect veți primi un raspuns cu forma de mai jos:

```
{
    "status": "ok",
    "proof": "eyJ0eXAiO...."
}
```

Listing 8: Format răspuns checker.

Proof-ul este un token JWT semnat cu o cheie cunoscută de către asistenți. Pe baza acestor semnături, asistenții pot verifica rezolvarea laboratorului.

Exercitiul 1

Faceti un request HTTP de tip POST către https://sprc.dfilip.xyz/lab1/task1 care să respecte următoarele specificații:

- URL Encoded Parameters:
  - nume = numele vostru;
  - grupa = grupa voastra.
- Parametrii de tip POST (format url-encodded):
  - secret = "SPRCisNice"
- Header adițional:
  - secret2 = "SPRCisBest"

Serverul va întoarce un JSON care caracterizează request-ul primit (îl puteți folosi la debug). Afișați pe ecran rezultatul.

Pentru a vi se puncta acest task, faceți acelasi request la https://sprc.dfilip.xyz/lab1/task1/**check** si salvati raspunsul primit.

Atenție! Folosiți HTTPS. Dacă folosiți HTTP, veți fi redirecționați (HTTP 301).

Exercitiul 2

Faceti un request de tip POST către sprc.dfilip.xyz/lab1/task2 care să conțină un payload encodat în formatul JSON pentru următorul dictionar:

```
{'username':'sprc', 'password':'admin', 'nume':'numele vostru'}
```

Listing 9: Conținut cerere pentru task-ul 2.

Serverul va întoarce un răspuns privind loginul vostru. Pentru a vi se puncta acest task, salvați răspunsul de succes primit.

Exercițiul 3

La acest task, ne propunem logarea unui client folosind conceptul de sesiune, deci va trebui să rețineți și să utilizati cookies-urile primite de la server.

Pentru login, veți face request-ul de la Task-ul 2, către https://sprc.dfilip.xyz/lab1/task3/login.

Rezolvarea task-ului presupune ca, **în aceesi sesiune**, să faceți request-ul de mai sus (etapa de login) și un request de tip GET către /lab1/task3/check (accesul la presursa protejată, care face verificarea că sunteti logați prin sesiune).

Pentru a vi se puncta acest task, salvați răspunsul primit la GET.

Python Hint: https://requests.readthedocs.io/en/master/api/#request-sessions.

Observații

- Checker-ul există pentru a facilita notarea laboratorului, condiția de primire a punctajului fiind prezentarea validării oferite de acestea. Totuși, scopul parcurgerii exercițiilor este generarea unor **soluții *programatice* corecte**. Astfel, soluțiile hardcodate sau care ocolesc rezolvarea cerințelor nu vor fi punctate.

Acknowledgements

- Exemplul de request din Listing 5 este preluat de pe siteul `http://swagger.io/`.