Web Programming

Server-side programming IV.

Leander Jehl | University of Stavanger

Server-side programming

- Part I. handling requests
- Part II. templating
- Part III. handling data
- Part IV. cookies and sessions

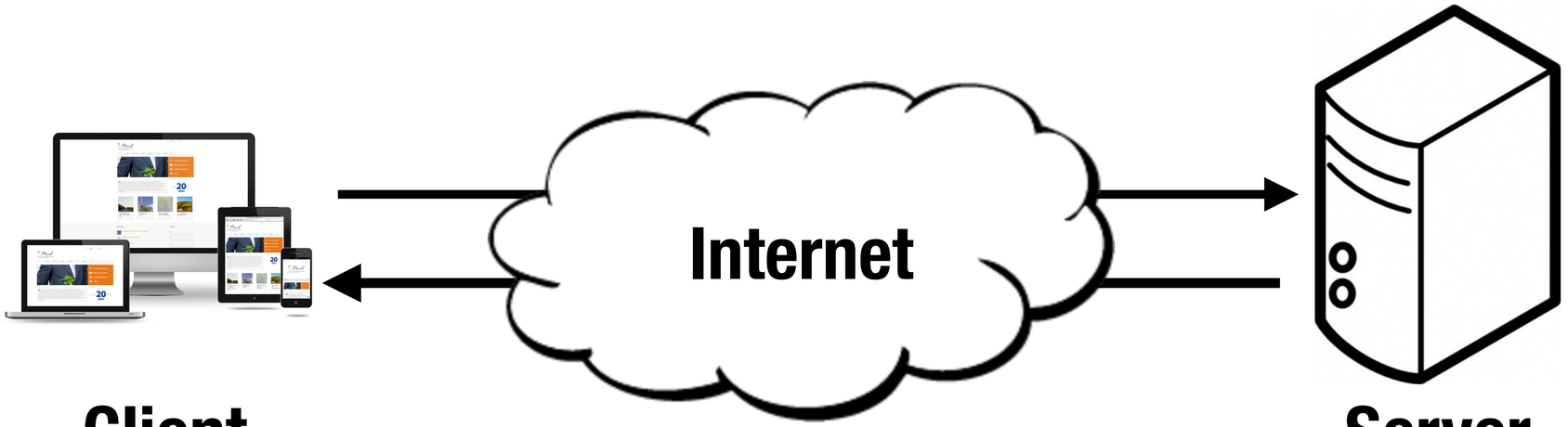
Some extra Flask bits

- Redirects and error pages
  - http://flask.pocoo.org/docs/0.12/quickstart/#redirects-and-errors
- Message flashing
  - http://flask.pocoo.org/docs/0.12/patterns/flashing/#message-flashing-pattern

Storing data

Files

Database



Client

Server

Cookie

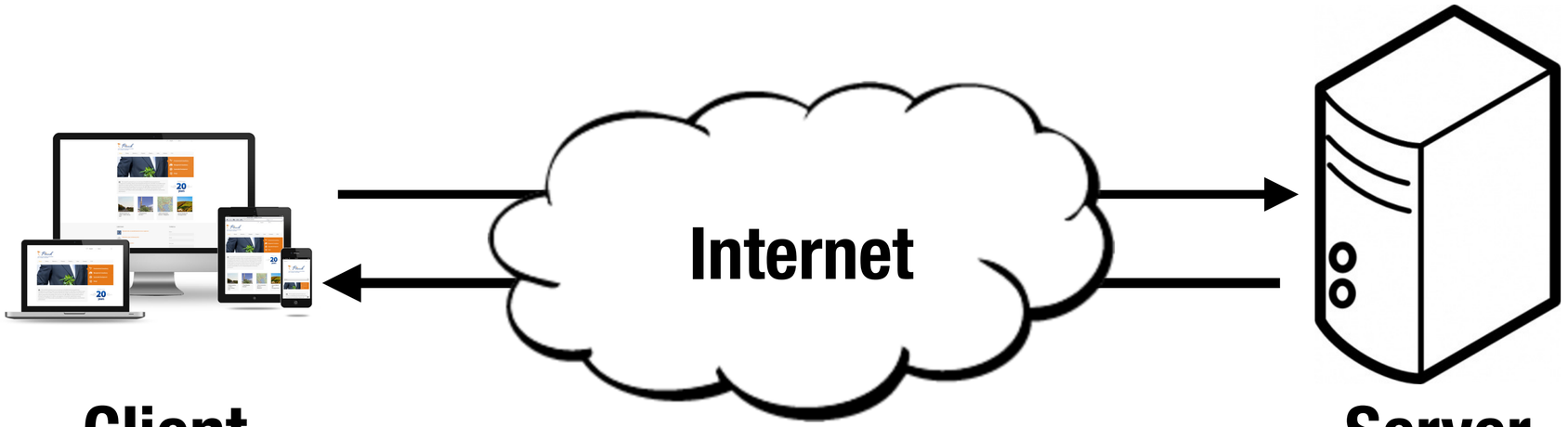
Session

Cookies

Storing data

Files

Database



Client

Server

Cookie

Session

Cookies

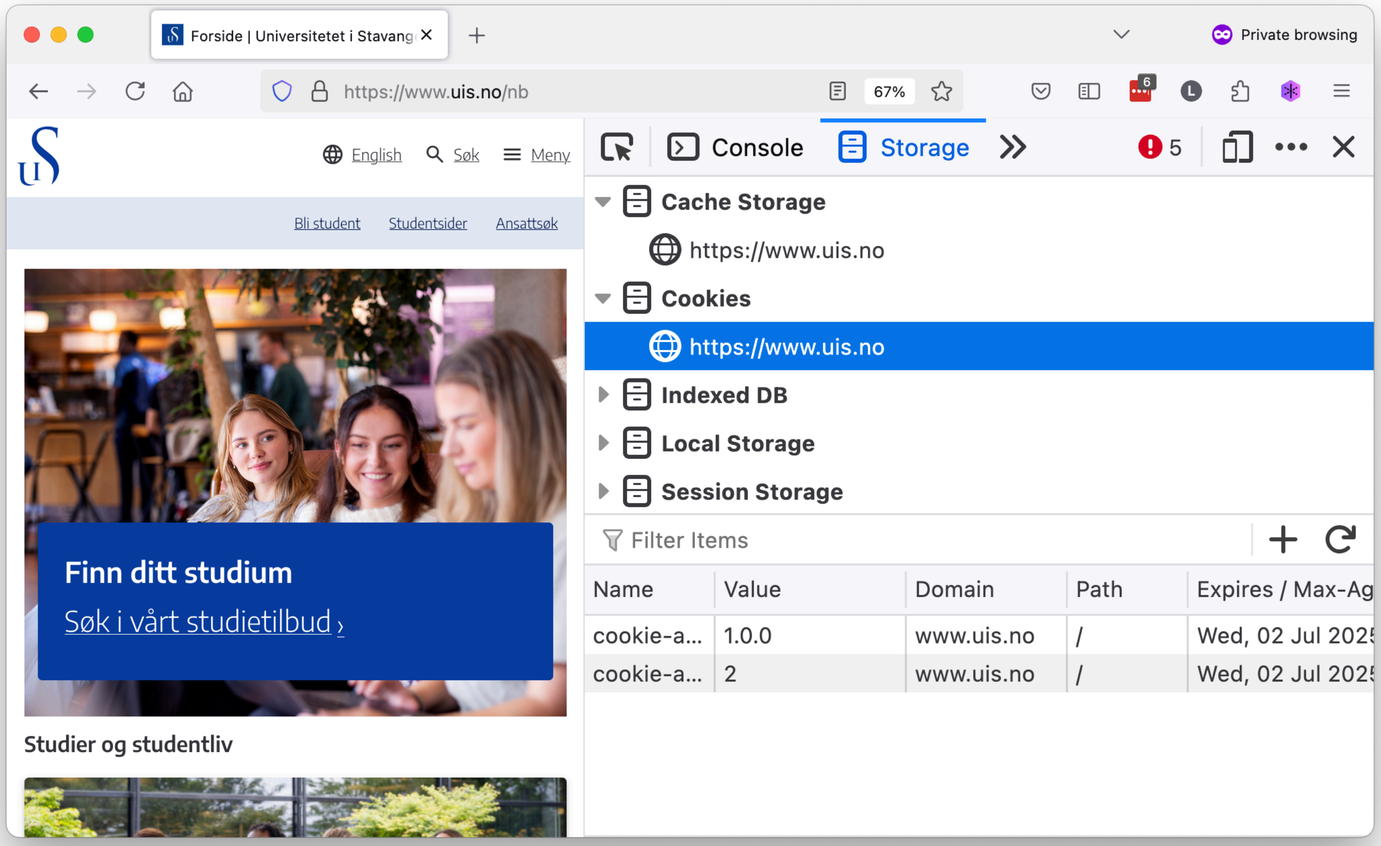
- Embedded on the user's computer
  - Small, often encrypted text files, located in the browser directories
- Cookies enable to remember and track data pertaining to a particular user (client) for a better visitor experience
  - Each time the same computer requests a page with a browser, it will send the cookie too
- Cookies are specific to the browser used
- Many misconceptions around cookies
  - ~~Transmit viruses~~
  - ~~Install malware on your computer~~

Cookies

- Within the context of a particular visit (always with respect to the domain that is shown in the browser's address bar)
  - **First-party cookie** => belongs to the same domain
  - **Third-party cookies** => belong to a different domain
- Typical usage
  - Tracking the user and her browsing activities (possibly for a long time)
  - Storing login information
- Same origin policy
  - You (as a site) can only view or set your own (i.e., first-party) cookie

Cookies can be viewed/edited

(developer tools / Storage tab / Cookies)



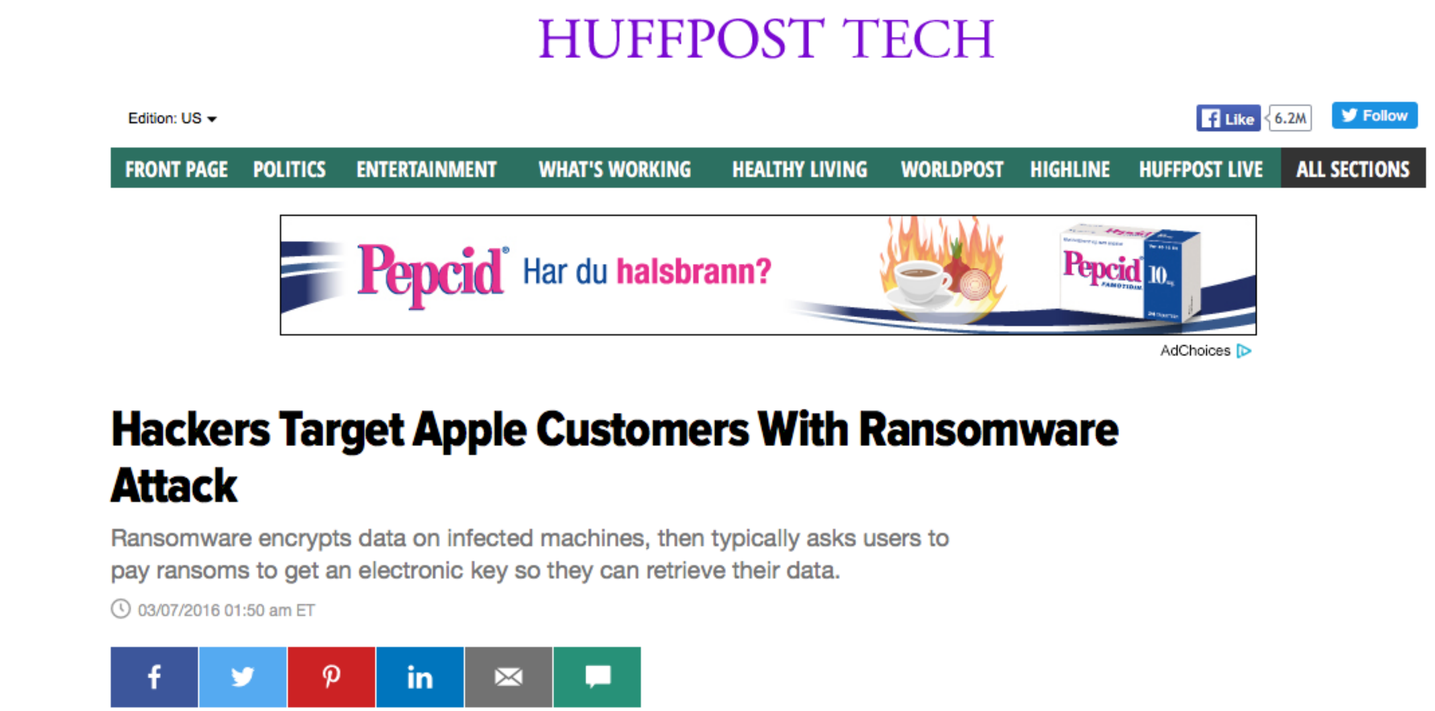
Third-party cookies

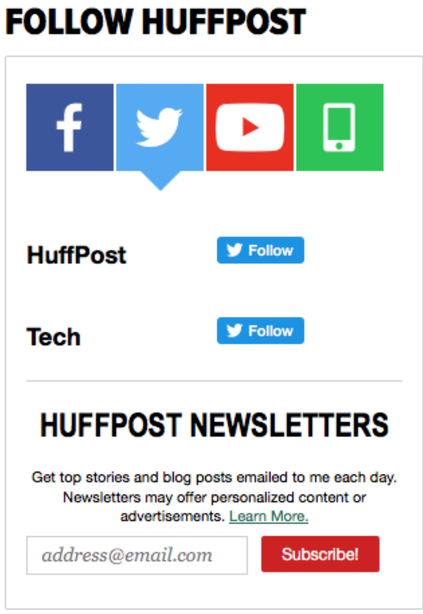
- Belong to domains different from the one shown in the address bar
- Typically used for "behind the scenes" tracking
  - So that advertisers can show you personalized banner ads
- When a piece of information is displayed from a third-party (image, advertisement, etc.), that site is allowed to set a cookie
- Each domain can only read the cookie it created!
- Can be blocked in the browser’s privacy settings!

User profiling with third-party cookies

- Suppose that a larger number of sites have banner adverts from www.advertiser.com
- It is possible for the advertiser to use its third party cookie to identify you as you move from one site to another site
- Even though it may not know your name, it can use the random ID number in the cookie to build up an anonymous profile of the sites you visit
  - “visitor 3E7ETW278UT regularly visits a music site, so show him/her adverts about music and music products”

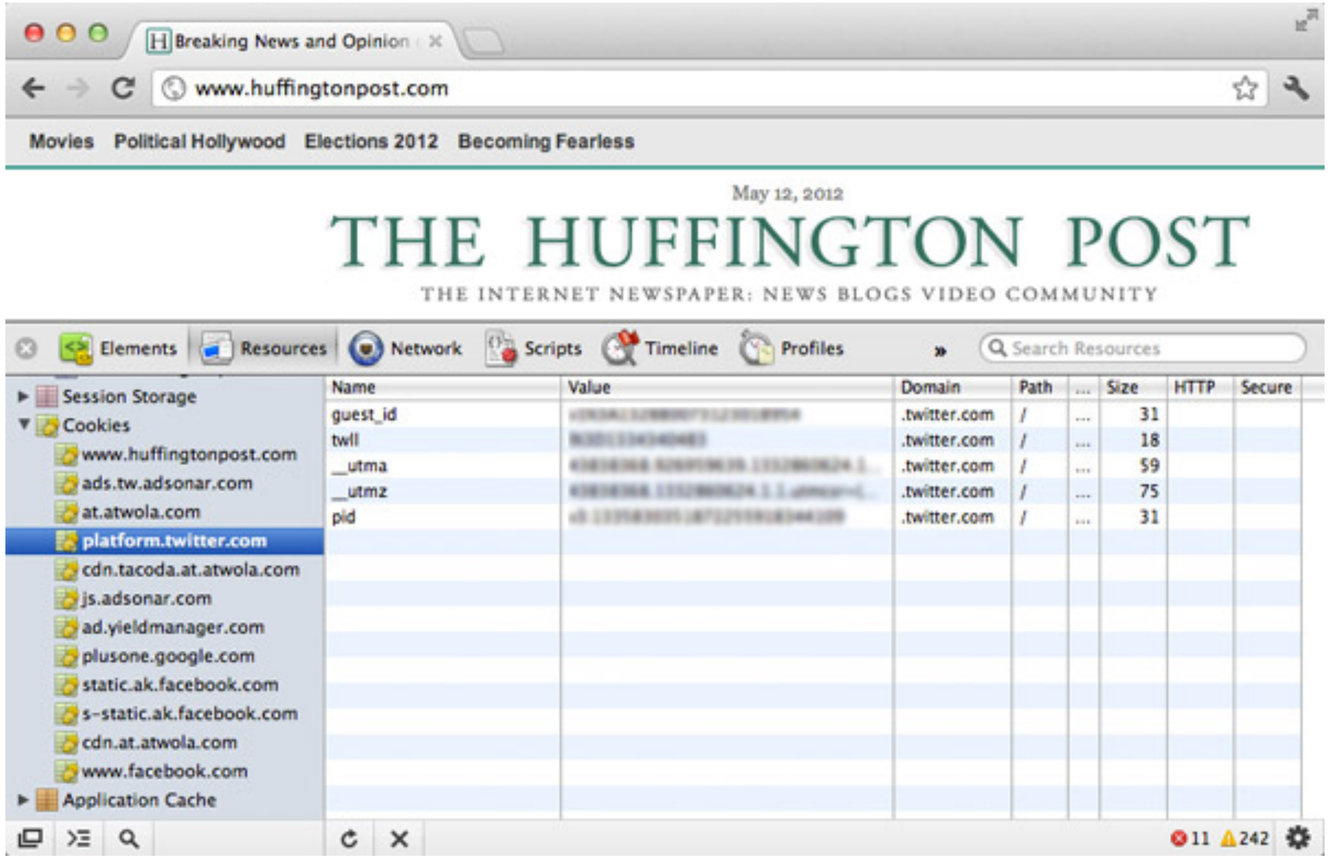
Example


HUFFPOST TECH
Edition: US
Like 6.2M
Follow
FRONT PAGE
POLITICS
ENTERTAINMENT
WHAT'S WORKING
HEALTHY LIVING
WORLDPOST
HIGHLINE
HUFFPOST LIVE
ALL SECTIONS
Pepcid
Har du halsbrann?
AdChoices
Hackers Target Apple Customers With Ransomware Attack
Ransomware encrypts data on infected machines, then typically asks users to pay ransoms to get an electronic key so they can retrieve their data.
03/07/2016 01:50 am ET


FOLLOW HUFFPOST
HuffPost
Follow
Tech
Follow
HUFFPOST NEWSLETTERS
Get top stories and blog posts emailed to me each day. Newsletters may offer personalized content or advertisements. Learn More.
address@email.com
Subscribe!

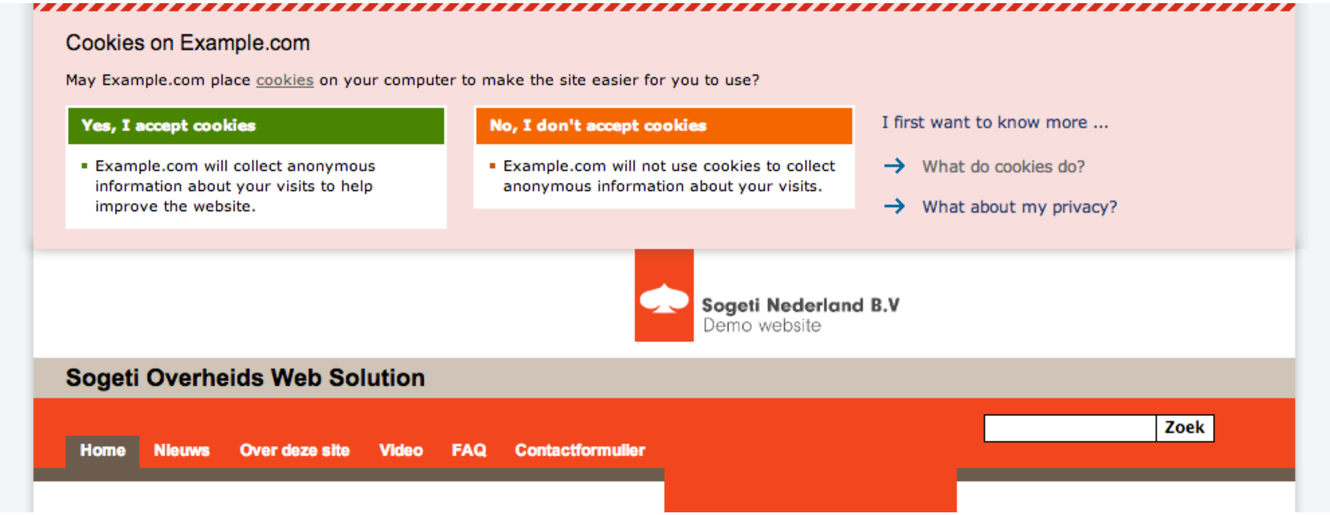
Example

Third-party cookies sent to Twitter



Cookie consent

- EU rules govern the use of cookies
  - Websites need to specifically gain the consent of their visitors



Cookies in Flask

- The cookies attribute of request contains a dictionary with all the cookies the client transmits
  - All cookie data are string!
- Reading cookies

```
username = request.cookies.get('username')
```

Use **`cookies.get(key)`** instead of **`cookies[key]`** to not get a KeyError if that variable is not in the cookie

- Storing cookies

```
response = make_response(render_template(...))
response.set_cookie("username", "the username")
return response
```

Cookies in Flask

- The cookies attribute of request contains a dictionary with all the cookies the client transmits
  - All cookie data are string!
- Reading cookies

```
username = request.cookies.get('username')
```

- Storing cookies

```
response = make_response(render_template(...))
response.set_cookie("username", "the username")
return response
```

Create a **Response** object, on which cookies can be set using the **set_cookie()** method

Cookies in Flask

- Expiry date
  - Additionally, it's possible to set an expiration date and time for a cookie
  - By default, Flask sets expiration to 31 days
  - The browser is responsible for managing the cookies' expiration, it's not possible to read these values on the server-side

```
import datetime

expiry_date = datetime.datetime.now() + datetime.timedelta(days=90)
response.set_cookie('id', my_id, expires=expiry_date)
```

expires should be a *datetime* object or a UNIX timestamp

Cookies in Flask

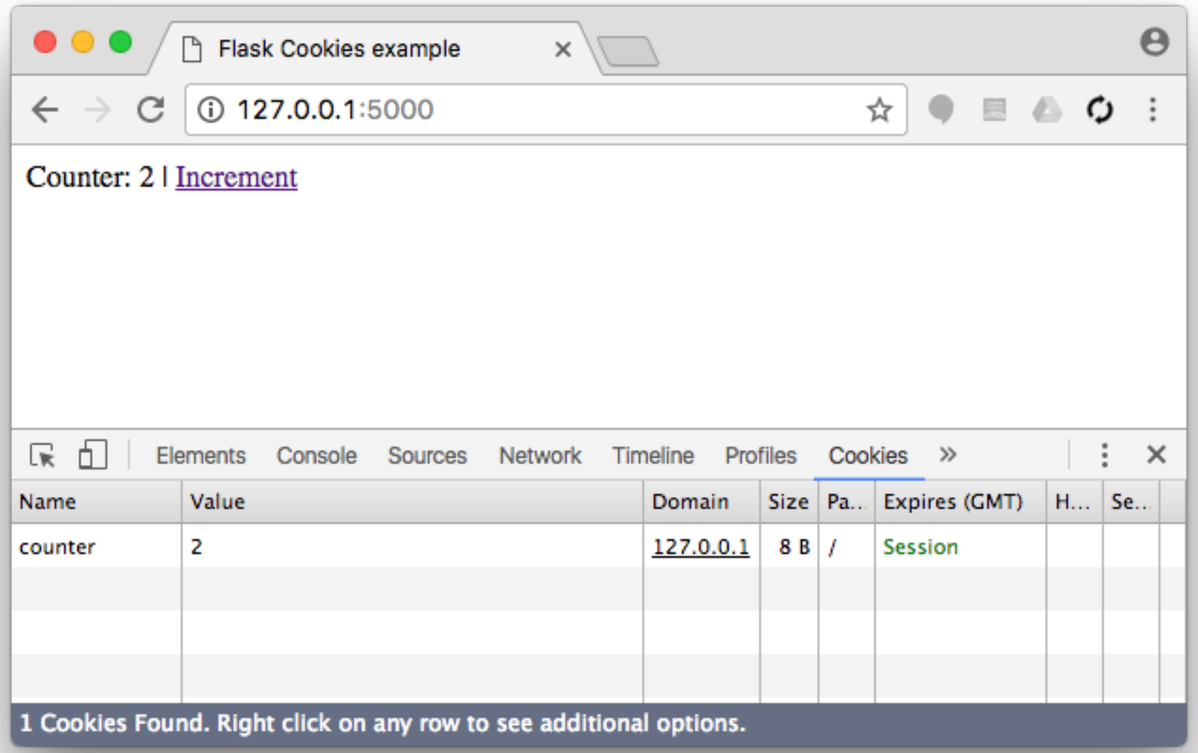
- Deleting cookies
  - Set it to a dummy value (empty string) and set its expiry date in the past

```
response.set_cookie('id', "", expires=0)
```

Example

examples/python/flask/6_cookies/app.py

- Incrementing a counter that is stored in a cookie



Exercise #1

github.com/dat310-2025/info/tree/master/
exercises/python/flask4

Cookies in JavaScrip

- Can also access cookies from JavaScript
  - access cookies via **`document.cookie`**
  - returns a string **“key=value; key=value”**

```
let cookies = document.cookie.split(";");
```

  - set cookies as string **“key=value”**

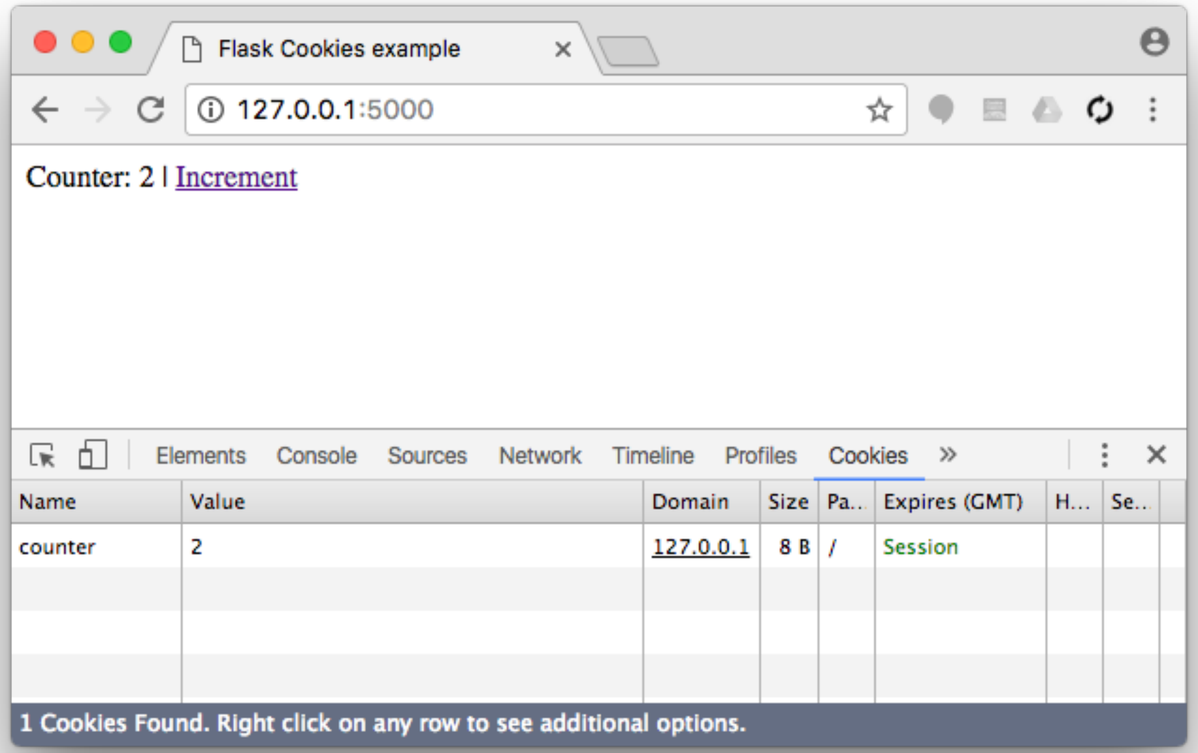
```
document.cookie = "counter=" + counter.toString();
```

https://www.w3schools.com/js/js_cookies.asp

Example

examples/python/flask/6_cookies_js/app.py

- Incrementing the counter in js

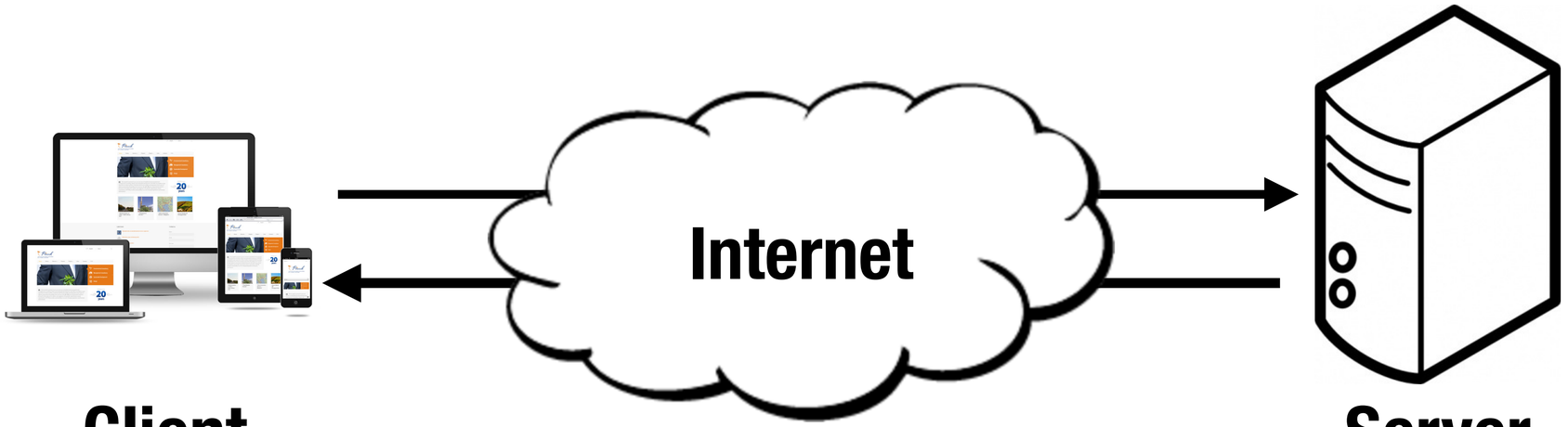


Sessions

Storing data

Files

Database



Client

Server

Cookie

Session

Sessions

- Store information on the server temporarily
  - It will be deleted after the user leaves the website (or closes the browser)
- Each browsing session is identified by a unique ID
  - sessionID is stored in a cookie
- The session is also a dictionary object with key-value pairs

A Note about Sessions in Flask

- Sessions, by definition, should be stored on the *server side*
- Flask, however, stores sessions by default on the *client side*, as encrypted cookies
- For server-side cookies in Flask, an extension is needed
  - E.g., https://pythonhosted.org/Flask-Session/
  - It works exactly the same way as the native Flask sessions, from the application's point of view

Sessions in Flask

- The server signs the cookie cryptographically. For this, it needs a secret key.

```
app.secret_key = "any random string"
```

  - You can generate a secret key, e.g., using a random generator

```
import os
os.urandom(24)  # copy-paste this output
```

- By default the session will be deleted when the user closes the browser. Can be set to permanent:

```
session.permanent = True
```

  - It will be set according to the config parameter **`permanent_session_lifetime`** (default: 31 days)

Sessions in Flask

- Reading a session variable

```
counter = session.get("key", None)
```

- Setting a session variable

```
session["key"] = value
```

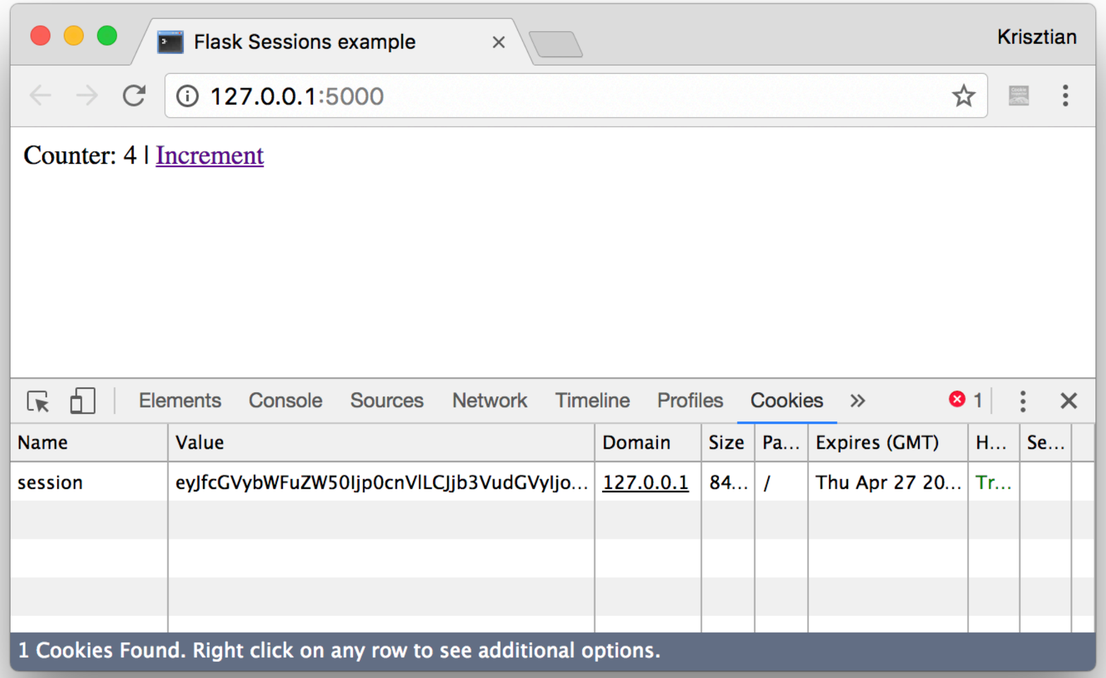
- Deleting a session variable

```
session.pop("key")
```

Example

examples/python/flask/7_sessions/app.py

- Incrementing a counter that is stored in a session



Exercise #2

github.com/dat310-2025/info/tree/master/
exercises/python/flask4

Resources

- Flask Cookies
  https://flask.palletsprojects.com/en/1.1.x/quickstart/#cookies
- Flask Session
  https://flask.palletsprojects.com/en/1.1.x/quickstart/#sessions
- W3School: Cookies in JS
  https://www.w3schools.com/js/js_cookies.asp
- Mozilla Cookies in JS
  https://developer.mozilla.org/en-US/docs/Web/API/Document/