06 – Data Management

//// Design of Distributed Systems
//

Dipl.-Inf. Michael Krug

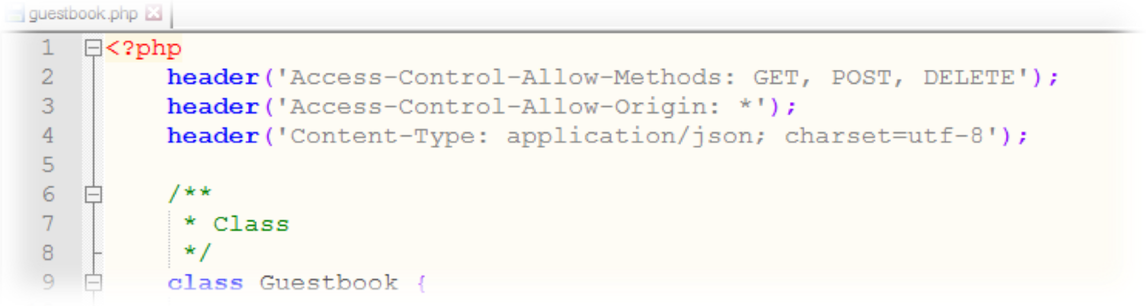
VSR.*Informatik*.TU-Chemnitz.*de*

1 Repetition

Follow-Up:

Why can you access the guestbook API on a different domain using the XMLHTTPRequest?

# Cross-Origin Resource Sharing



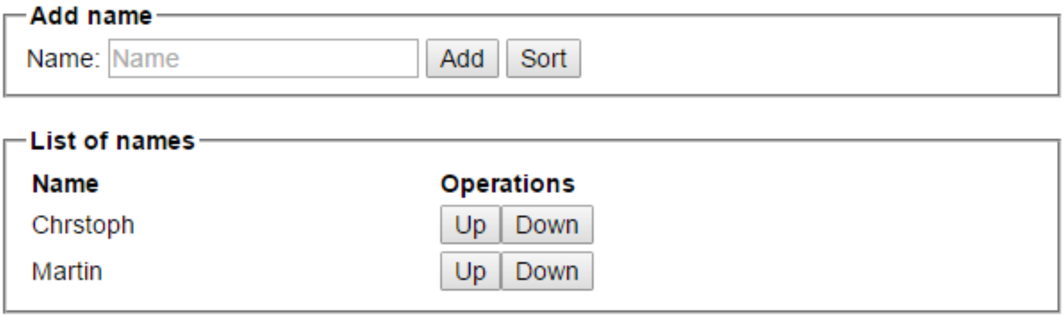
```php
<?php
    header('Access-Control-Allow-Methods: GET, POST, DELETE');
    header('Access-Control-Allow-Origin: *');
    header('Content-Type: application/json; charset=utf-8');

    /**
     * Class
     */
    class Guestbook {
```

# Homework5 500 character challenge winners

0. Me :	*373 - 397*
1. Tran Bach :	492
2. Victor Flamenco :	482 (Code in body + 9)
3. Viet Hung Nguyen :	502 (Code in body + 9)
4. Aaditya Prakash :	*495 (Top row above header +X)*
5. Dang Vu Nguyen Hai :	*499 (Modified HTML +X)*

Which Questions came up regarding

# Homework5 ?



Add name

Name: Name Add Sort

List of names

Name	**Operations**
Chrstoph	Up Down
Martin	Up Down

2 Data Management

So far, we have spoken about common

# Web Development Techniques

that exist to build a modern website

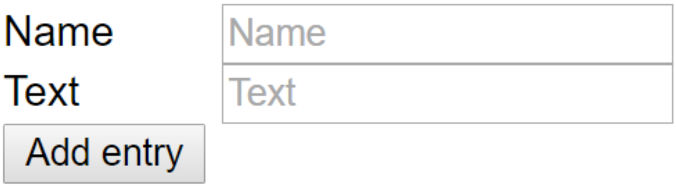
We now concentrate on aspects how to exchange

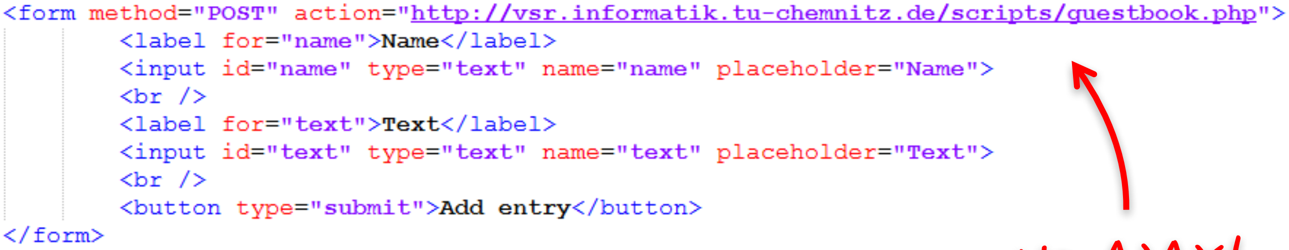
# information

in a client-server infrastructure

2.1 Forms

HTML Forms





```
<form method="POST" action="http://vsr.informatik.tu-chemnitz.de/scripts/guestbook.php">
        <label for="name">Name</label>
        <input id="name" type="text" name="name" placeholder="Name">
        <br />
        <label for="text">Text</label>
        <input id="text" type="text" name="text" placeholder="Text">
        <br />
        <button type="submit">Add entry</button>
</form>
```

No AJAX!

HTML5 Form Elements

- text input field
- multi-column textarea
- check button
- radio button
- drop-down selection list
- button

2.2 HTTP

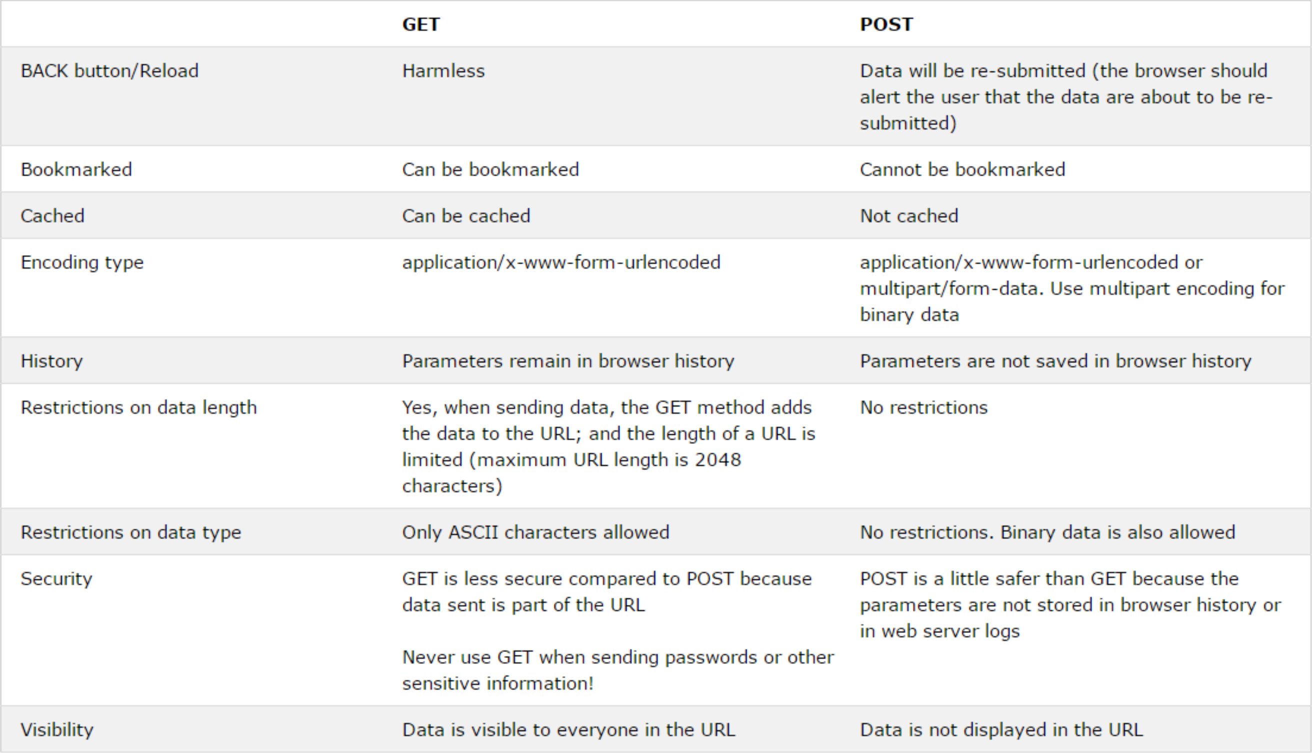
Repetition: HTTP Requests

```
GET             /scripts/guestbook.php HTTP/1.1
Host:           vsr.informatik.tu-chemnitz.de
Connection:     keep-alive
User-Agent:     Mozilla/5.0
Accept:         text/html
```

Repetition: HTTP Requests

```
POST         /scripts/guestbook.php HTTP/1.1
Host:        vsr.informatik.tu-chemnitz.de
Connection:  keep-alive
User-Agent:  Mozilla/5.0
Accept:      text/html
Content-Length: 49
Content-type: application/x-www-form-urlencoded

name=Hans&text=This%20is%20a%20Guestbook%2Entry
```

GET vs. POST



	GET	POST
BACK button/Reload	Harmless	Data will be re-submitted (the browser should alert the user that the data are about to be re-submitted)
Bookmarked	Can be bookmarked	Cannot be bookmarked
Cached	Can be cached	Not cached
Encoding type	application/x-www-form-urlencoded	application/x-www-form-urlencoded or multipart/form-data. Use multipart encoding for binary data
History	Parameters remain in browser history	Parameters are not saved in browser history
Restrictions on data length	Yes, when sending data, the GET method adds the data to the URL; and the length of a URL is limited (maximum URL length is 2048 characters)	No restrictions
Restrictions on data type	Only ASCII characters allowed	No restrictions. Binary data is also allowed
Security	GET is less secure compared to POST because data sent is part of the URL Never use GET when sending passwords or other sensitive information!	POST is a little safer than GET because the parameters are not stored in browser history or in web server logs
Visibility	Data is visible to everyone in the URL	Data is not displayed in the URL

Repetition: HTTP Response

```
HTTP/1.1 200 OK
Date:           Thu, 01 Dec 2016 12:30:24 GMT
Server:         Apache/2.2.31
Content-Length: 1322
Connection:     Keep-Alive
Keep-Alive:     timeout=3, max=100
Content-Type:   text/html
```

Repetition: HTTP Status Codes

20X Success

- *200 OK*
- *201 Created*

30x Redirection

- *301 Moved Permanently*
- *302 Found (Moved Temporarily)*
- *303 See other*

40x Client Error

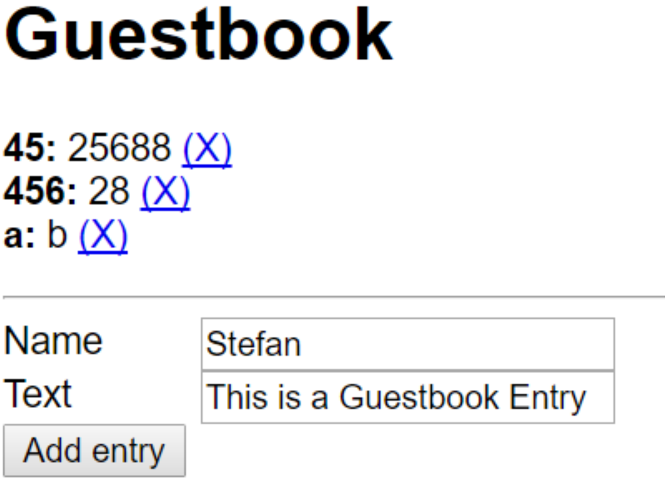
- *400 Bad Request*
- *401 Unauthorized*
- *403 Forbidden*
- *404 Not Found*

50x Server Error

- *500 Internal Server Error*

2.3 FPR

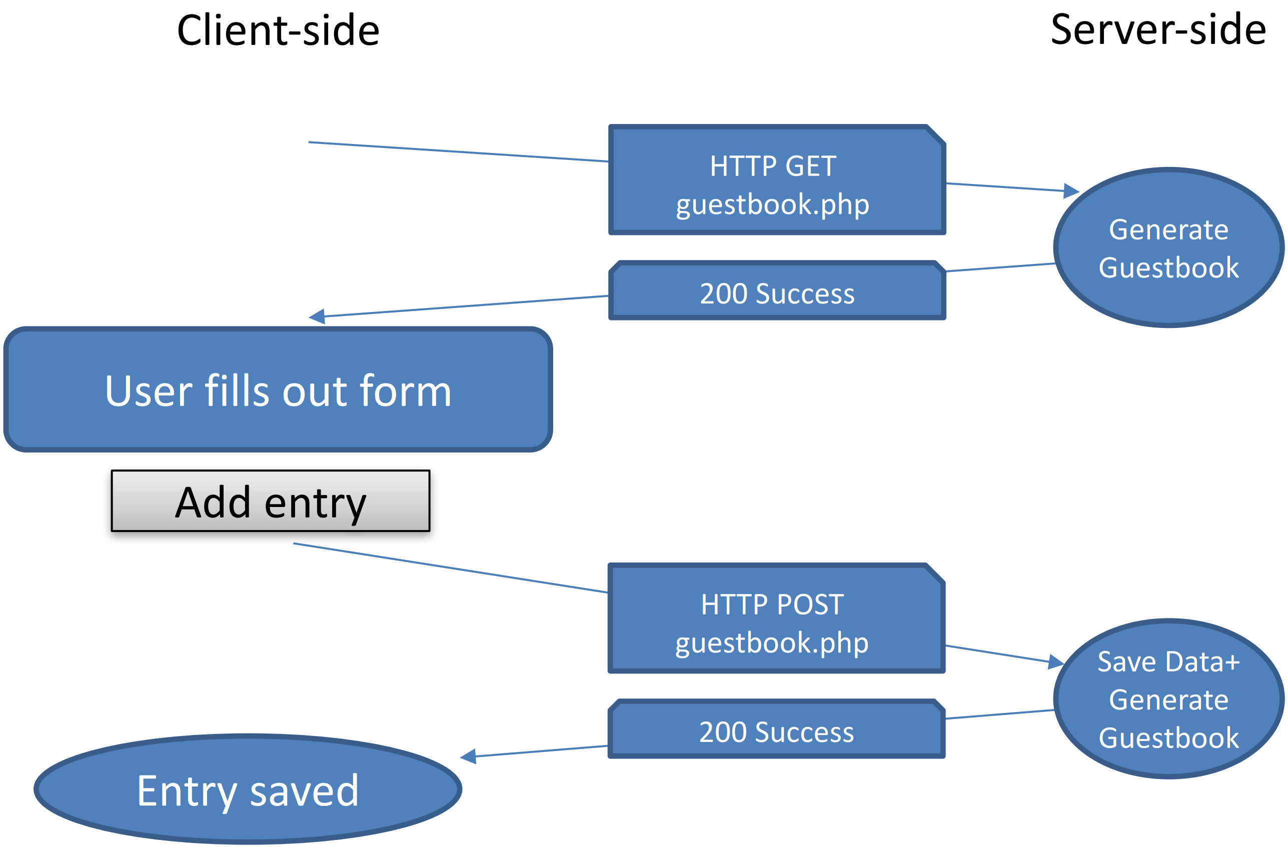
What happens if a user clicks the "Add entry" submit button?



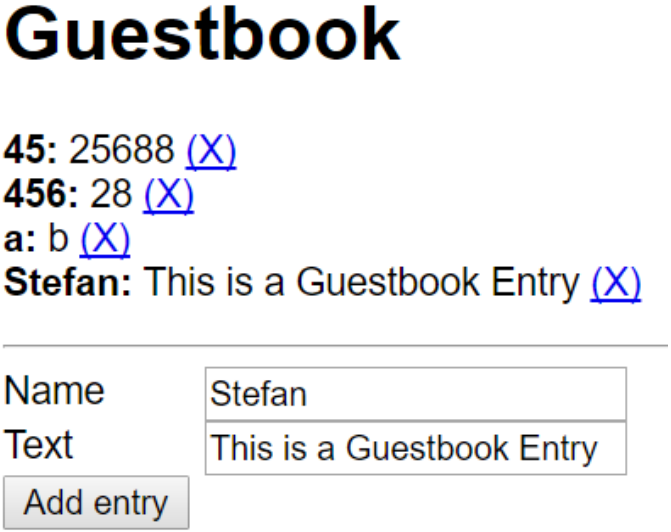
Guestbook

45: 25688 (X)
456: 28 (X)
a: b (X)

Name	Stefan
Text	This is a Guestbook Entry


Client-side
Server-side
HTTP GET guestbook.php
Generate Guestbook
200 Success
User fills out form
Add entry
HTTP POST guestbook.php
Save Data+ Generate Guestbook
200 Success
Entry saved

The data is sent to the server by using a HTTP POST Request

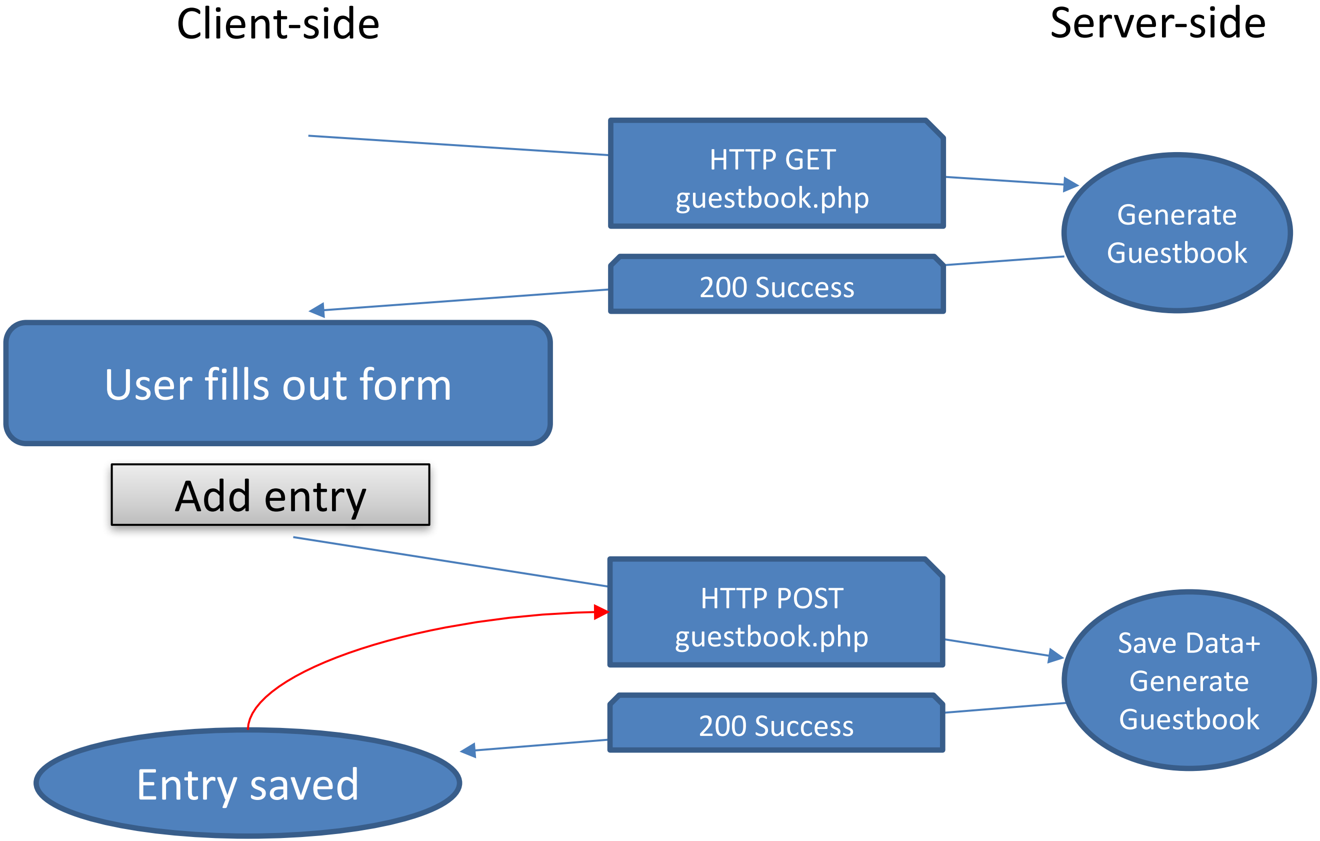


Guestbook

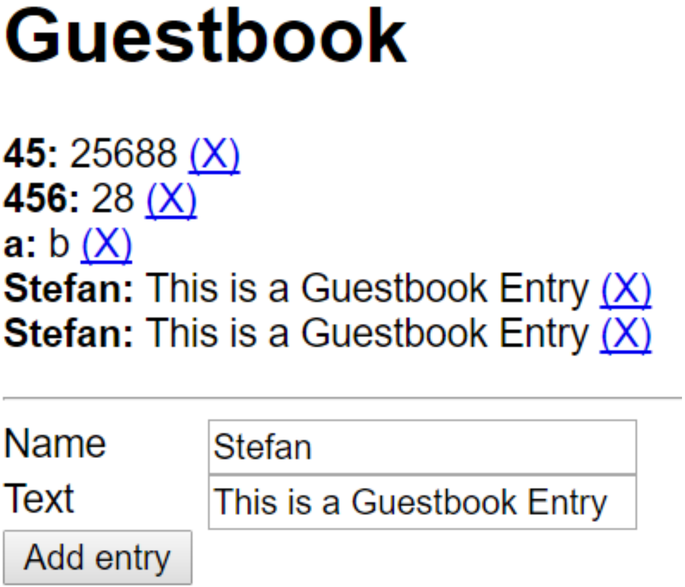
45: 25688 (X)
456: 28 (X)
a: b (X)
Stefan: This is a Guestbook Entry (X)

Name	Stefan
Text	This is a Guestbook Entry

What happens, if the user now refreshs the page, e.g. by clicking the refresh button of the browser?


Client-side
Server-side
HTTP GET guestbook.php
Generate Guestbook
200 Success
User fills out form
Add entry
HTTP POST guestbook.php
Save Data+ Generate Guestbook
200 Success
Entry saved

The last request is repeated …

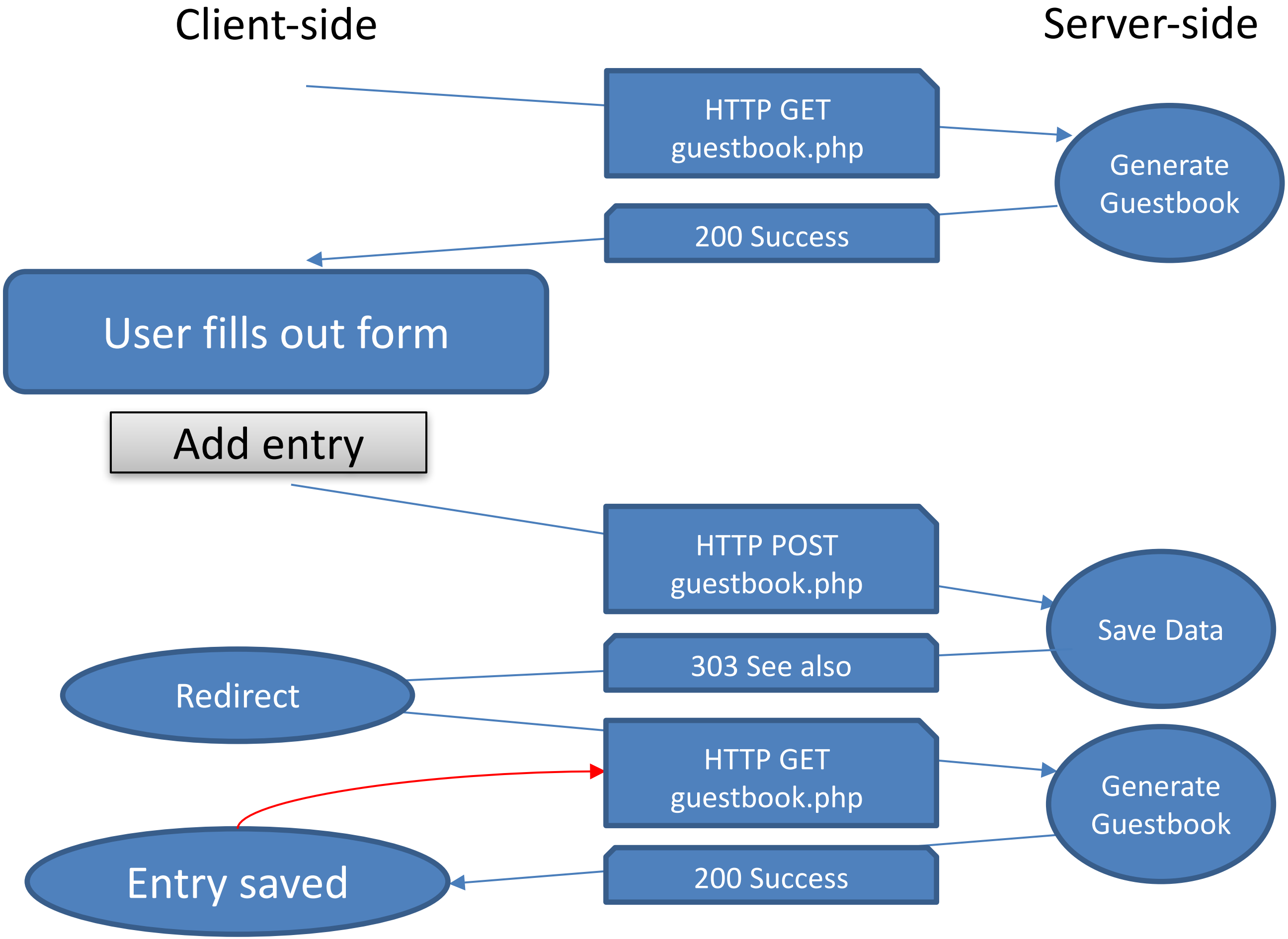


Guestbook

45: 25688 (X)
456: 28 (X)
a: b (X)
Stefan: This is a Guestbook Entry (X)
Stefan: This is a Guestbook Entry (X)

Name	Stefan
Text	This is a Guestbook Entry

? How can we prevent this?


Client-side
Server-side
HTTP GET
guestbook.php
Generate
Guestbook
200 Success
User fills out form
Add entry
HTTP POST
guestbook.php
Save Data
303 See also
Redirect
HTTP GET
guestbook.php
Generate
Guestbook
200 Success
Entry saved

This pattern is called

# Form-Post-Redirect (FPR)

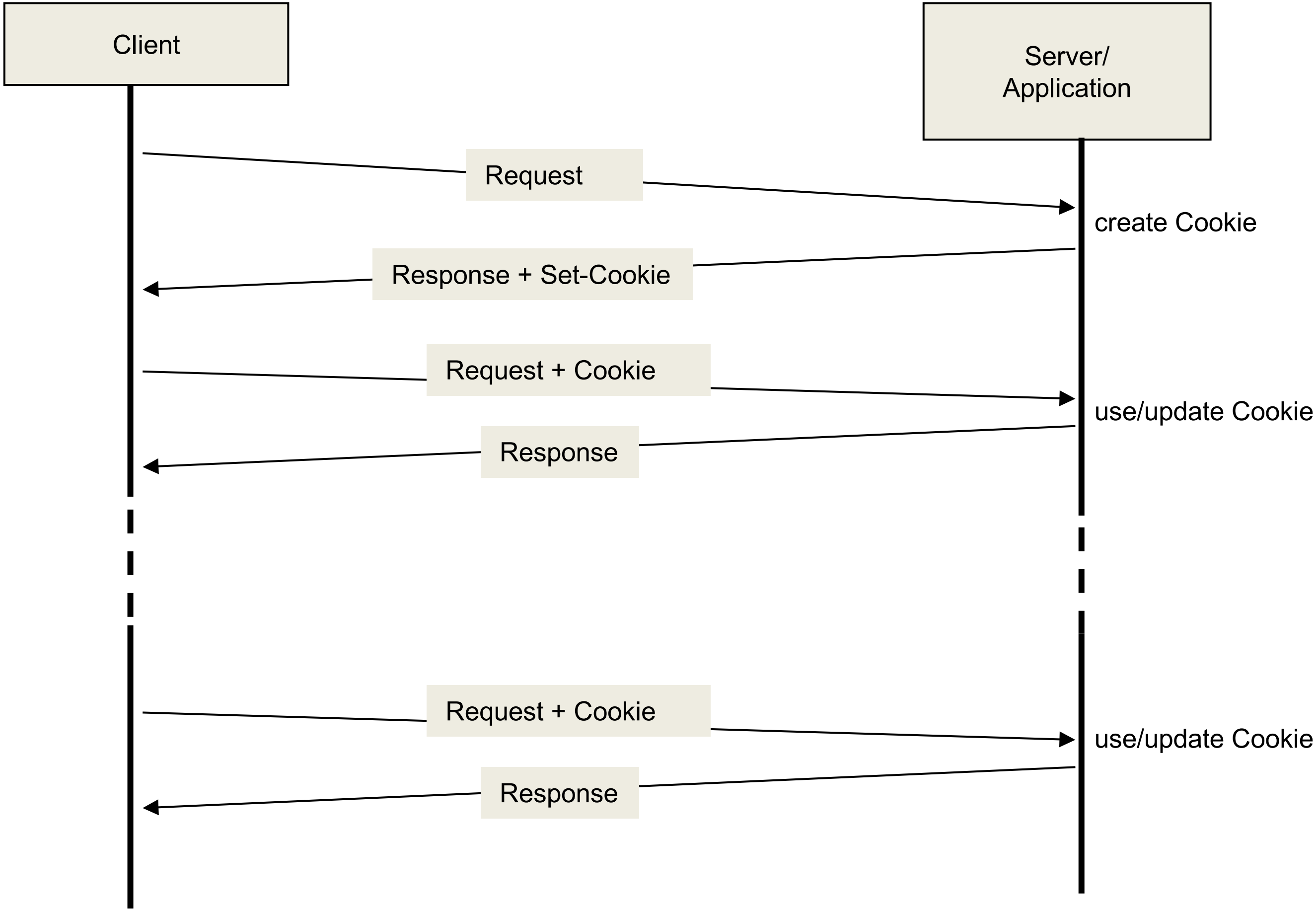
2.4 Cookies

A HTTP Cookie

- is a small piece of textual information stored by the web browser on the computer of the user
- it extends the stateless HTTP protocol by means of keeping stateful information

In Detail:

- mechanism to store a small amount of data (up to 4KB) at the client [RFC6265]
- associated with a specific web site
- sent in the HTTP header
- sent with each HTTP request
- can last for only one session (until browser is closed) or persist across sessions
- expires some time in the future


Client
Server/ Application
Request
create Cookie
Response + Set-Cookie
Request + Cookie
use/update Cookie
Response
Request + Cookie
use/update Cookie
Response

Creation of a Cookie

```
HTTP/1.1 200 OK
Date:            Thu, 01 Dec 2016 12:30:24 GMT
Server:          Apache/2.2.31
Content-Length: 1322
Connection:     Keep-Alive
Keep-Alive:     timeout=3, max=100
Content-Type:   text/html
Set-Cookie:      lastUser=Stefan;
      expires=Tue, 29 Dec 2016 19:30:00 GMT;
      Max-Age=2592000;
      Path=/scripts/guestbook.php
```

Cookie Data in a HTTP Request

```
GET           /scripts/guestbook.php HTTP/1.1
Host:         vsr.informatik.tu-chemnitz.de
Connection:   keep-alive
User-Agent:   Mozilla/5.0
Accept:       text/html
Cookie:       lastUser=Stefan
```

Application areas

- Session management (usually supported by *Session* objects in programming languages)
- Personalization
- Tracking

VSR

Thank You!

Michael.Krug@informatik.tu-chemnitz.de

VSR.*Informatik*.TU-Chemnitz.*de*