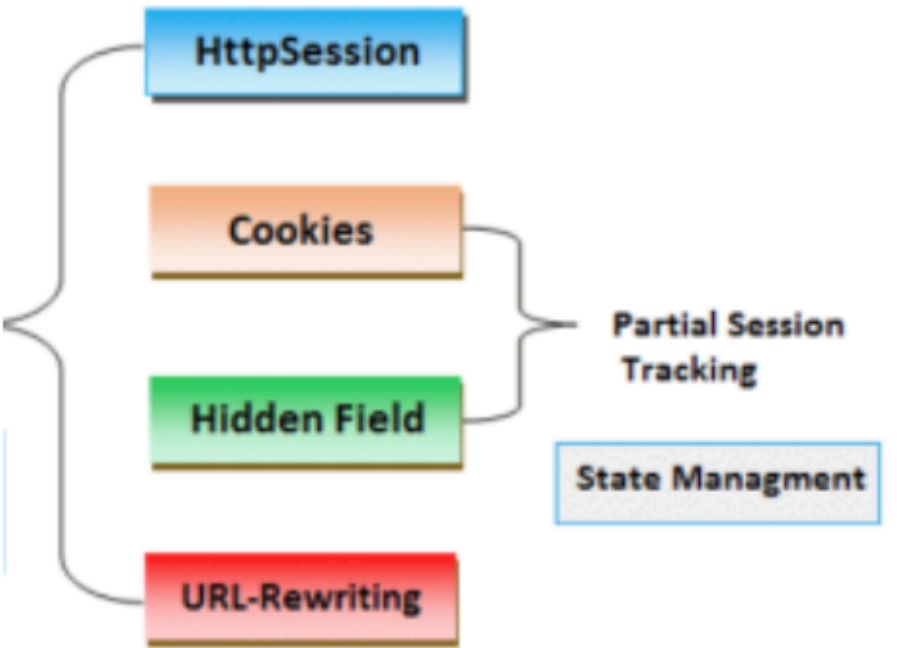


Sessions

Web Development

Sessions



HttpSession

Cookies

Hidden Field

URL-Rewriting

Partial Session Tracking

State Management

PLAY VIDEO

Sessions enable us to track a user, enabling login, logout and also to follow the user from page to page in an application

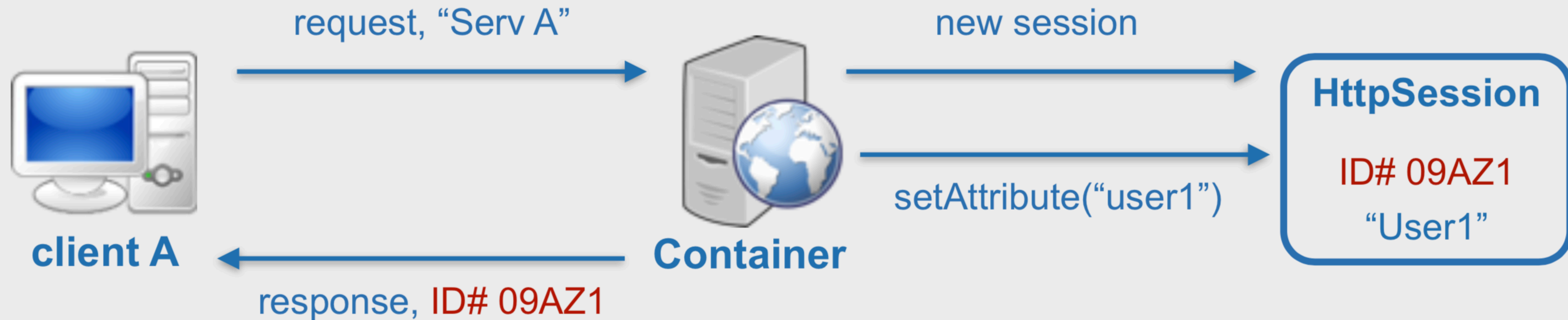
How to Make an Application out of a Web Page?

- On the internet, a web page is a web page is a web page...
 - If you surf from ./page1.html to ./page2.html these are two unique requests.
 - The server doesn't know anything about the fact that both pages are visited by the same user.
- Sessions are the technique used to logically group several requests into a "group" (called a session)
 - If you start a session, the server will know that it's still the same user who surfed from ./page1.html to ./page2.html

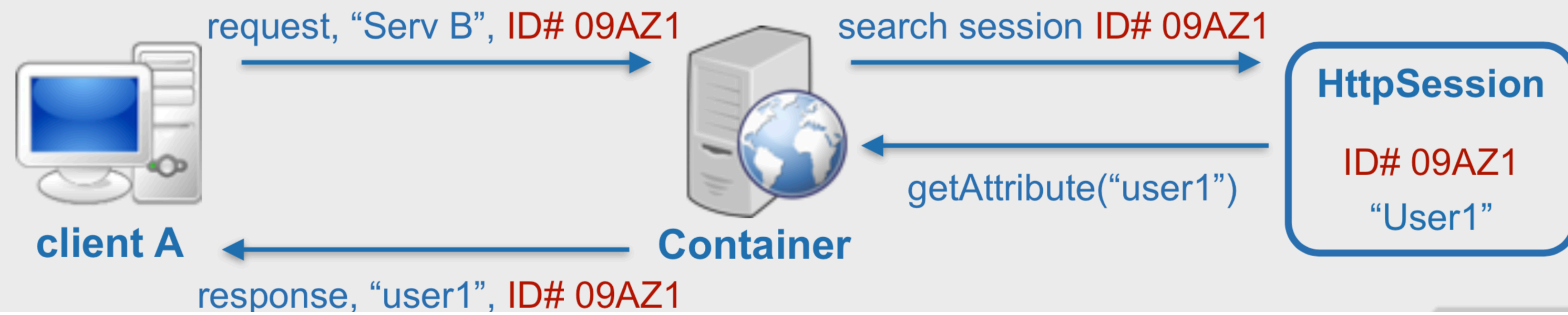


Session Tracking

First Request

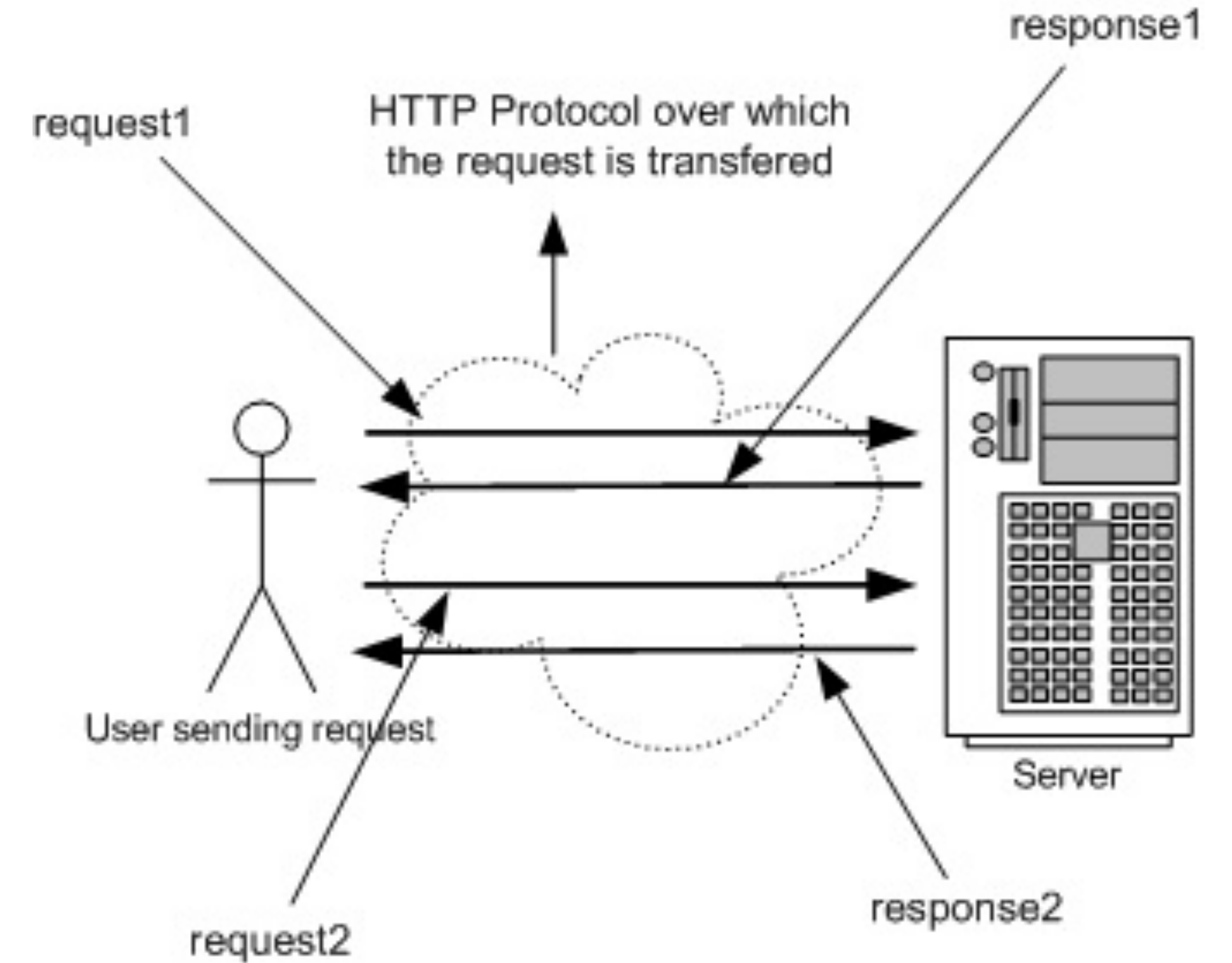


Subsequent Request



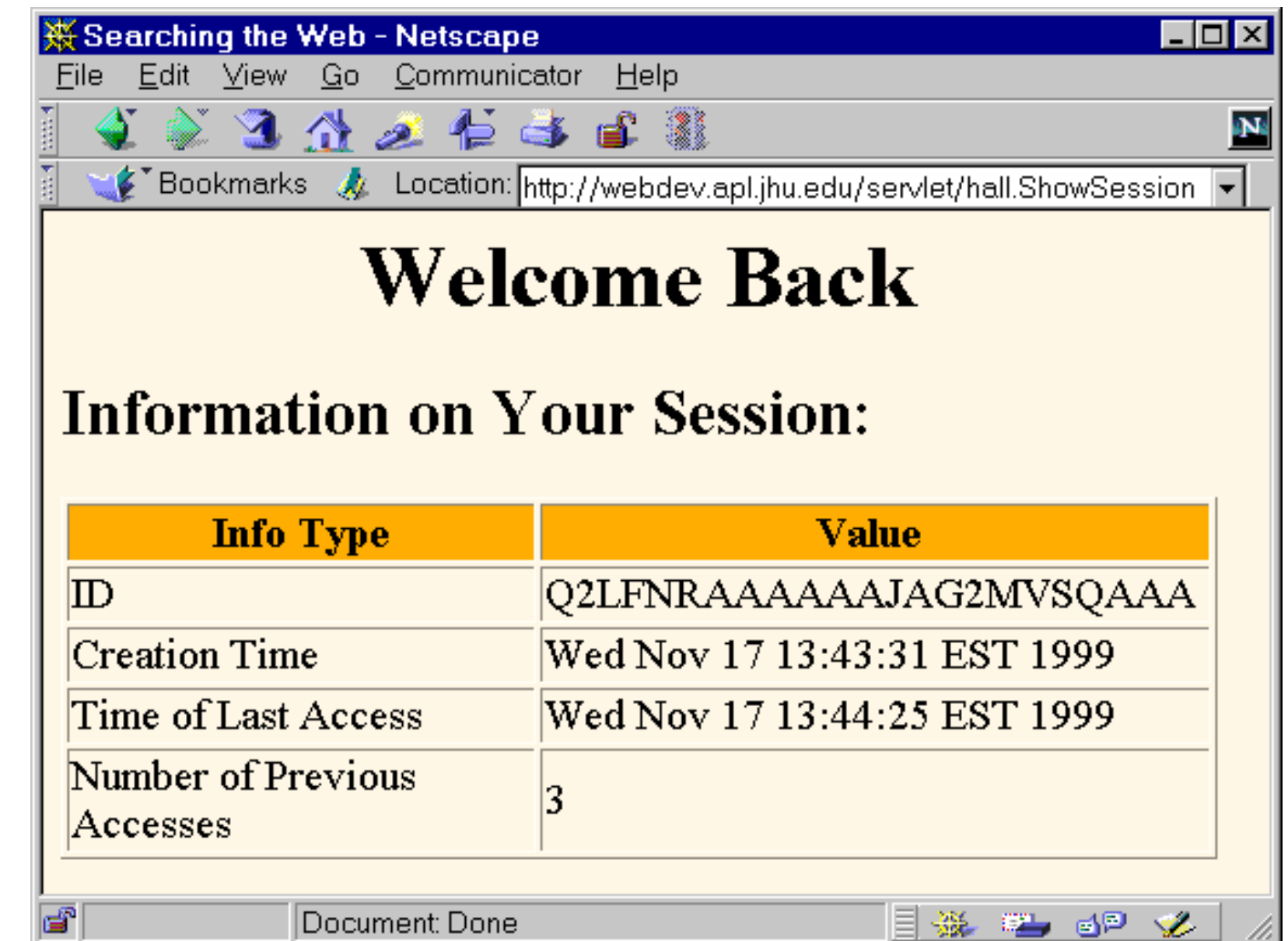
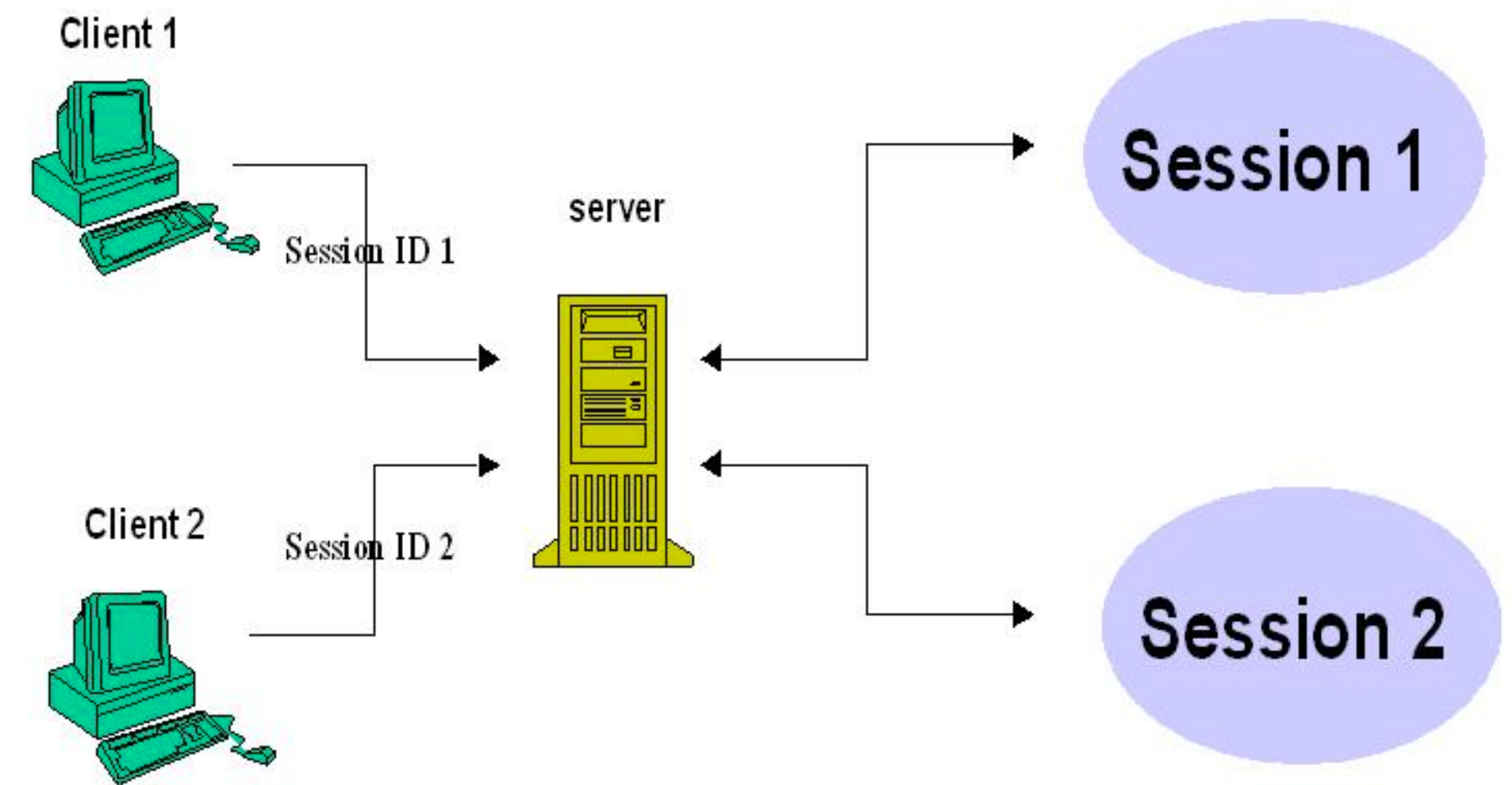
Sessions

- HTTP itself is “stateless”
 - no state stored on the server between requests from the same client
- but many web apps are stateful
 - necessary to connect requests from the same user / browser / browser-window, e.g. shopping cart, appointments calendar etc...
- *Session*
 - multiple requests performed in a stateful context
- *Session tracking*
 - technique that allows sessions in stateless environments



- User surfs to http://demo.com
 - Server (on 1st request / if no sessionID stored on client)
 - generates unique session id, which is mapped to ...
 - ... a session-object
 - stored in memory (lost on shutdown), in a file or in database
 - can contain anything (list of articles, game state, counters, ...)
 - Session id is added to the response
- from now on:
 - each subsequent request from the same user (browser) must contain the session id ...
 - ... which is used by the server to map to the session-object
- No data gets stored on the client, except SessionID

Session Tracking



Session Tracking Techniques

Cookie

URL Rewriting

Hidden Form Field

JSON Web Token (JWT)

Cookies

Cookies

First Response



client A



Http Response

```
HTTP/1.1 200 OK  
Location: http://www.abcd.com/login  
Set-Cookie: JSESSIONID=09AZ1  
Domain=.abcd.com;path=/;HttpOnly  
.....
```



Container



Subsequent Requests



client A



Http Request

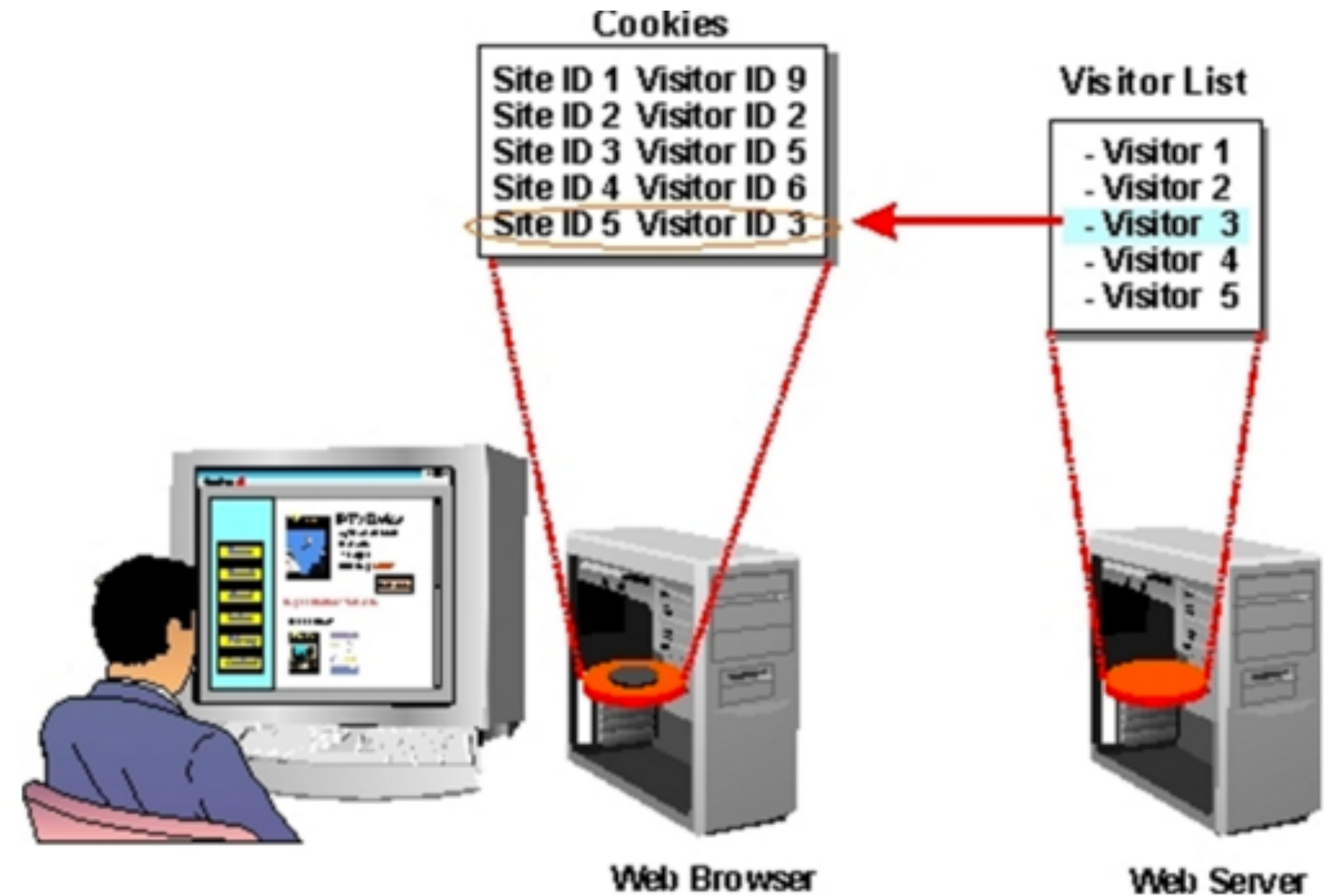
```
POST/login.do HTTP/1.1  
Host: www.abcd.com  
Cookie: JSESSIONID=09AZ1  
.....
```



Container

Cookies

1. Server creates a cookie with session-id on first request
2. Server maps id to a new user-specific session object
3. The session-id is sent to the client with the first response
4. ..and automatically added by the browser on each further request (to the same address/domain/...)
5. Server receives request + cookie with session-id
6. Server maps session-id to session-object



donation-web cookie (in browser)

The screenshot shows a web browser window with the URL `localhost:9000/dashboard`. The page displays a 'homer simpson's Todo List' with two items: 'Make tea' and 'Go for snooze', each with a 'Delete' button. Below the list is a form with a 'Title' label and an input field containing the text 'Title'.

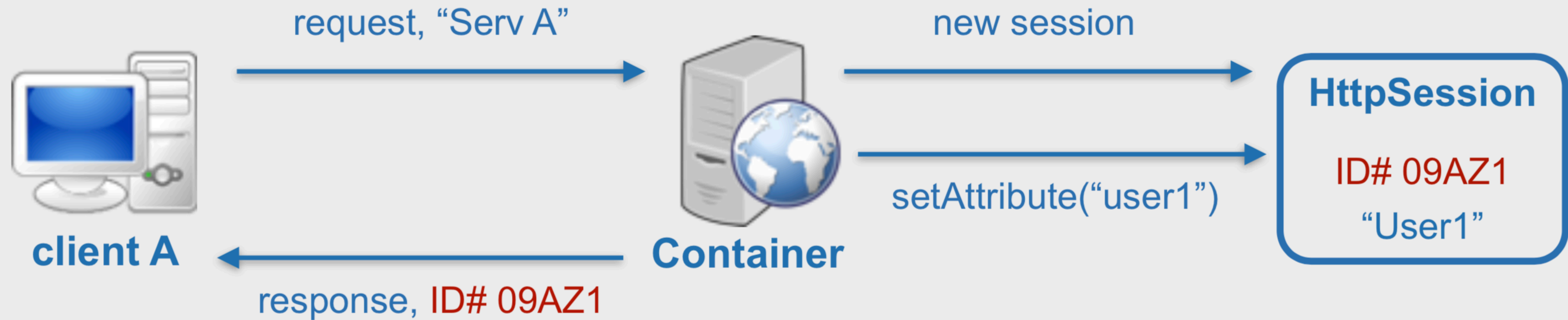
The Chrome DevTools Application tab is open, showing the 'Cookies' section for the domain `http://localhost:9000`. The following table represents the data shown in the DevTools interface:

Name	Value	Do...	Path	Exp...	Size	HTTP	Secure	SameS...
PLAY_SESSION	1813c9cb72f6f7b3954da2ce827b1fdee94972cc-logged_in_Mem...	loc...	/	Ses...	73			

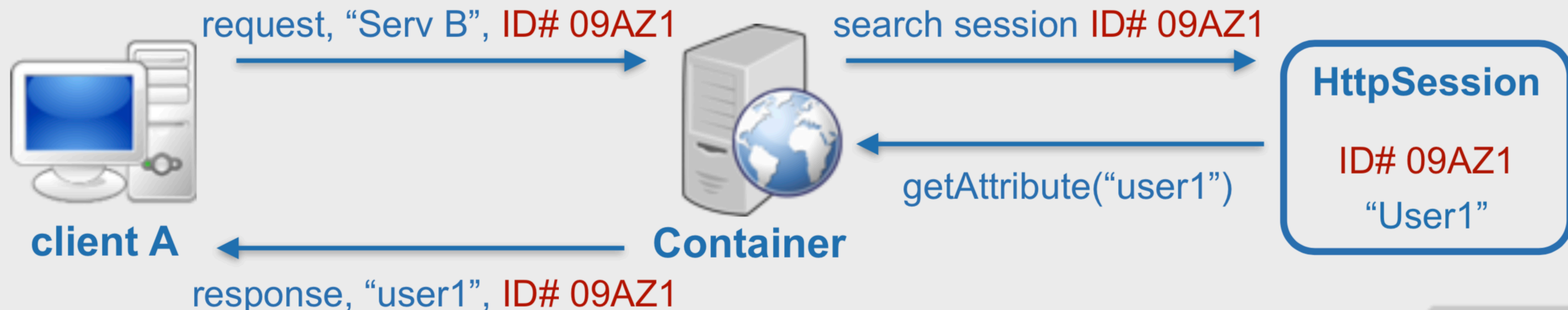
URL Rewriting

URL Rewrite

First Request



Subsequent Request



URL Rewrite

- Server adds the session-id to all links the user can follow
 - `http://server/myhome`
- is changed to
 - `http://server/myhome?sessionid=123`
- session-id must be dynamically added
 - functionality usually offered by scripting frameworks

Hidden Form Field

Hidden Form Fields

- In HTML, we can define "hidden" fields in a form
 - `<input type="hidden" name="sessionid" value="123">`
- These fields are not visible and cannot be changed by the client
- Usage:
 - server creates a session-object for each client and generates a unique ID
 - When HTML documents are created and sent back, the hidden form field is automatically generated containing the actual ID
 - Upon form submit, the session ID is automatically sent back to the server
 - The server can associate this call with an already existing session

Hidden Form Filed Example

Todo List Dashboard Ab

homer simpson's Todo List

Todo

Make tea Delete

Go for snooze Delete

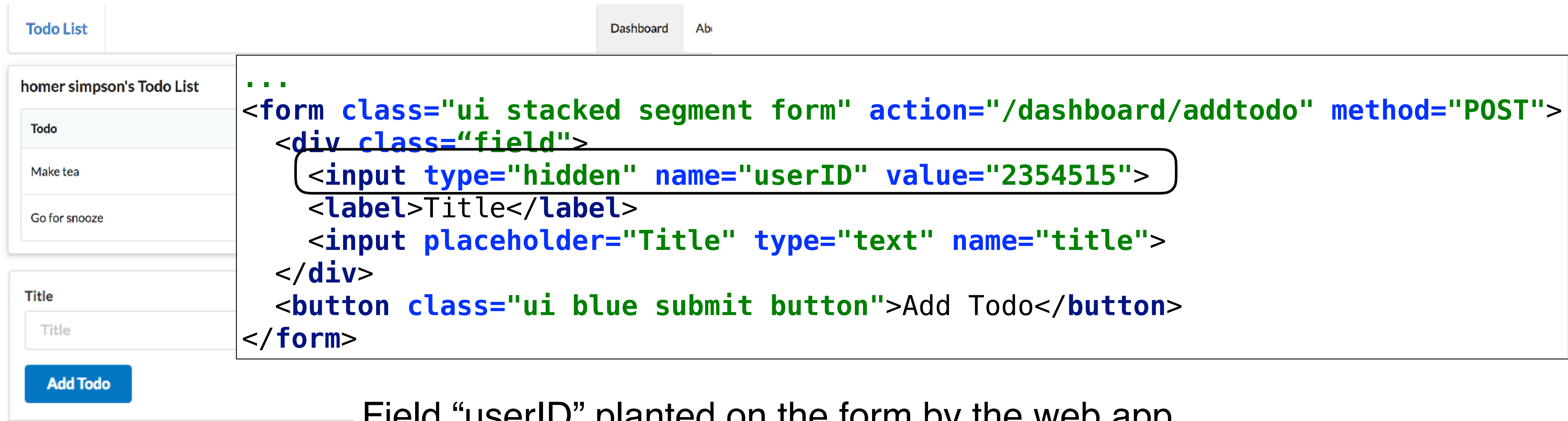
Title

Title

Add Todo

```
...  
<form class="ui stacked segment form" action="/dashboard/addtodo" method="POST">  
  <div class="field">  
    <input type="hidden" name="userID" value="2354515">  
    <label>Title</label>  
    <input placeholder="Title" type="text" name="title">  
  </div>  
  <button class="ui blue submit button">Add Todo</button>  
</form>
```


Hidden Form Field Example



```
...  
<form class="ui stacked segment form" action="/dashboard/addtodo" method="POST">  
  <div class="field">  
    <input type="hidden" name="userID" value="2354515">  
    <label>Title</label>  
    <input placeholder="Title" type="text" name="title">  
  </div>  
  <button class="ui blue submit button">Add Todo</button>  
</form>
```

Field "userID" planted on the form by the web app
- and set to the ID of the specific user

The application can use this ID to locate the user
in the database, and ensure the new data goes to
the correct database entry

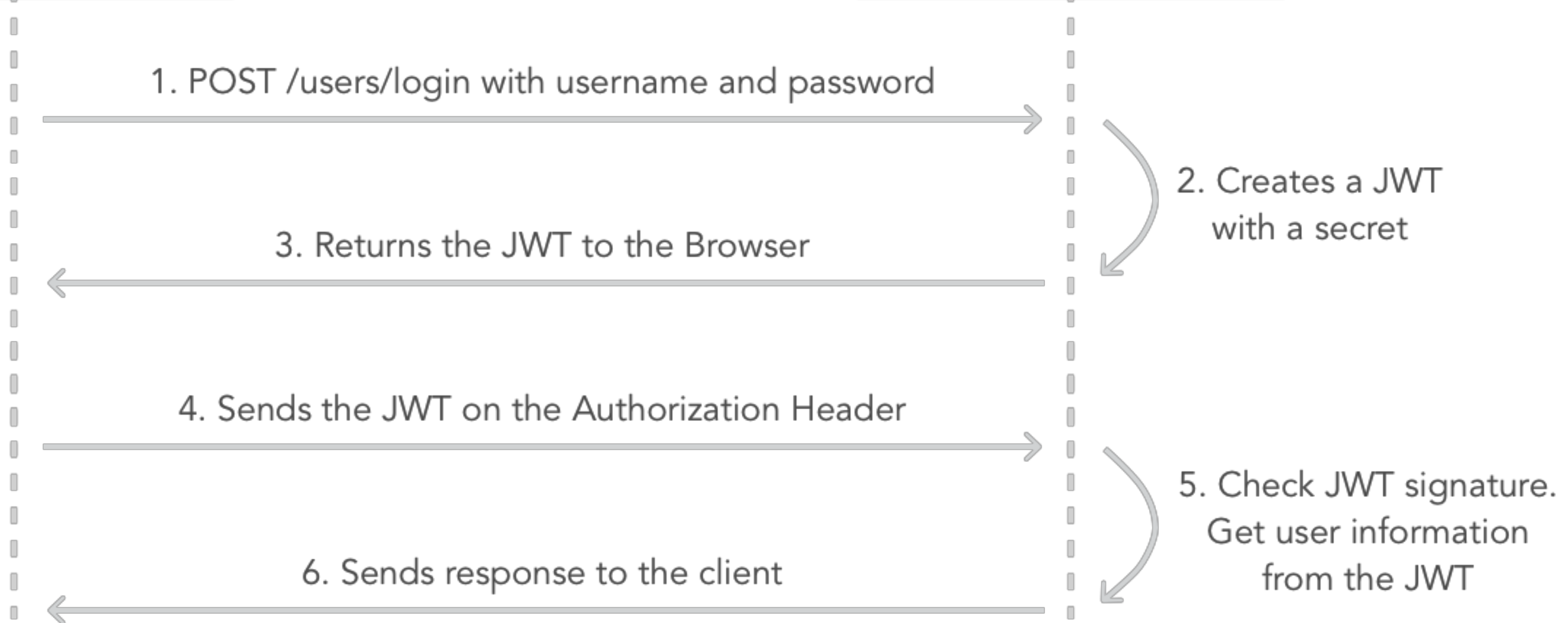
JSON Web Token

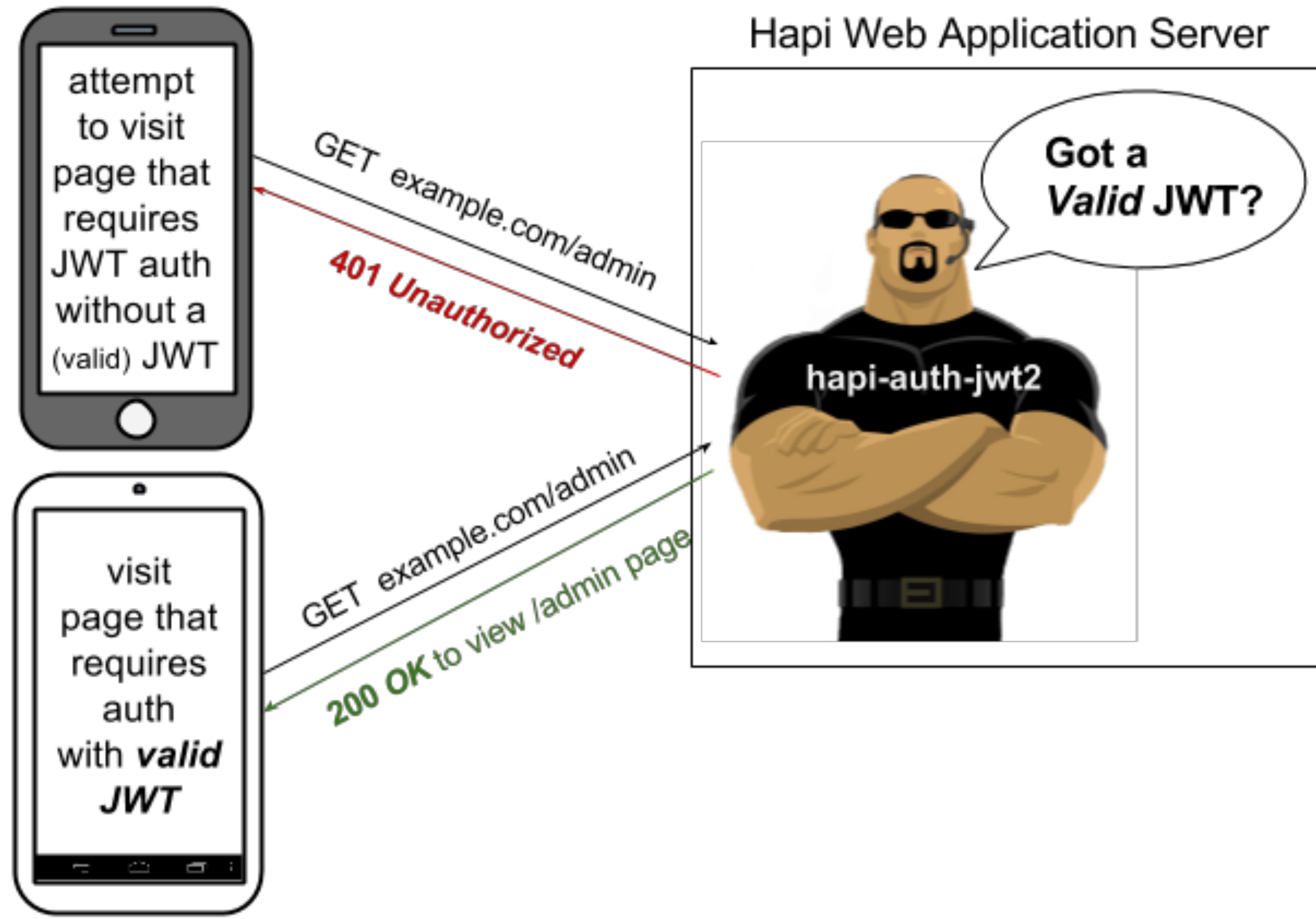
Json Web Token

- An open standard that defines a compact and self-contained way for securely transmitting information between parties as a JSON object.
- **Compact:** Because of its smaller size, JWTs can be sent through an URL, POST parameter, or inside an HTTP header.
- **Self-contained:** The payload contains all the required information about the user, avoiding the need to query the database more than once.
- **Authentication:** Once the user is logged in, each subsequent request will include the JWT, allowing the user to access routes, services, and resources that are permitted with that token.
- **Information Exchange:** JSON Web Tokens are a good way of securely transmitting information between parties, because they can be signed.

Browser

Server





Web Frameworks

- Cookies generally preferred.
- However, framework may try to ‘abstract away’ specific session management technology, and deliver simpler abstraction to the programmer
- Framework may in fact be able to switch between different techniques depending on circumstances.
- Play only supports Cookies