howest
hogeschool

Laravel: Eloquent (DB) continued, Validation, Cookies and Sessions

Web, Mobile and Security

Frédéric Vlummens

Agenda

- Eloquent: recap
  - Selecting all records
  - Adding a record
- Eloquent use cases
  - Selecting a specific record
  - Updating a specific record
  - Deleting a specific record
- Validation
- Cookies
- Sessions

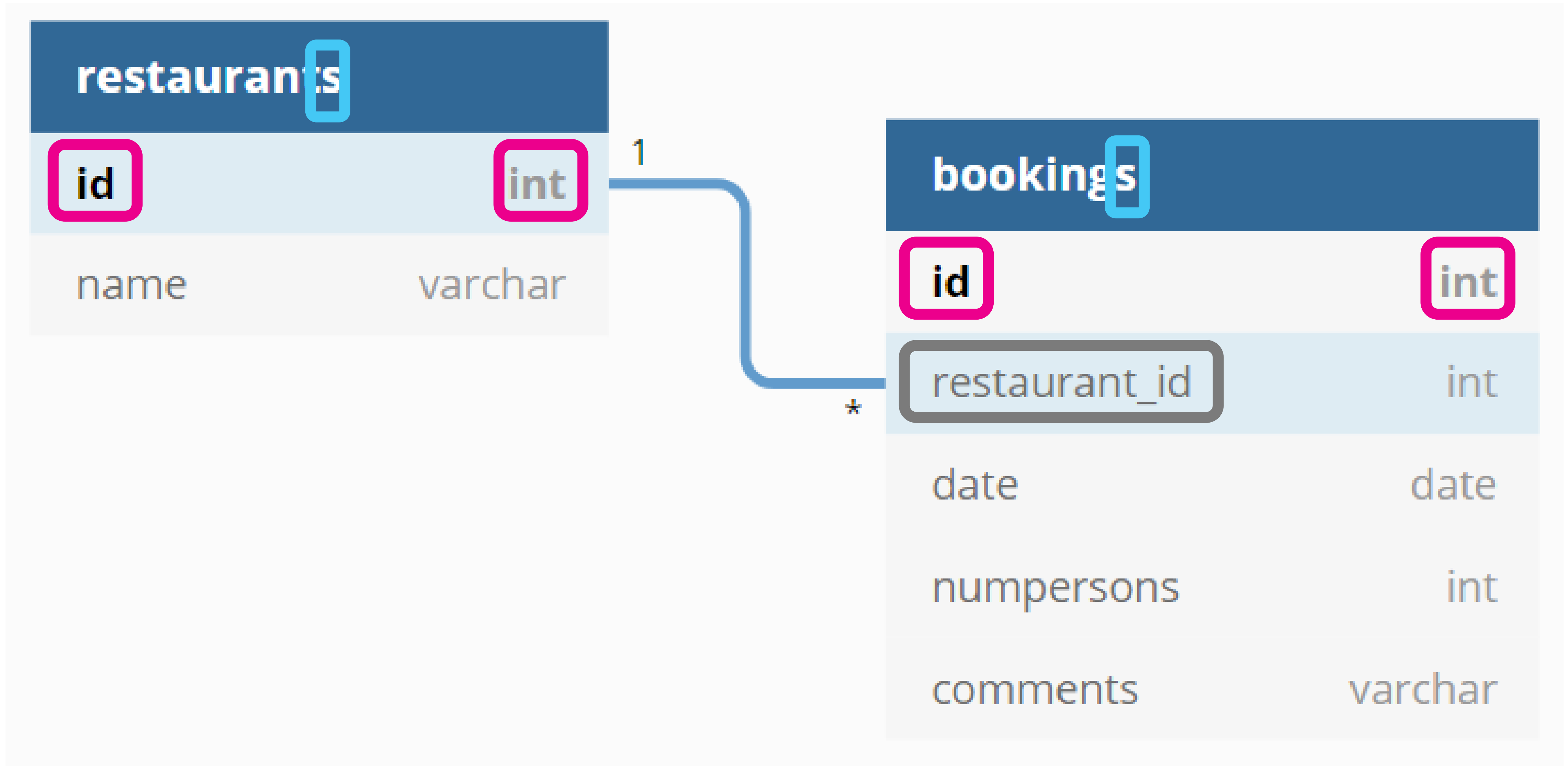
howest
hogeschool

Eloquent: recap

Eloquent: recap

- Eloquent = ORM (Object-Relational Mapper)
- Database tables have corresponding models, used to interact with tables
- We do not write SQL code ourselves, but let Eloquent generate it for us
- Convention over configuration:
    - Table names are plural, corresponding Models singular
    - Each table has a PK field called id of type INTEGER AUTO_INCREMENT
    - One-to-many relationships are handled in the database by taking singular of table and suffixing foreign key field with _id
    - These are conventions: we do not need to explain Laravel the pluralization rules or primary keys
    - As long as we follow the rules, Eloquent knows what to do

Eloquent: recap



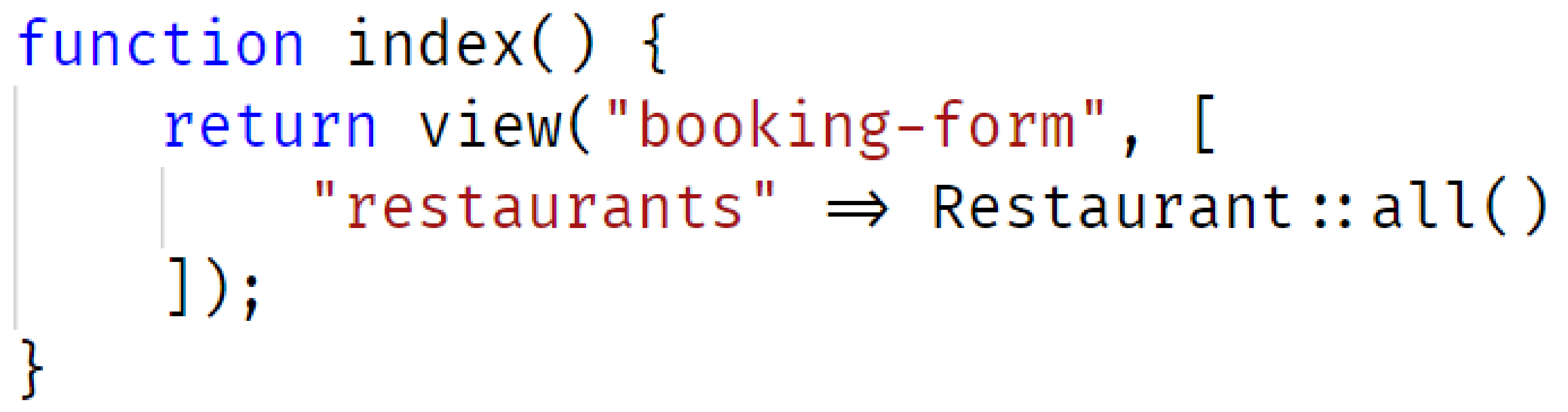
Database table names are pluralized

Primary keys are auto increment integers called id

For 1 to many relationships, the foreign key name consists of the related table name in singular, suffixed by _id

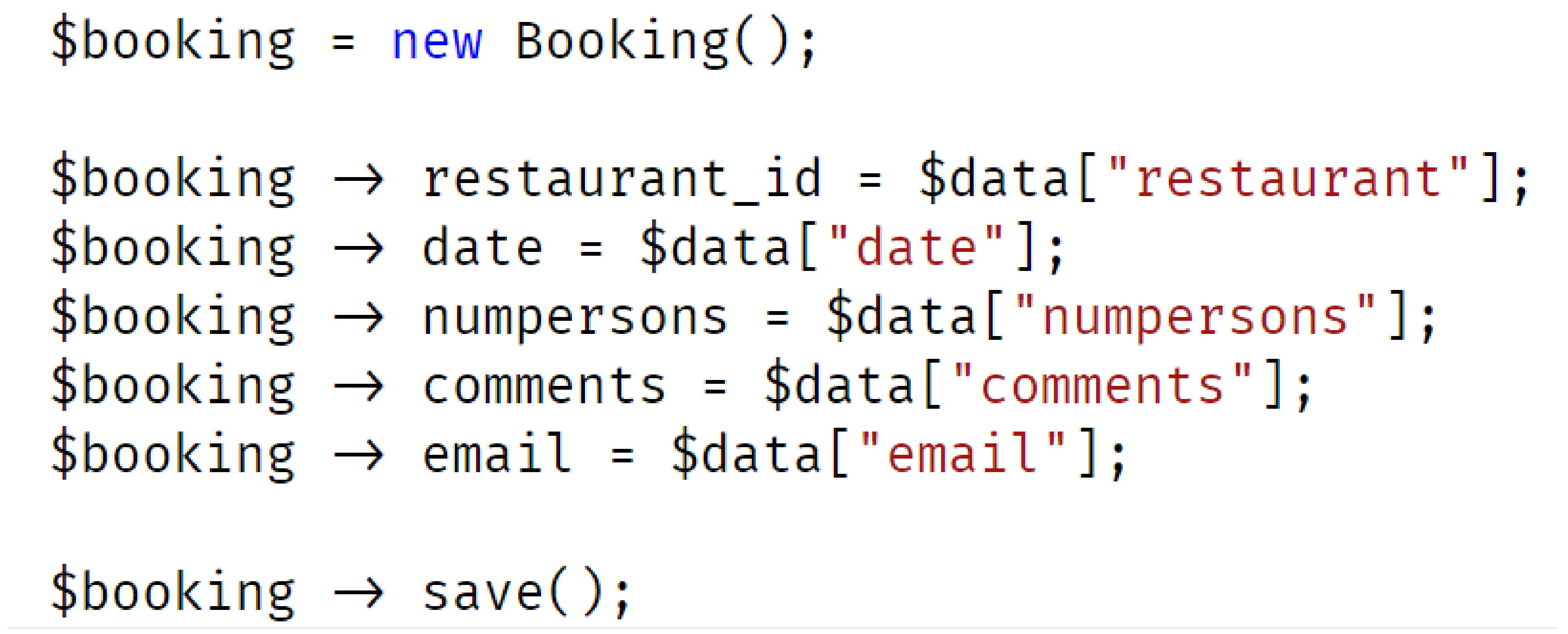
Eloquent: recap

- Selecting all records using all():



```php
function index() {
    return view("booking-form", [
        "restaurants" => Restaurant::all()
    ]);
}
```

- Adding a specific record using save():



```php
$booking = new Booking();

$booking -> restaurant_id = $data["restaurant"];
$booking -> date = $data["date"];
$booking -> numpersons = $data["numpersons"];
$booking -> comments = $data["comments"];
$booking -> email = $data["email"];

$booking -> save();
```

Eloquent: additional use cases

Eloquent use cases

- Using the find() method, select a record based on its primary key

```
$restaurant = Restaurant::find(5); // find restaurant with id = 5
```

- Using the save() method, you can also update existing records:

```
$restaurant = Restaurant::find(2); // find restaurant with id = 2
$restaurant → name = "New name";
$restaurant → save();
```

- Using the delete() method, you can delete an existing record:

```
$restaurant = Restaurant::find(2); // find restaurant with id = 2
$restaurant → delete();
```

howest hogeschool

Validation

Validation

- Validate form input on server-side
- Do not depend on client-side validation only!
- Define rules per parameter
- Rules are combined using the | symbol
- When validation fails, user gets returned to originating view
- $errors variable can be used to display validation errors

Validation

```php
function addBooking(Request $request) {
    $data = $this -> validateBooking($request);

    // ...

function validateBooking($request) {
    $rules = [
        "restaurant" => "required",
        "numpersons" => "required|integer|min:1|max:6",
        "email" => "required|email",
        "date" => "required|date|after_or_equal:today",
        "comments" => "string|nullable"
    ];

    $data = $request -> validate($rules);

    return $data;
}
```

Keys must match name attributes of your form fields

Individual rules per parameter
See https://laravel.com/docs/master/validation

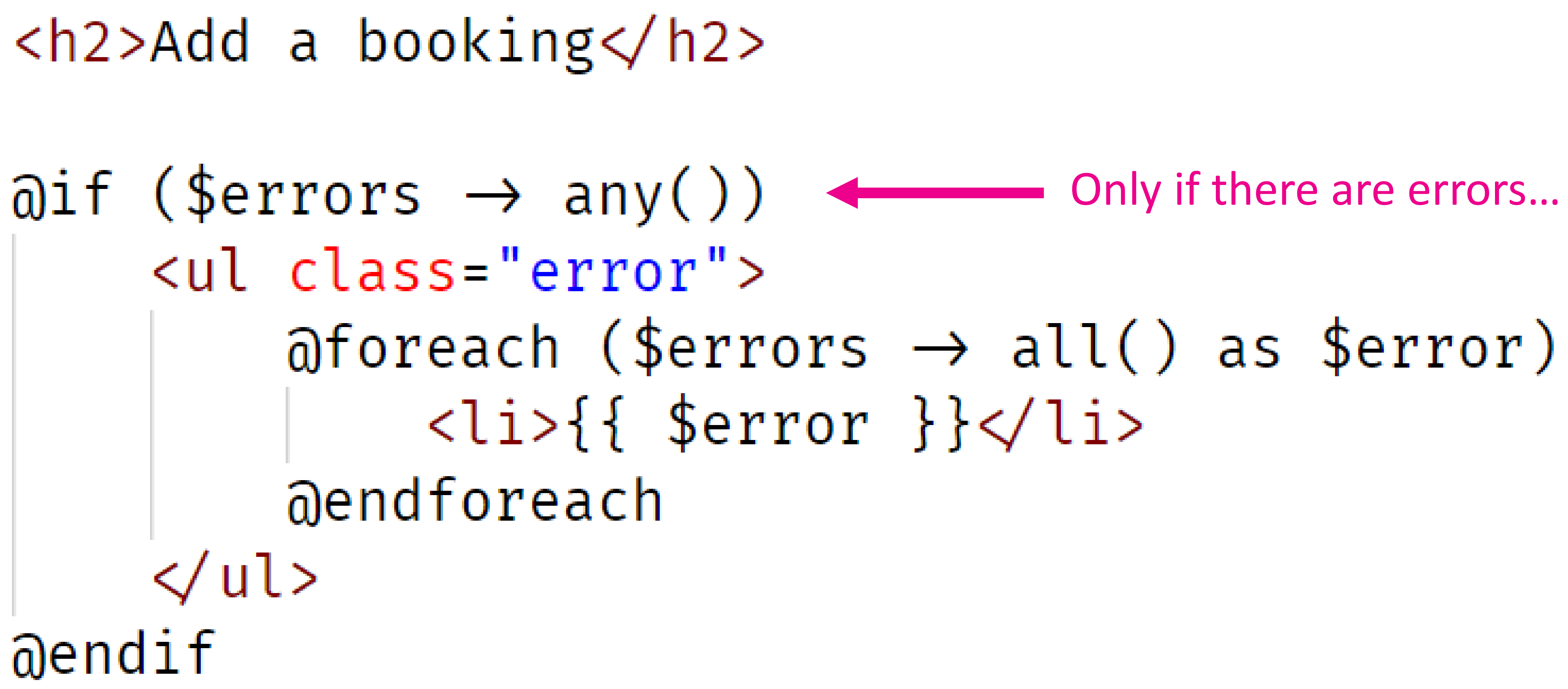
Validation

```php
function addBooking(Request $request) {
    $data = $this → validateBooking($request);

    $booking = new Booking();

    $booking → restaurant_id = $data["restaurant"];
    $booking → date = $data["date"];
    $booking → numpersons = $data["numpersons"];
    $booking → comments = $data["comments"];
    $booking → email = $data["email"];

    $booking → save();
```

validate() method returns an associative array, containing all validated values

Validation

- In Blade file:



```
<h2>Add a booking</h2>

@if ($errors → any())
    <ul class="error">
        @foreach ($errors → all() as $error)
            <li>{{ $error }}</li>
        @endforeach
    </ul>
@endif
```

Only if there are errors…

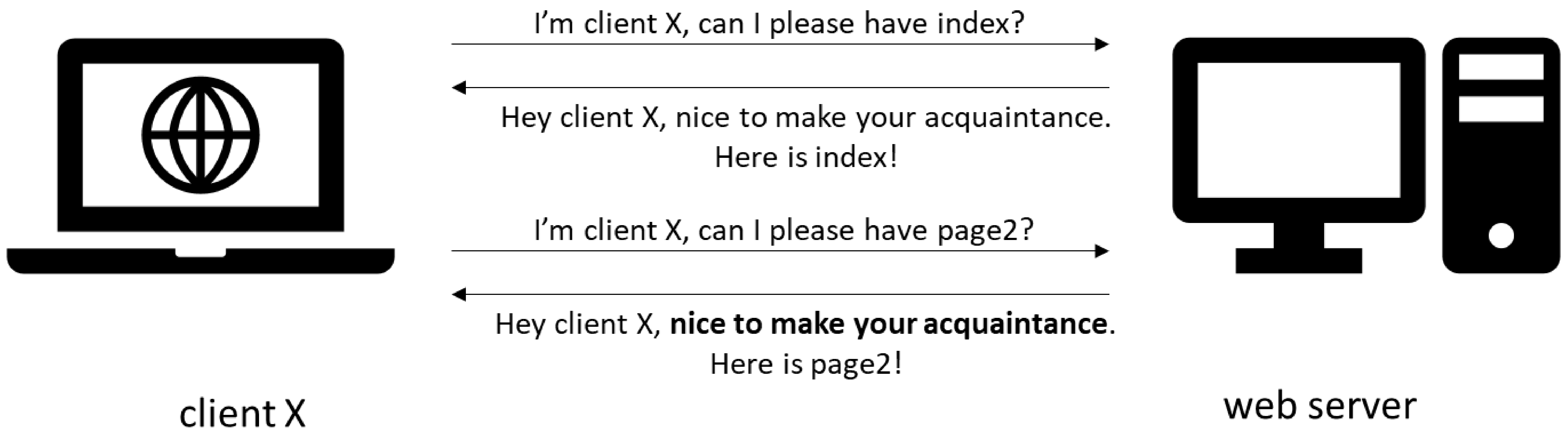
…loop over them and print them out

howest
hogeschool

Cookies

HTTP = stateless protocol

- When submitting a form, all previous data is lost
- (Except if we store in database)
- Reason: HTTP is a stateless protocol
- Each request is independent from the subsequent one


I'm client X, can I please have index?
Hey client X, nice to make your acquaintance. Here is index!
I'm client X, can I please have page2?
Hey client X, **nice to make your acquaintance**. Here is page2!
client X
web server

HTTP = stateless protocol

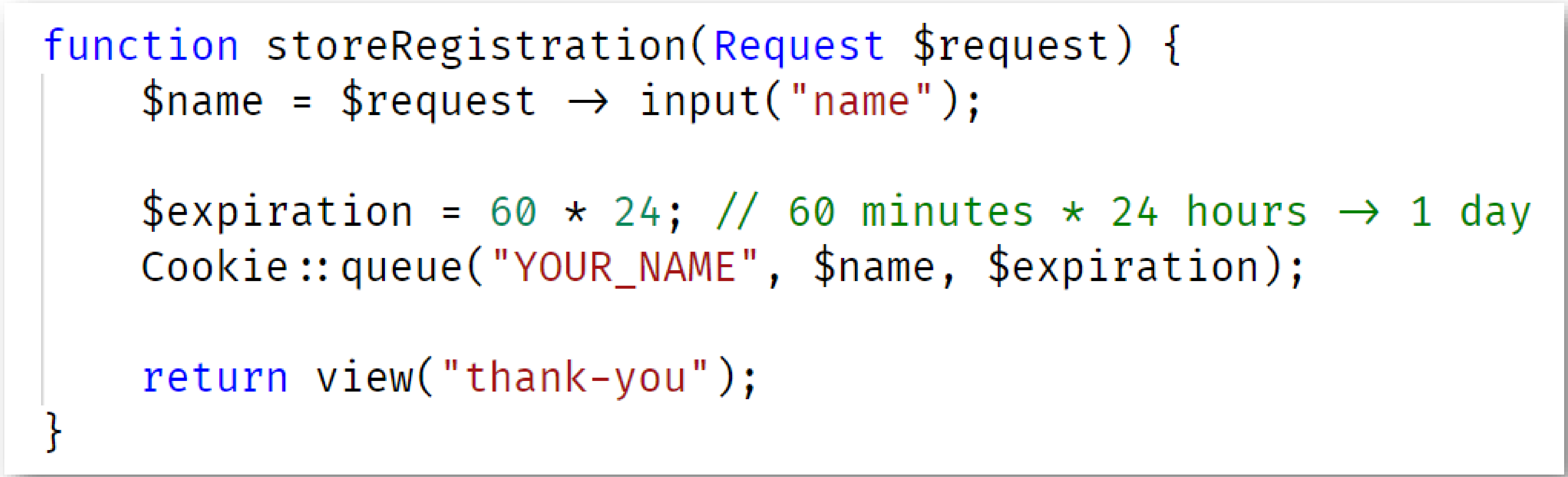
- Somehow, we must make sure our webserver code “remembers” us
- Solutions have been developed:
  - Cookies
  - Sessions

Cookie

- Small text file
- Sent from website (server) and stored by browser (client)
- Upon each subsequent request, the cookie is sent back to the server
- This way, the server “recognizes” the client from previous requests
- Circumvent the statelessness of the HTTP protocol

Cookies in Laravel

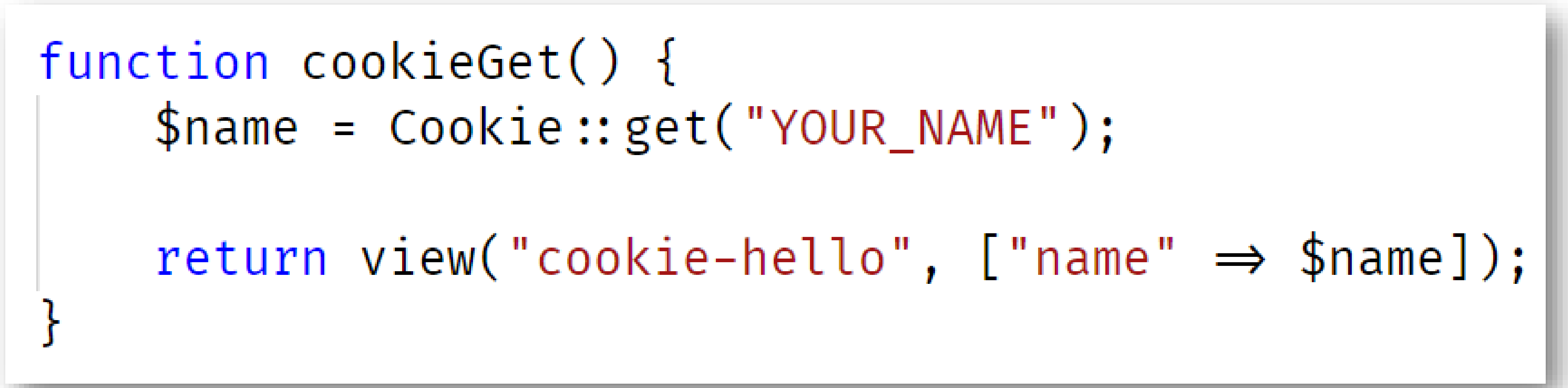
- A cookie has a **name**, **value** and **experiation time**
- We put our cookie in the queue. It will be handled by Laravel and sent back to client via response.



```
function storeRegistration(Request $request) {
    $name = $request → input("name");

    $expiration = 60 * 24; // 60 minutes * 24 hours → 1 day
    Cookie::queue("YOUR_NAME", $name, $expiration);

    return view("thank-you");
}
```

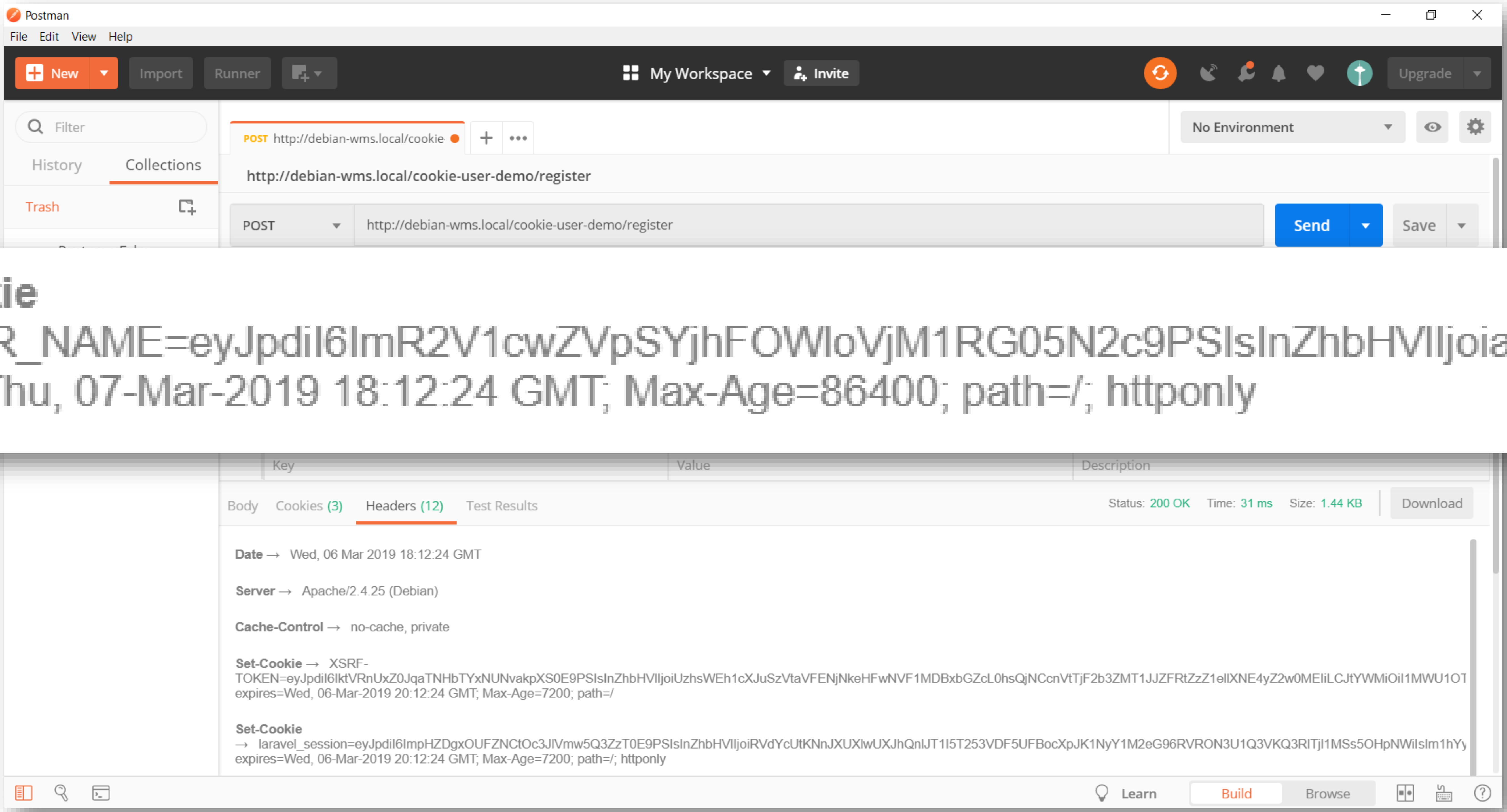
Cookies in Laravel

- Retrieving the cookie:



```
function cookieGet() {
    $name = Cookie::get("YOUR_NAME");

    return view("cookie-hello", ["name" ⇒ $name]);
}
```

Cookies: inspecting request – response using Postman



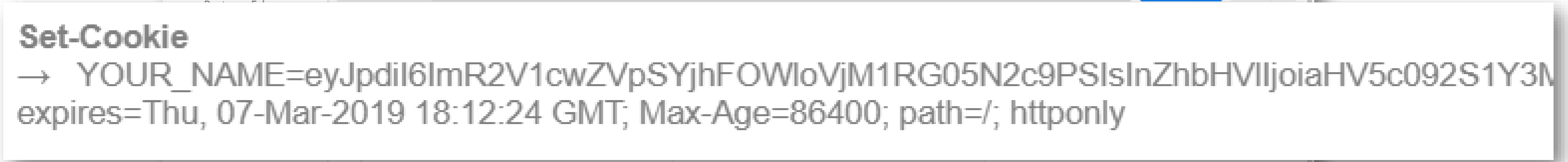
POST http://debian-wms.local/cookie

http://debian-wms.local/cookie-user-demo/register

POST http://debian-wms.local/cookie-user-demo/register

Status: 200 OK Time: 31 ms Size: 1.44 KB

```
Date → Wed, 06 Mar 2019 18:12:24 GMT

Server → Apache/2.4.25 (Debian)

Cache-Control → no-cache, private

Set-Cookie → XSRF-
TOKEN=eyJpdil6IktVRnUxZ0JqaTNHbTYxNUNvakpXS0E9PSIsInZhbHVlIjoiUzhsWEh1cXJuSzVtaVFENjNkeHFwNVF1MDBxbGZcL0hsQjNCcnVtTjF2b3ZMT1JJZFRtZzZ1elIXNE4yZ2w0MEliLCJtYWMiOiI1MWU1OT
expires=Wed, 06-Mar-2019 20:12:24 GMT; Max-Age=7200; path=/

Set-Cookie
→ laravel_session=eyJpdil6ImpHZDgxOUFZNCtOc3JlVmw5Q3ZzT0E9PSIsInZhbHVlIjoiRVdYcUtKNnJXUXlwUXJhQnlJT1I5T253VDF5UFBocXpJK1NyY1M2eG96RVRON3U1Q3VKQ3RITjI1MSs5OHpNWilsIm1hYy
expires=Wed, 06-Mar-2019 20:12:24 GMT; Max-Age=7200; path=/; httponly
```



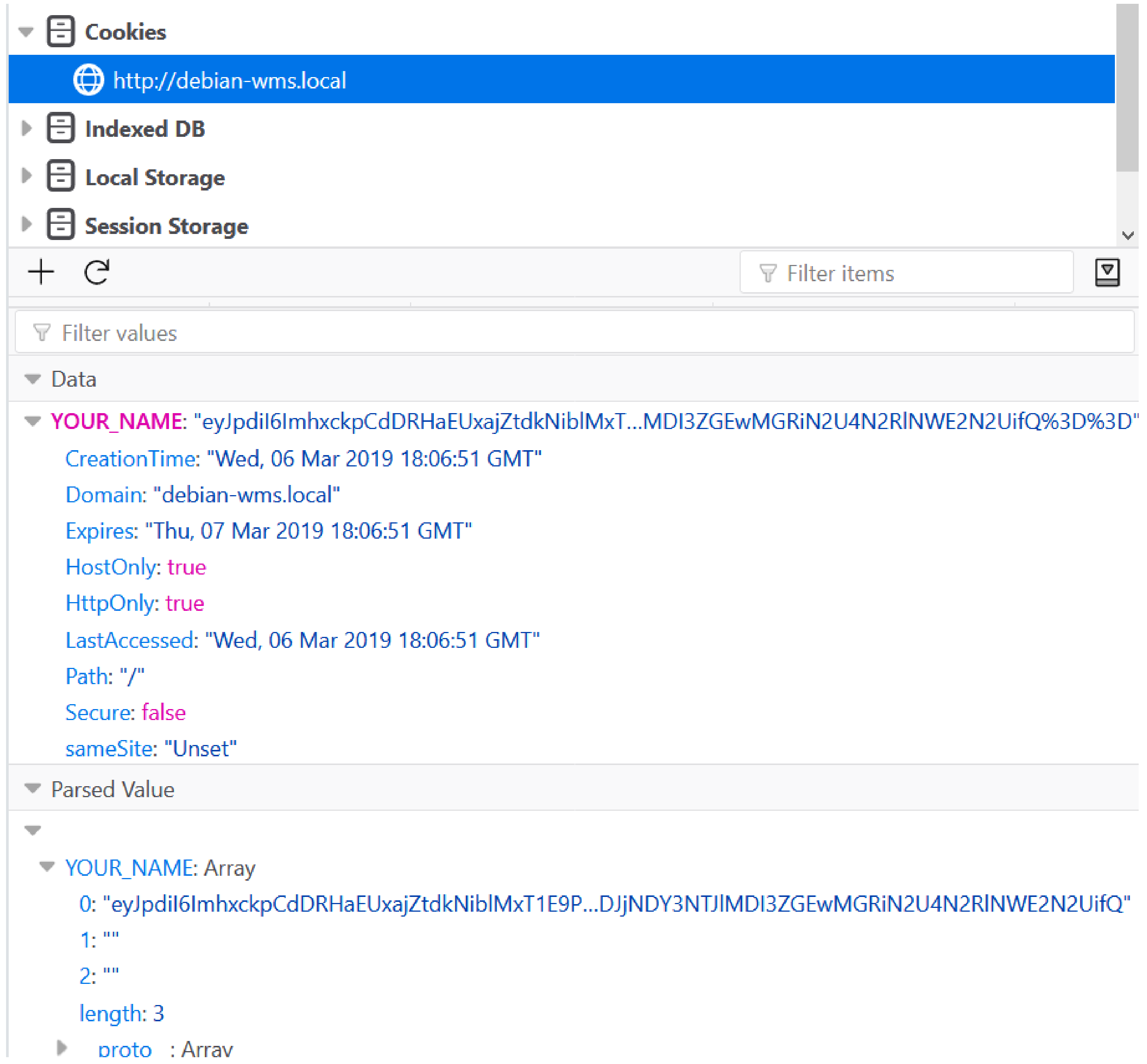
Set-Cookie
→ YOUR_NAME=eyJpdiI6ImR2V1cwZVpSYjhFOWloVjM1RG05N2c9PSIsInZhbHVlIjoiaHV5c092S1Y3N
expires=Thu, 07-Mar-2019 18:12:24 GMT; Max-Age=86400; path=/; httponly

howest hogeschool

Cookies: what about security?

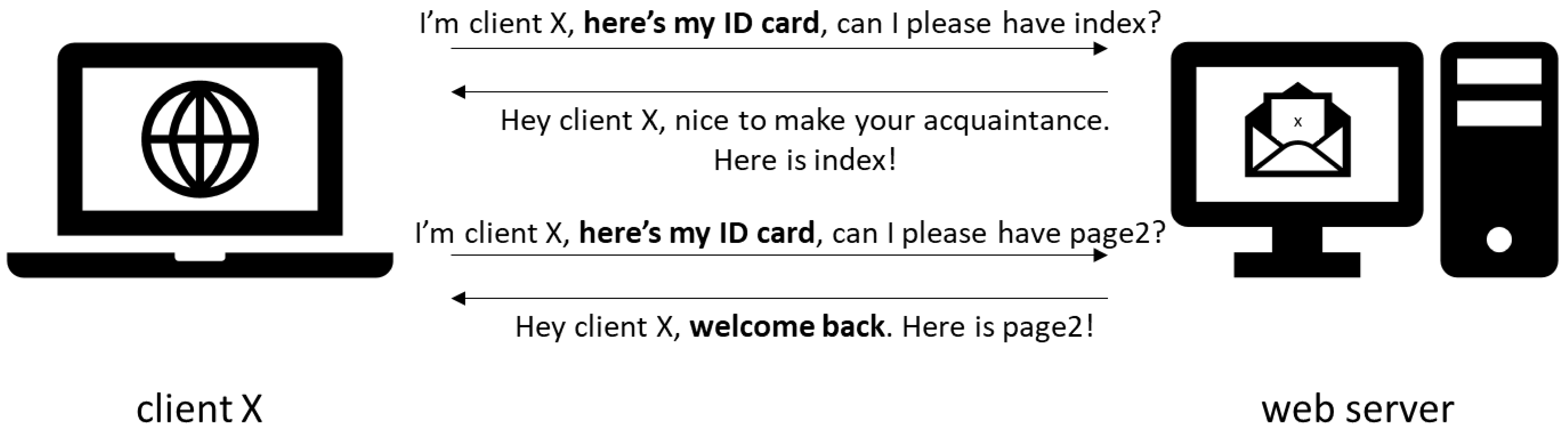
- Information is stored locally
- Transmitted with each request
- What about confidential data?
- Some solutions:
  - Encryption of cookie value (=default behavior in Laravel)
  - HTTPS

Visualisation in browser:



```
Data
YOUR_NAME: "eyJpdiI6ImhxckpCdDRHaEUxajZtdkNiblMxT...MDI3ZGEwMGRiN2U4N2RlNWE2N2UifQ%3D%3D"
  CreationTime: "Wed, 06 Mar 2019 18:06:51 GMT"
  Domain: "debian-wms.local"
  Expires: "Thu, 07 Mar 2019 18:06:51 GMT"
  HostOnly: true
  HttpOnly: true
  LastAccessed: "Wed, 06 Mar 2019 18:06:51 GMT"
  Path: "/"
  Secure: false
  sameSite: "Unset"
Parsed Value

  YOUR_NAME: Array
    0: "eyJpdiI6ImhxckpCdDRHaEUxajZtdkNiblMxT1E9P...DJjNDY3NTJlMDI3ZGEwMGRiN2U4N2RlNWE2N2UifQ"
    1: ""
    2: ""
    length: 3
    proto  : Array
```

Cookies: overcoming statelessness



I'm client X, **here's my ID card**, can I please have index?
Hey client X, nice to make your acquaintance. Here is index!
I'm client X, **here's my ID card**, can I please have page2?
Hey client X, **welcome back**. Here is page2!
x
client X
web server

Cookies versus local storage

Cookies:

- Key-value pairs (strings)
- Used to obtain state in stateless HTTP world
- Transmitted with each Request – Response

Local storage:

- Key-value pairs (strings)
- Used for local data only
- If you want data in local storage available on server, you need to send it explicitly (⇔ cookies)

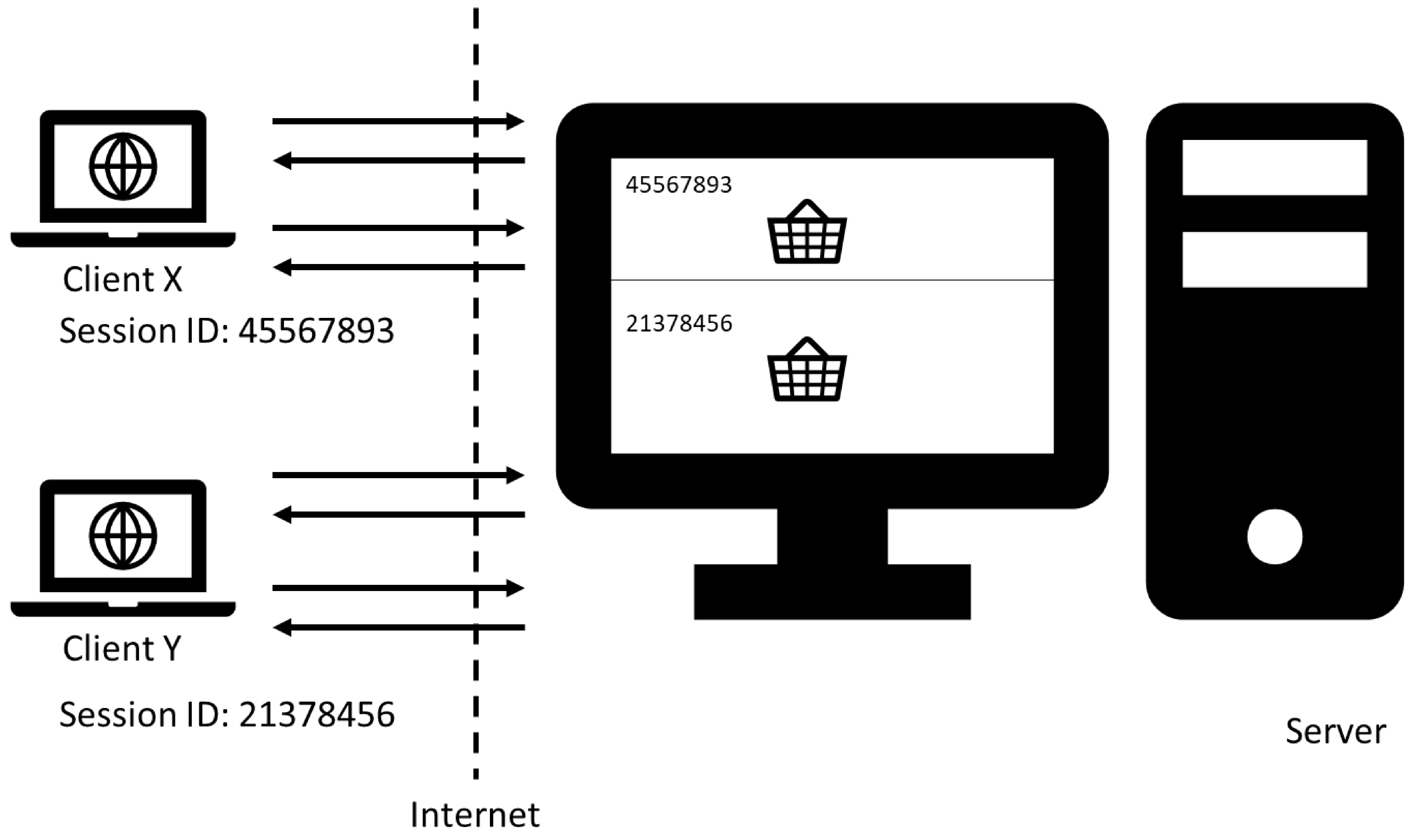
howest
hogeschool

Sessions

Overcoming HTTP statelessness

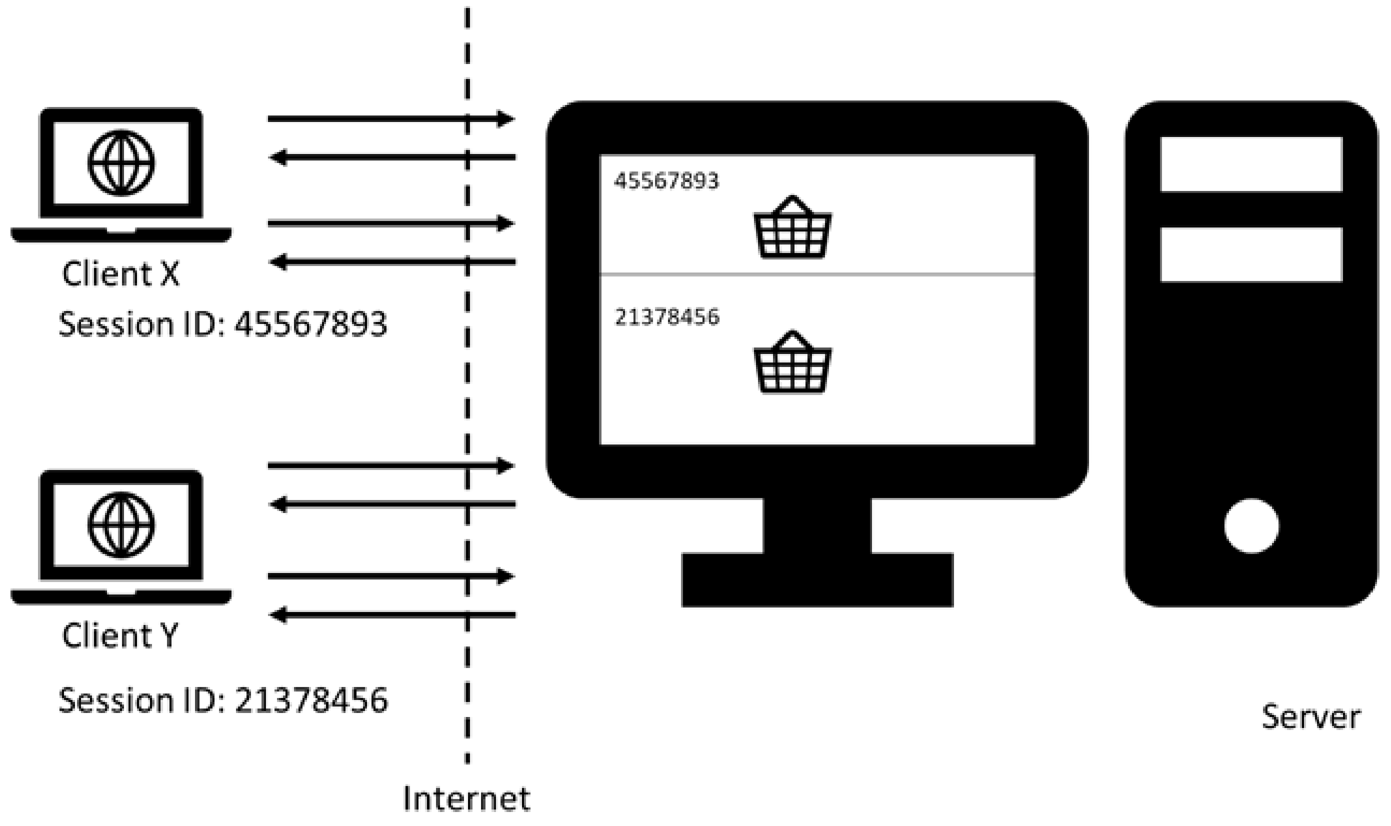
- We now know how to store information in HTTP cookies
- Information is stored on client
- Transmitted to server with each request
- **What if we want to store more data?**
- **Data not to be manipulated at the client side?**
- Examples:
  - Contents of shopping cart
  - Restaurant bookings
- Solution: sessions

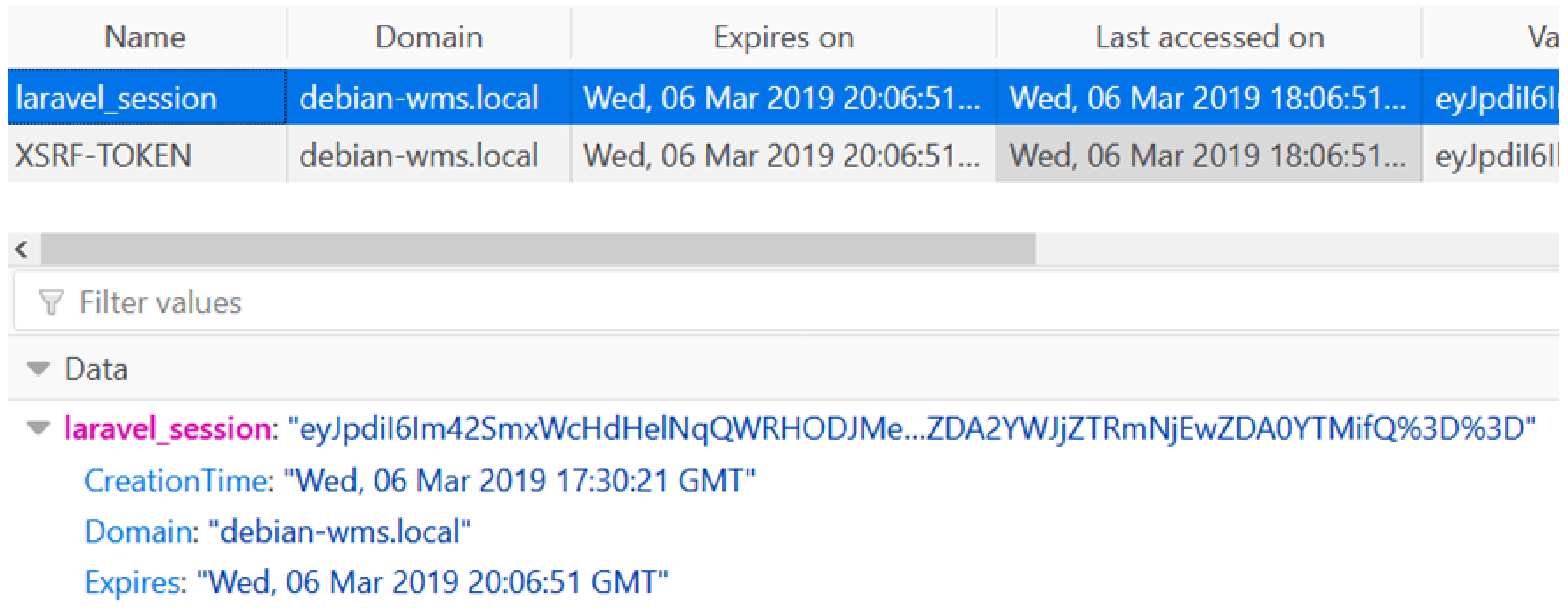
Sessions


Client X
Session ID: 45567893
Client Y
Session ID: 21378456
Internet
45567893
21378456
Server

Sessions

- Each session gets a unique (session) ID
- ID stored, usually in a cookie
- On server side, lots of info can be stored, associated with cookie (e.g. shopping cart)





Name	Domain	Expires on	Last accessed on	Va
laravel_session	debian-wms.local	Wed, 06 Mar 2019 20:06:51...	Wed, 06 Mar 2019 18:06:51...	eyJpdiI6I
XSRF-TOKEN	debian-wms.local	Wed, 06 Mar 2019 20:06:51...	Wed, 06 Mar 2019 18:06:51...	eyJpdiI6I

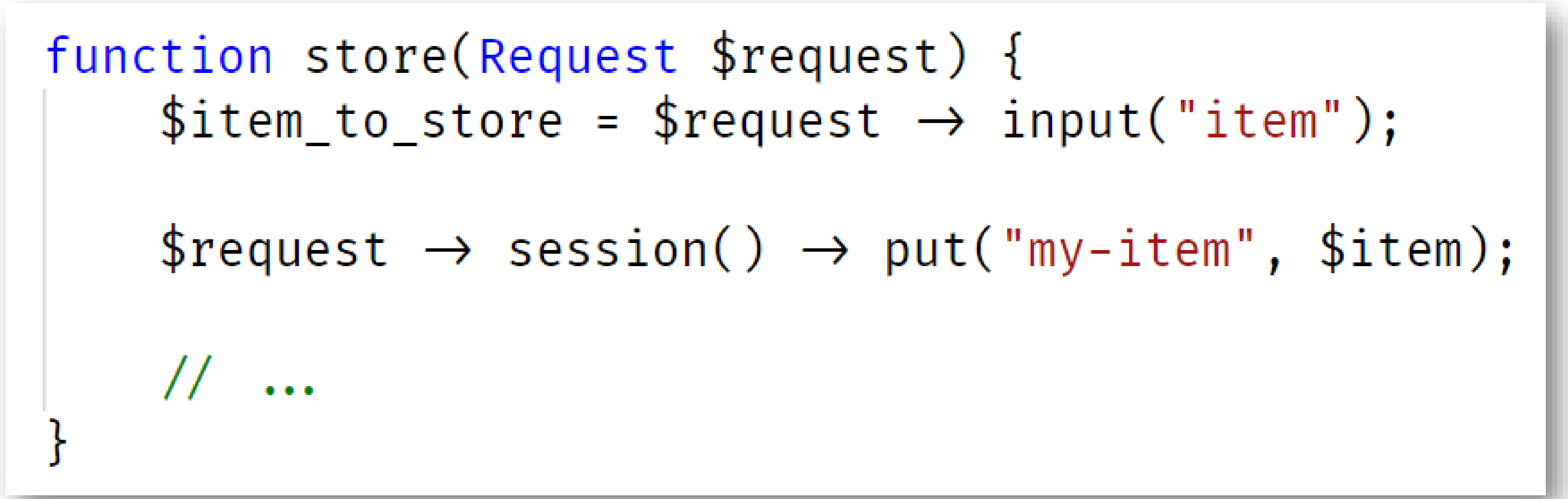
Filter values

Data

laravel_session: "eyJpdiI6Im42SmxWcHdHeINqQWRHODJMe...ZDA2YWJjZTRmNjEwZDA0YTMifQ%3D%3D"
CreationTime: "Wed, 06 Mar 2019 17:30:21 GMT"
Domain: "debian-wms.local"
Expires: "Wed, 06 Mar 2019 20:06:51 GMT"

Sessions in Laravel

- Store something in the session:

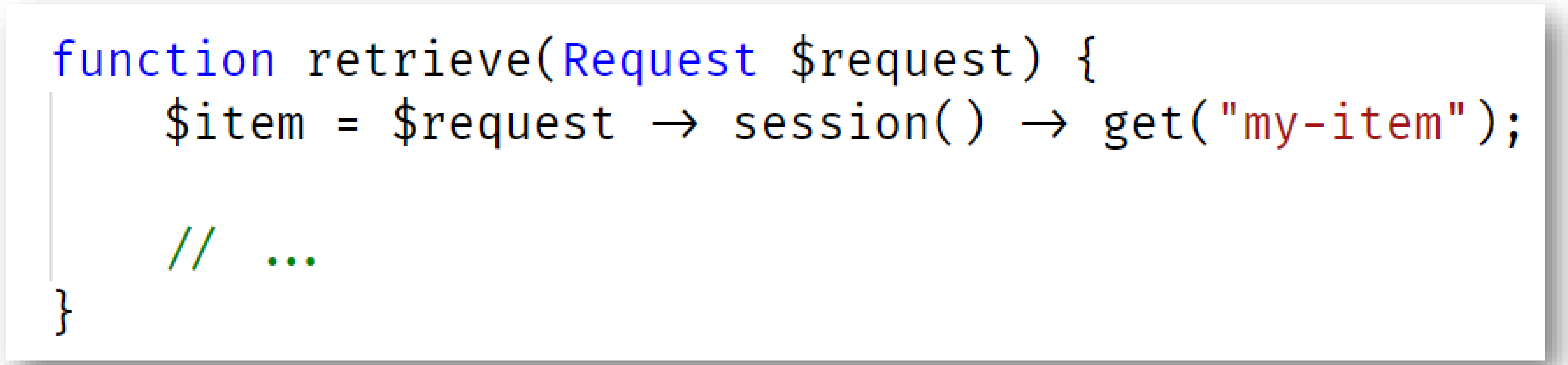


```
function store(Request $request) {
    $item_to_store = $request → input("item");

    $request → session() → put("my-item", $item);

    // ...
}
```

- You obtain a reference to the session via $request -> session() method
- Each item has a key (here "my-item") and value (here contents of $item)

Retrieving something from the session

- Retrieving something from the session

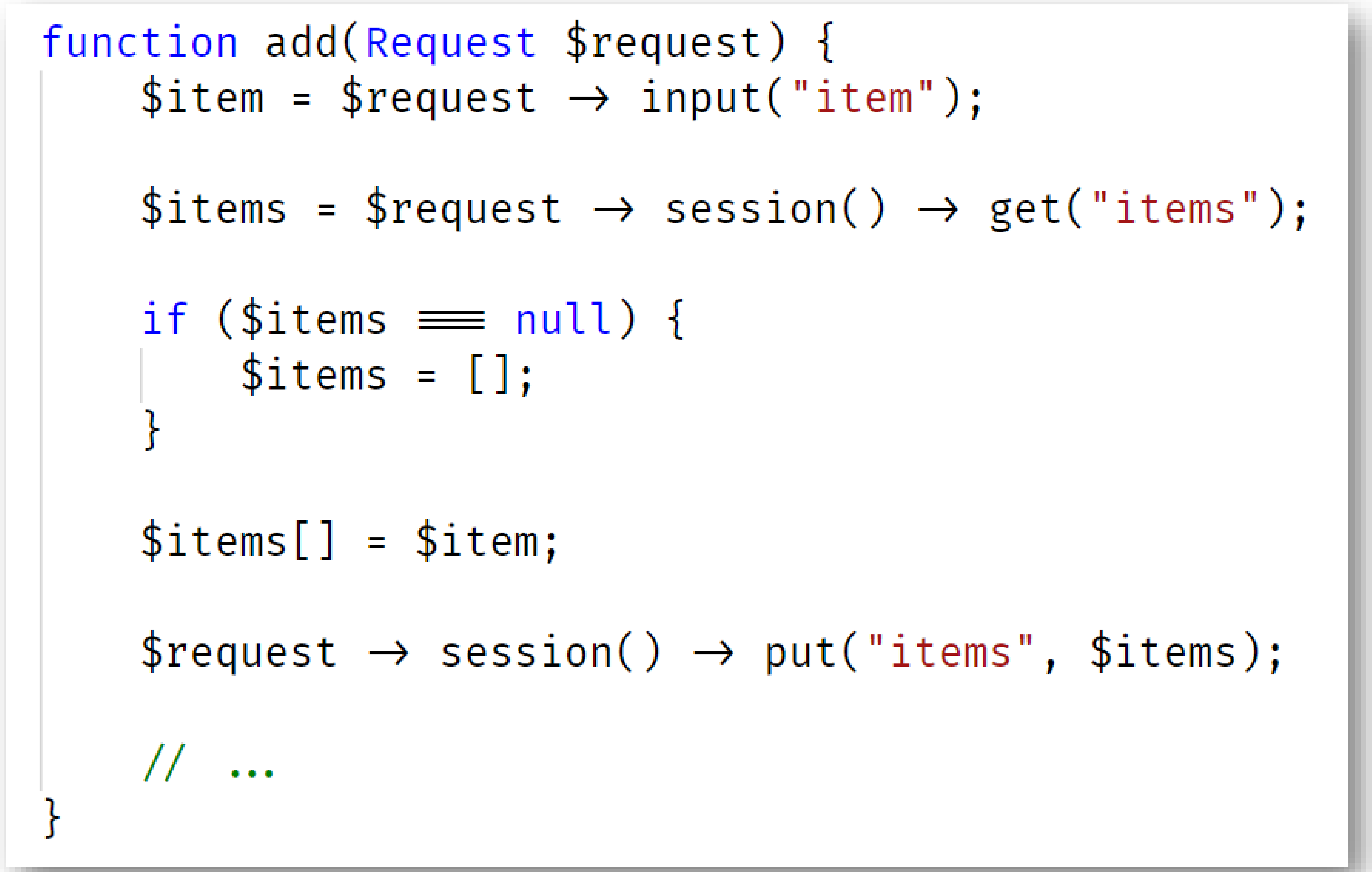


```
function retrieve(Request $request) {
    $item = $request → session() → get("my-item");

    // ...
}
```

- You obtain a reference to the session via $request -> session() method
- Retrieve item based on its key

Sessions in Laravel

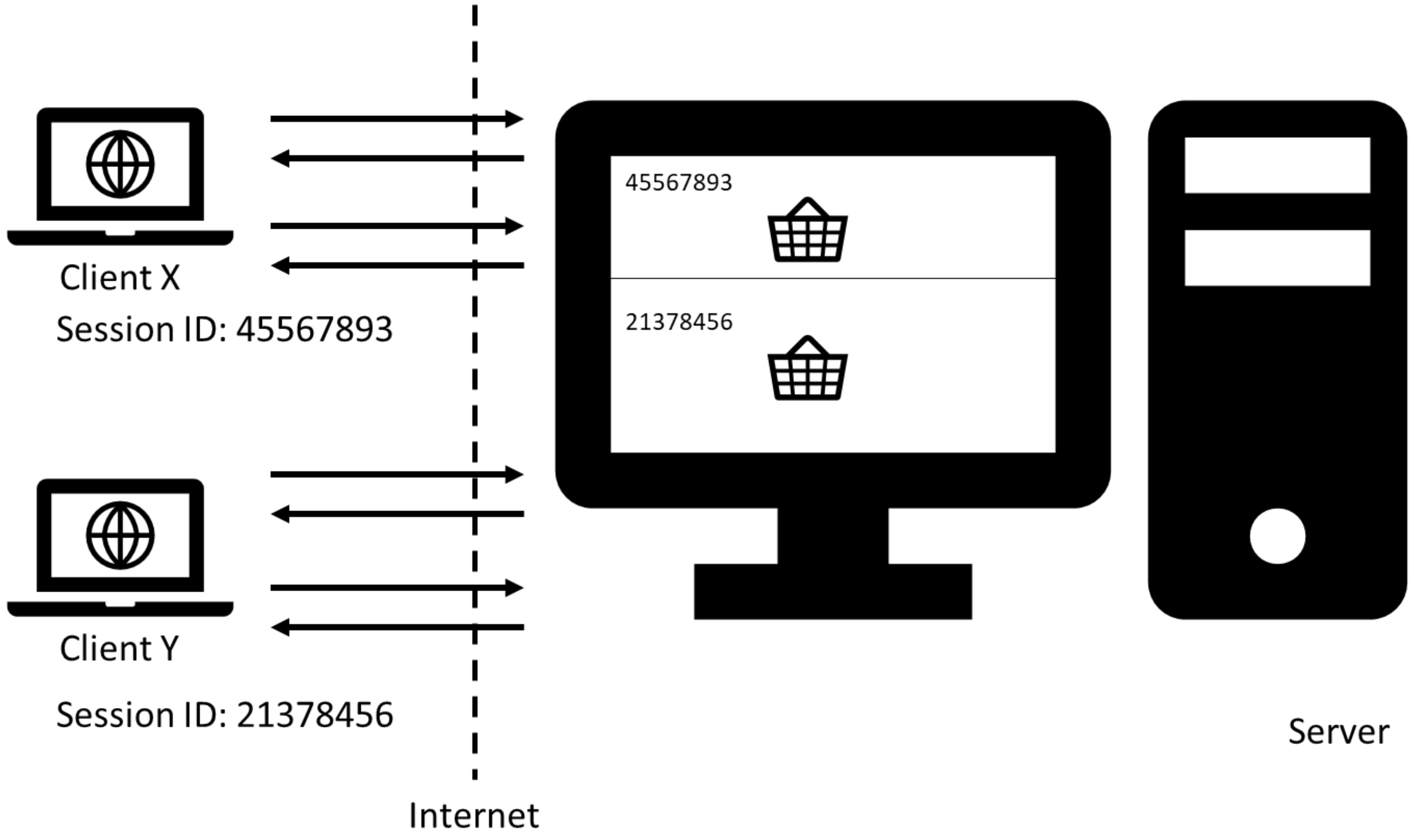
- Keeping list of items (array) in the session



```php
function add(Request $request) {
    $item = $request -> input("item");

    $items = $request -> session() -> get("items");

    if ($items === null) {
        $items = [];
    }

    $items[] = $item;

    $request -> session() -> put("items", $items);

    // ...
}
```

More information?

- https://laravel.com/docs/master/session



Questions?