

# Web, Mobile and Security

## Lab: Laravel - Intro

### 0. Preparing the work environment

- Launch your virtual machine, ensuring that all parameters are setup correctly (cfr. to the first lab).
- Try navigating to <http://debian-wms.local>, which should display the default start page.
- Try opening the share `\\debian-wms.local\code` from your Windows Explorer, which should give you access to the various web projects you already developed. P
- **Make sure to execute the procedure *Enabling URL rewriting* before you continue. Otherwise Laravel will not work!**

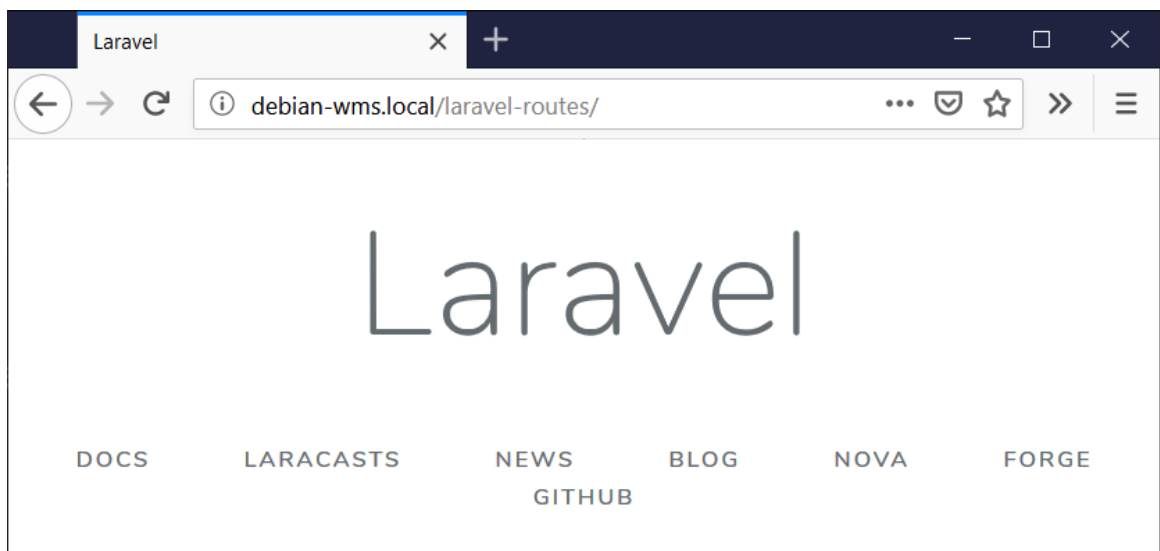
### 1. Exercise 1: Laravel Routing

- Create a new Laravel application called `laravel-routing` by executing the following command from within your PuTTY.

```
./build-laravel-project laravel-routing
```

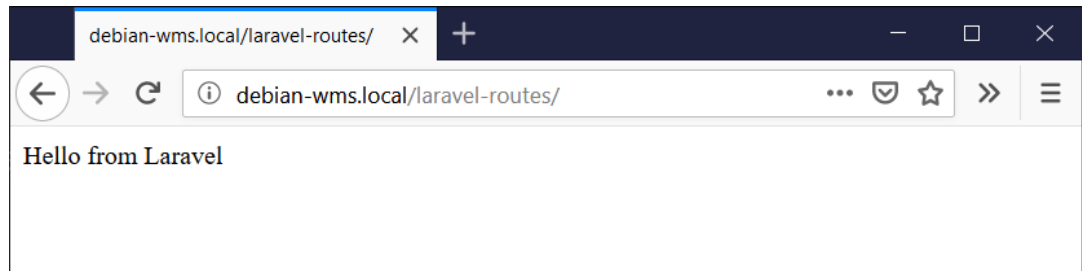
#### Some attention points:

- Make sure to connect as user `user` when executing the command above.
- Do not execute the command above more than once with the same parameter (`laravel-routing`), as this may give unwanted errors.
- Try surfing to <http://debian-wms.local/laravel-routing> and make sure you get the default Laravel start page:

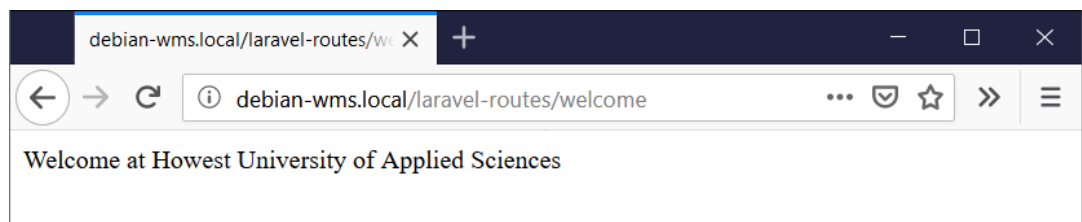


- Routing exercises:

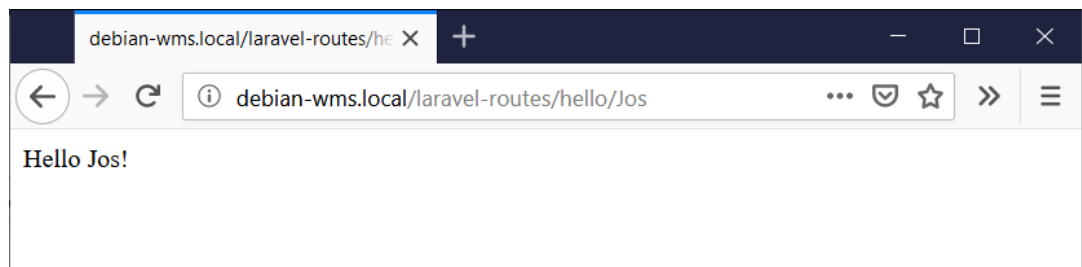
1. Define a **default** route so that navigating to <http://debian-wms.local/laravel-routing/> displays the message *Hello from Laravel* in the visitor's web browser **instead of** the default Laravel start page mentioned above.



2. Define a route so that navigating to <http://debian-wms.local/laravel-routing/welcome> displays a welcome message *Welcome at Howest University of Applied Sciences* in the visitor's web browser.



3. Define a route so that navigating to [http://debian-wms.local/laravel-routing/hello/\[name\]](http://debian-wms.local/laravel-routing/hello/[name]) displays a welcome message. For example, navigating to <http://debian-wms.local/laravel-routing/hello/Frederic> will display *Hello Frederic!* in the visitor's web browser.



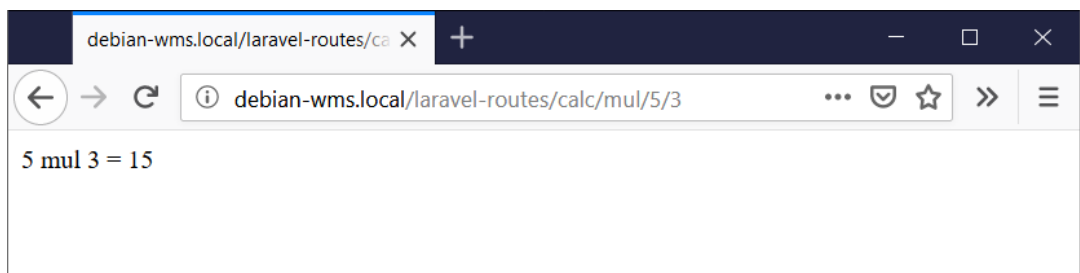
4. Define a route so that navigating to [http://debian-wms.local/laravel-routing/calc/\[operator\]/\[operand1\]/\[operand2\]/](http://debian-wms.local/laravel-routing/calc/[operator]/[operand1]/[operand2]/) will perform a calculation and displays the result in the visitor's web browser.

The following are valid values for **[operator]**:

- **add** → addition
- **mul** → multiplication
- **sub** → subtraction
- **div** → division

For example, to multiply 5 and 3, the user should be able to issue the following request:

<http://debian-wms.local/laravel-routing/calc/mul/5/3>



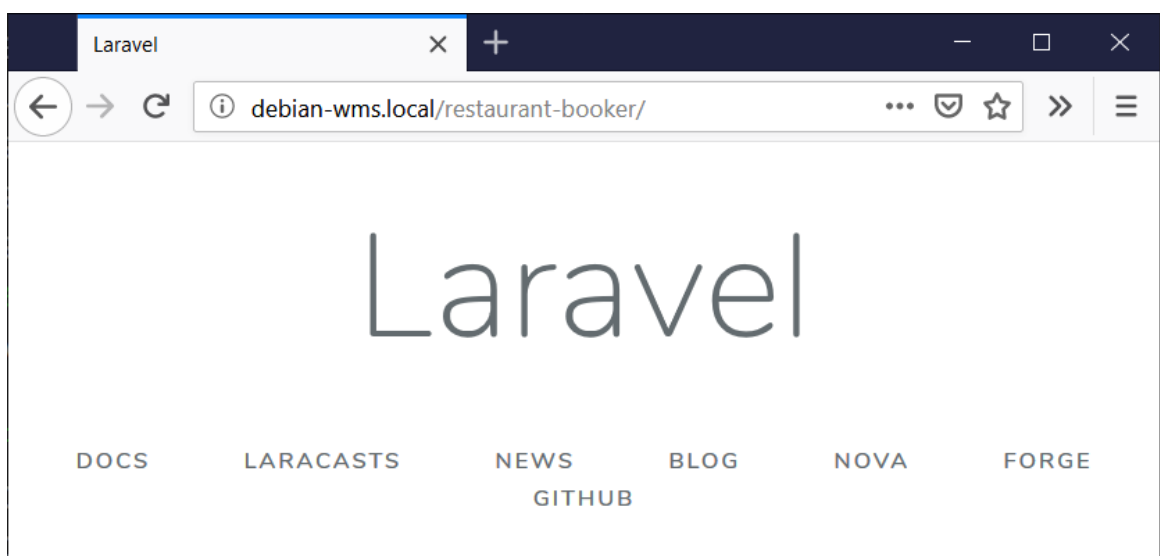
Hint: the **switch** statement may be of assistance to decide between the four operations. Take a look at <http://php.net/manual/en/control-structures.switch.php> for more information.

## 2. Exercise 2: Restaurant Booker

- Create a new Laravel application called **restaurant-booker** by executing the following command from within your PuTTY.

```
./build-laravel-project restaurant-booker
```

- Try surfing to <http://debian-wms.local/restaurant-booker/> and make sure you get the default Laravel start page:



- Create a view **booking.blade.php** to display a web form for a restaurant booking site. The form should at least contain the following input fields:
  - Lastname: textfield, required
  - Firstname: textfield, required
  - Number of persons: dropdownlist, containing the values 1 to 8, required
  - Date: datefield, required
  - Time: dropdownlist, required, containing the following possibilities:
    - 18:00
    - 18:30
    - 19:00
    - 19:30
  - Remarks: textarea, optional

- Hints:

- To easily generate the dropdownlist containing the numbers 1 → 8, you can use the Blade `@for` construct. Take a look at <https://laravel.com/docs/master/blade#loops> for more information.
- Do you get an error? Try to understand the error message displayed in the browser. Correct the error and try again.
- Your CSS files should be placed in the directory `/public/css` of the application.
- To reference the CSS files stored in that directory from your view(s), use the `asset` function as follows:

```
<title>Restaurant Booker</title>
<link rel="stylesheet" type="text/css" href="{{ asset('css/reset.css') }}" />
<link rel="stylesheet" type="text/css" href="{{ asset('css/screen.css') }}" />
```

- Create a controller called `BookingController` by issuing the following command from within the directory of your app in PuTTY:

```
php artisan make:controller BookingController
```

- This will generate a file called `BookingController.php` in the folder `/app/Http/Controllers`.
- Add a function to the controller to display the view you created a few steps earlier.
- In the correct file, add a route so that issuing a GET request to `http://debian-wms.local/restaurant-booker/` triggers the controller function that displays the view.
- Try out your code by navigating to `http://debian-wms.local/restaurant-booker/`. You should get something comparable to the screenshot below:

Restaurant Booker

Add your booking now

Lastname:

Firstname:

Number of persons:

Date:

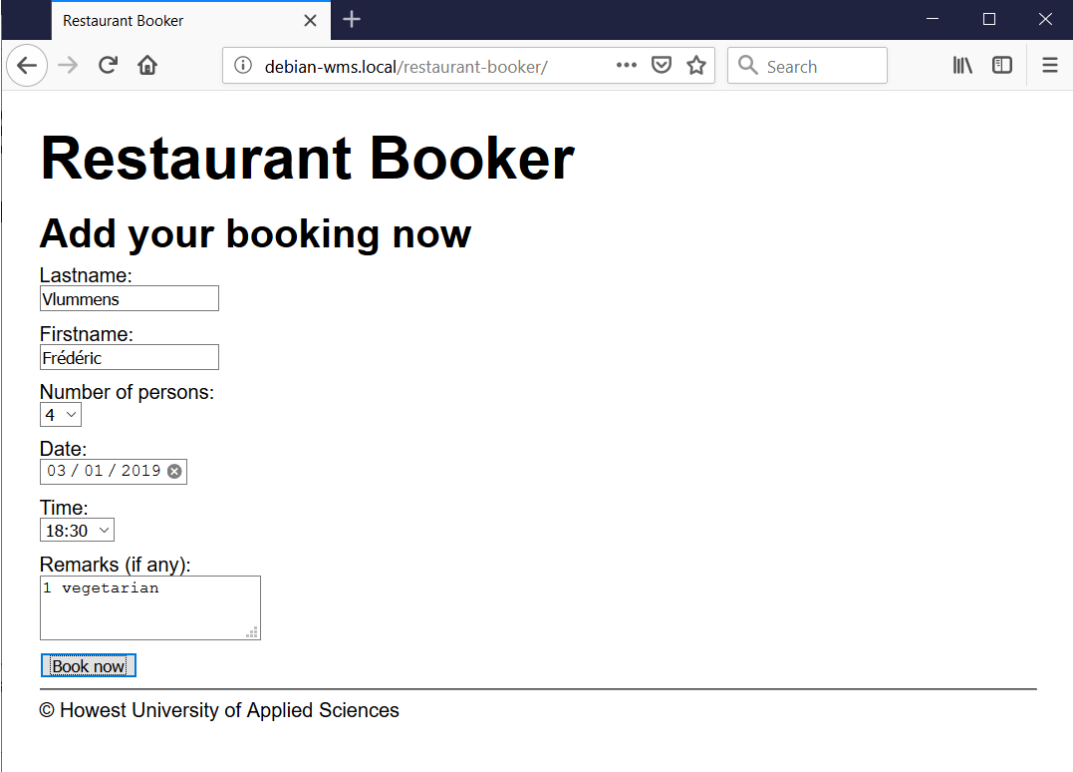
Time:

Remarks (if any):

© Howest University of Applied Sciences

- Next, we will write the code to process the booking. In our app, this will just mean displaying a summary of the booking at the top of the page.
- In your controller file, add a new function which retrieves the submitted information from the **Request** object and passes it to the correct view, which is the same one as before, i.e. **booking.blade.php**.
- Make sure a correct route is assigned (will you be using GET or POST?) to the controller function so that the form submission triggers the execution of the function.
- Upon submission of the form, the view should display a nice summary of the information received:

- Before submission:



A browser window titled "Restaurant Booker" shows a form for making a reservation. The URL in the address bar is "debian-wms.local/restaurant-booker/". The form includes fields for Lastname (Vlummens), Firstname (Frédéric), Number of persons (4), Date (03 / 01 / 2019), Time (18:30), and Remarks (1 vegetarian). A "Book now!" button is at the bottom of the form. Below the form, the copyright notice "© Howest University of Applied Sciences" is displayed.

**Restaurant Booker**

**Add your booking now**

Lastname: Vlummens

Firstname: Frédéric

Number of persons: 4

Date: 03 / 01 / 2019

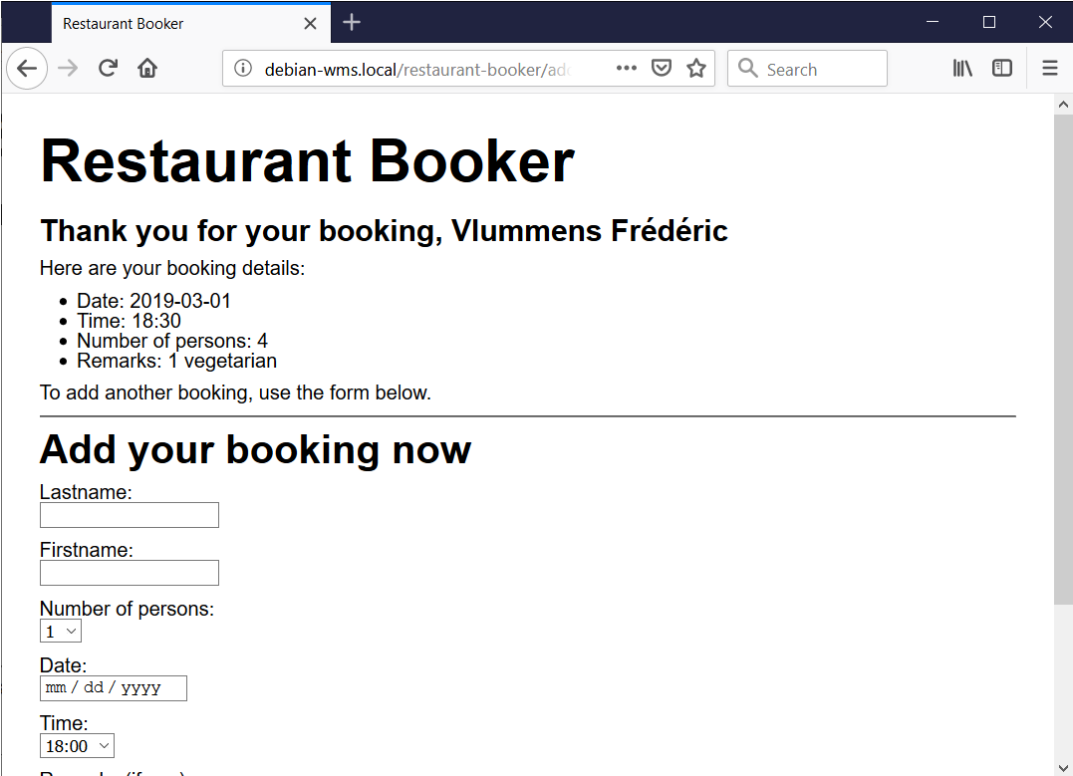
Time: 18:30

Remarks (if any): 1 vegetarian

[Book now!](#)

© Howest University of Applied Sciences

- After submission:



The browser window now shows a confirmation message: "Thank you for your booking, Vlummens Frédéric". Below this, the booking details are listed: Date: 2019-03-01, Time: 18:30, Number of persons: 4, and Remarks: 1 vegetarian. A message encourages the user to use the form below to add another booking. The form fields are reset, with the Number of persons set to 1 and the Time set to 18:00. The Date field is empty and shows a placeholder "mm / dd / yyyy".

**Restaurant Booker**

**Thank you for your booking, Vlummens Frédéric**

Here are your booking details:

- Date: 2019-03-01
- Time: 18:30
- Number of persons: 4
- Remarks: 1 vegetarian

To add another booking, use the form below.

**Add your booking now**

Lastname:

Firstname:

Number of persons: 1

Date: mm / dd / yyyy

Time: 18:00

Remarks (if any):