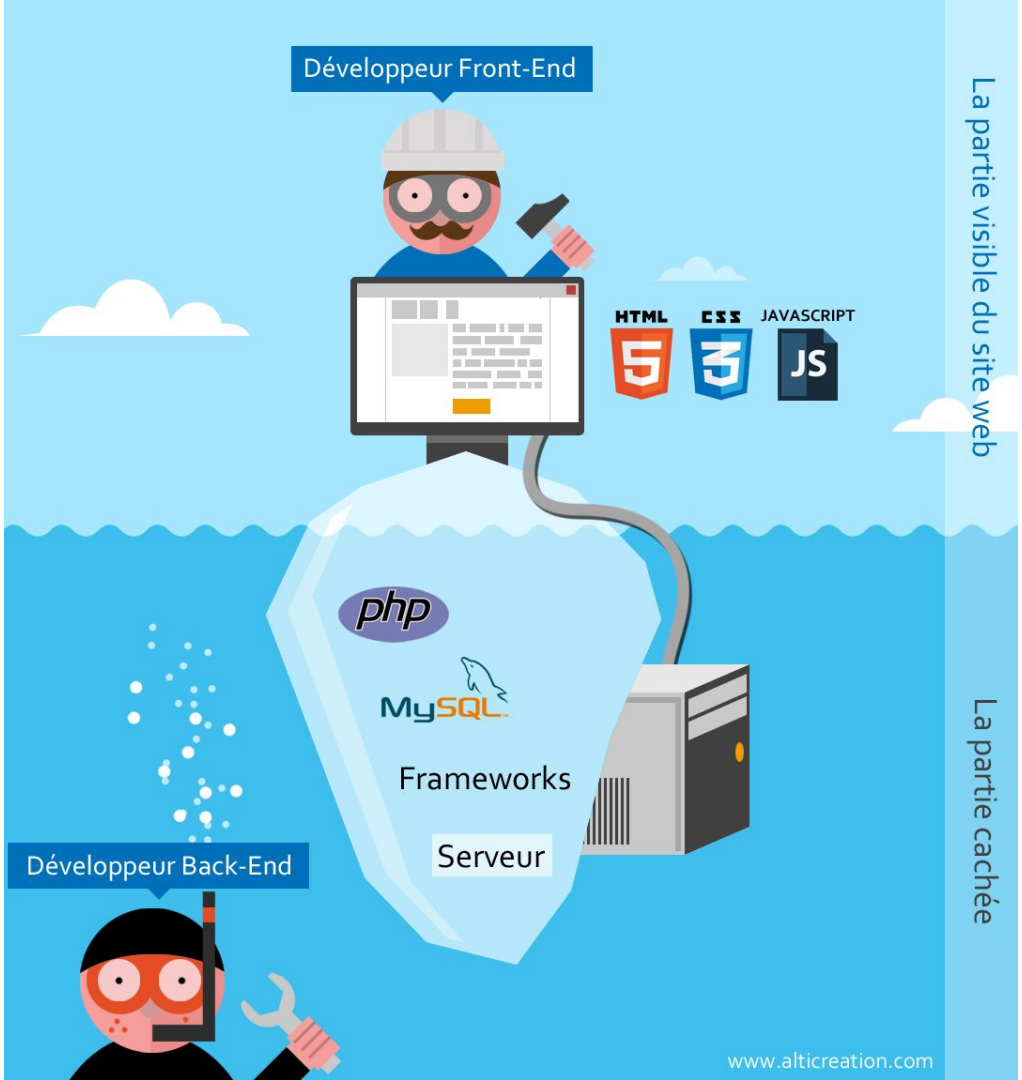
Advanced Web Development

Course overview

1) A bit of theory
2) The Project
3) Setup all the pieces
4) Let’s code!

Intro

- What do you expect from this class?
- What is Web Development?
- Why do you need to learn Web Dev?
- What are the jobs related to Web Dev?


Développeur Front-End
HTML
CSS
JAVASCRIPT
La partie visible du site web
php
MySQL
Frameworks
Serveur
La partie cachée
Développeur Back-End
www.alticreation.com

Some stacks


node
METEOR
ANGULARJS
by Google
Java
Microsoft
.NET
HTML
JS
CSS
RAILS
sf
hadoop
python
php
jQuery
write less, do more.

The Project

Create an item management website (e.g.: blog, online library… but not ticket management!)

You can choose the language / MVC framework that you want. The course is based on Slim Framework with PHP but feel free to use something you want to learn or master.

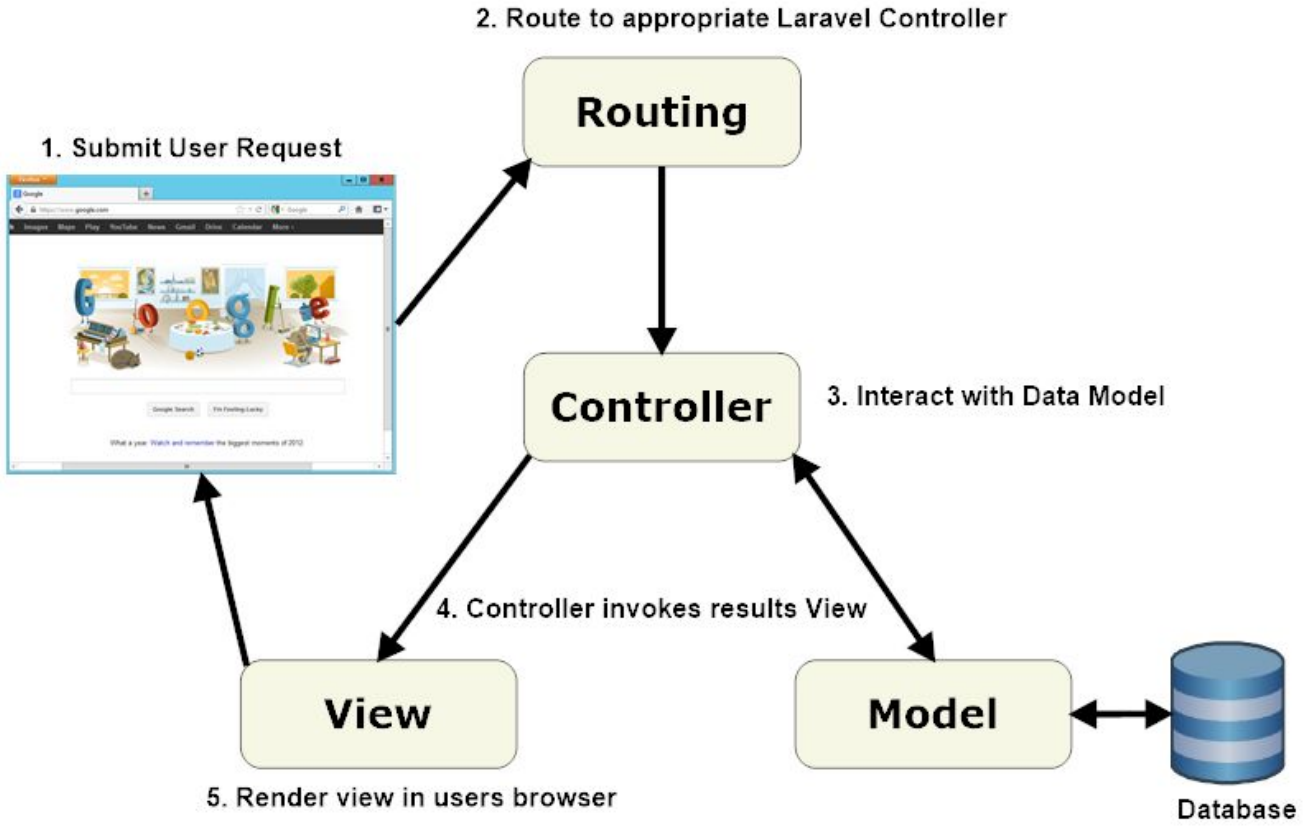
Goals:

- Item management: Show all, Show one, Add, Update, Delete
- Use of an ORM (Eloquent, Doctrine…)
- Create UI/UX with CSS (Twitter bootstrap preferably)
- Deployment on any cloud / dedicated server that you want
- Build a real website that works without bugs

What I expect:

- Teams of 2 maximum
- Due date: 11:59pm Dec 8th (+1 pt bonus) / Dec 22nd
- Github repo with all your source files (without logs / vendor etc. Only your source code)
- URL (heroku, dedicated or anything else)
- An email - please proceed as follow:
    - Title : AWD esiea 2017
    - Content : github repo link / URL / name 1 / name 2
    - **Nothing else. If any question, please send me a separate email**
    - Email: ancast+esiea17@gmail.com

MVC design pattern



TD 1 - How to install?

1) Install MySQL -> http://dev.mysql.com/downloads/mysql/
2) Install composer in order to install Slim Framework -> https://getcomposer.org/download/
3) Install Slim Framework -> https://www.slimframework.com/
4) Create & set up Github account
5) Make everything work together

TD 1 - Let’s play with the framework

1) Use routes.php to add another route for the same view
2) Create another view
3) Play with arguments

TD 2 - Entities and ORM

1) Install and configure Eloquent (an ORM) to your project -> http://www.slimframework.com/docs/cookbook/database-eloquent.html
   To make it work properly, you'll need to install Doctrine: "php composer.phar require doctrine/dbal". Create a database using command line or GUI tools (e.g.: Sequel for Mac…)
2) Create a folder "Models" in src and create an Entity (a PHP class basically) of your choice in this folder - no need for getters / setters. You should use the 3rd link in "Useful links". Then add the code of the next slide in public/index.php
3) Create a route "/install" with a controller that builds a schema using the next slide. It should also populate your DB with 2 or 3 items.
4) Create route / controller / view to show all the items
5) Create route / controller / view to show one item

TD 2 - Some code to add

To be added in public/index.php line 13:

```
spl_autoload_register(function ($classname) {
    require (__DIR__ . '/../src/Models/' . $classname . ".php");
});
```

Establish DB connection:

```
$this->db;
```

Building schema in the /install controller:

```
$this->db;
    $capsule = new \Illuminate\Database\Capsule\Manager;
    $capsule::schema()->create('articles', function
(\Illuminate\Database\Schema\Blueprint $table) {
        $table->increments('id');
        $table->string('name');
        // Include created_at and updated_at
        $table->timestamps();
    });
// Delete table if needed
// $capsule::schema()->dropIfExists('articles');
```

TD 2 - Useful links

Add and configure Eloquent with Slim Framework:
http://www.slimframework.com/docs/cookbook/database-eloquent.html

To build a schema and create a model:
https://vkbansal.me/blog/using-eloquent-outside-laravel/

Insert / Update / Retrieve / Delete a model:
https://laravel.com/docs/5.3/eloquent

TD 3 - Framework and HTML

1) Create a view with a HTML form which submit to a POST route
2) Create route / controller / view to add an item
3) Create route / controller / view to delete one item
4) Create route / controller / view to update one item
5) Install and include bootstrap to your views: http://getbootstrap.com/
6) Add some design to the “show all” view with buttons linking to Add / Update / Delete
7) Proceed with other functions

TD 4 - HTML and Deployment

1) Create your own CSS sheet and link it to your templates using the same way that Twitter Bootstrap.
2) Write and use your own styles to make your item management platform unique!
3) Create an account on Hostinger / Google cloud / Amazon Web Services (or any) and set it up to deploy your application… or use a dedicated server!

Grading schema

Main

Not deployed -> -5

No CSS -> -5

SQL in your code -> -3

Non working add -> -2

Non working edit -> -2

Non working delete -> -2

Non working showOne -> -2

Non working showAll -> -2

Non working install -> -2

Misc

Bad variable name -> -1 to -3

Bad class name -> -1

Bad separation between MVC -> -2

Code in wrong files -> -2

Unverified form variables -> -2

Unverified model returns -> -2

Useless files -> -1

Appendix

What is an object?

- A bunch of arguments and methods
- Can extend another object
- Can implement an interface
- Difference between private / public method
- Getter & setter

PHP memo

$a = 1 : initialise a to 1

$o = new Object() : initialise an object Object and link it to the var $o

is_null() : return true if null

empty() : return true if null or empty

$_POST : array that contains all POST vars

Var_dump: show type & content of a var

exit: stop script execution

PHP Getter / Setter Generator:

http://mikeangstadt.name/projects/getter-setter-gen/

Additional Resources

GitHub cheat sheet:
https://services.github.com/on-demand/downloads/github-git-cheat-sheet.pdf

Why use a framework :
http://arunviswanathan7.blogspot.fr/2014/07/top-10-reasons-why-you-should-use-php_10.html

What is MVC?
https://www.youtube.com/watch?v=qXRcVhWxuaU