The MVC Structure

MVC is a structure for building applications, software, and websites.
The acronym stands for:

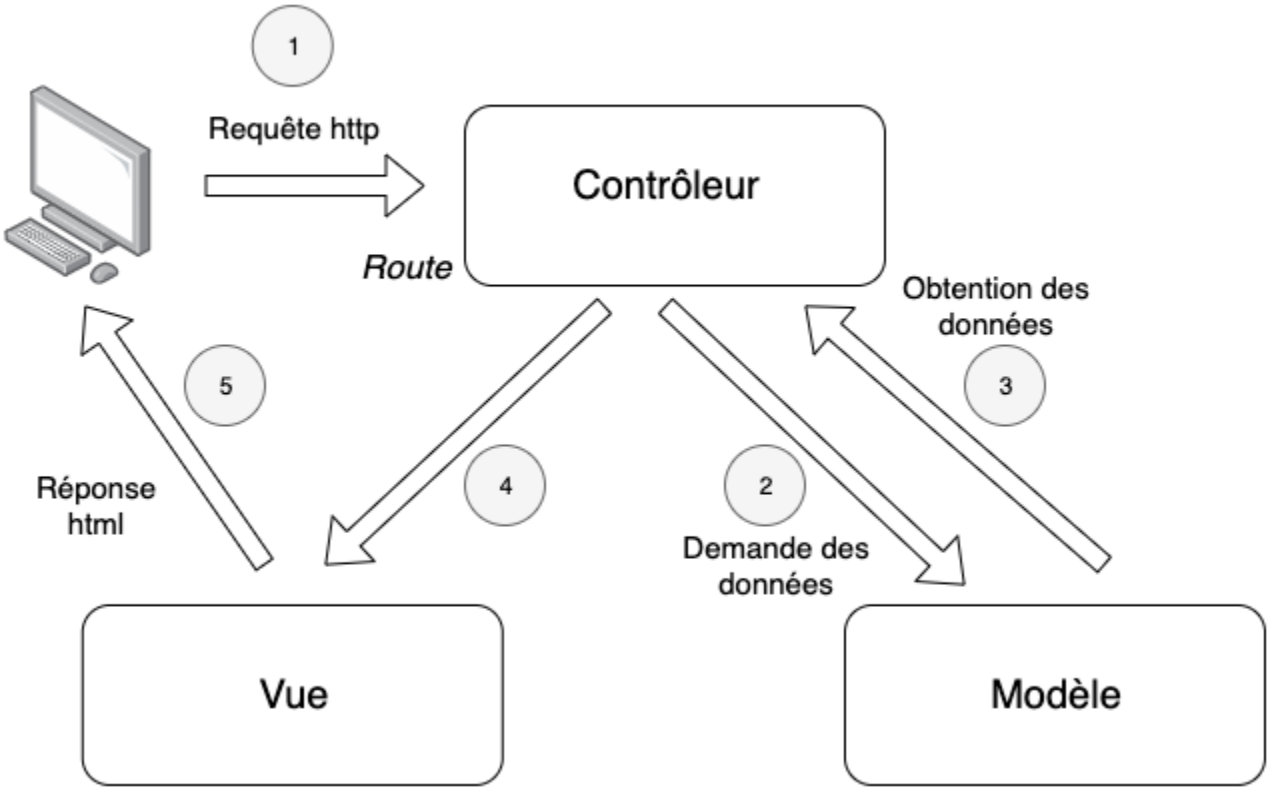
- Model
- View
- Controller

It helps organize your code by dividing it into three distinct parts.
Until now, we had our database queries, algorithms, and HTML all in the same file. When we wanted to change something, we had to dig through dozens of lines to find what we were looking for.

The principle of MVC is to have:

- **Controllers**:
  Files that group all the logic/algorithms of our pages, organized by topic.
- **Models**:
  Files that contain all the database queries, also organized by topic.
- **Views**:
  Files that contain only the display (HTML) of the page.

In MVC, all user requests are sent to the controllers. The controller may exchange data with the model, and finally, the view is returned to the user.



Blog in MVC Version

Here we will reproduce the blog website created during the CRUD course, but in MVC version.

1. Router

It's possible to use an MVC structure without a router.
But that would force us to have as many controllers as we have pages (or to pass a lot of information through GET parameters).
Our goal here is to group pages by topic, so we will improve our router to handle MVC.

Steps:

1. `.htaccess` – same as in the router lesson.
2. `routes.php` – add controllers and functions.
3. `index.php` – call the appropriate functions.

2. Our First Model

Let's create a `model` folder with a file named `userModel.php`. Inside, we'll include our database connection file using PDO.

As mentioned earlier, models are used to group all the database queries.
As the file name suggests (`userModel.php`), this model will contain queries related to users.

3. Our First Views

We'll now create a `view` folder containing a `user` subfolder. Inside it, we'll create four files:

- `list.php`
- `inscription.php`
- `delete.php`
- `update.php`

These files will only contain the display part of the page: HTML and PHP `echo`.

4. Our First Controller

Let's create a `controller` folder containing a file named `userController.php`.

Inside this file, we will create 4 functions representing the 4 pages related to users.
These functions will contain all interactions with the model, algorithmic logic, and the inclusion of the appropriate view.

Model with MongoDB Version

We have already built a full CRUD in MVC using MySQL, but now we will use MongoDB instead.