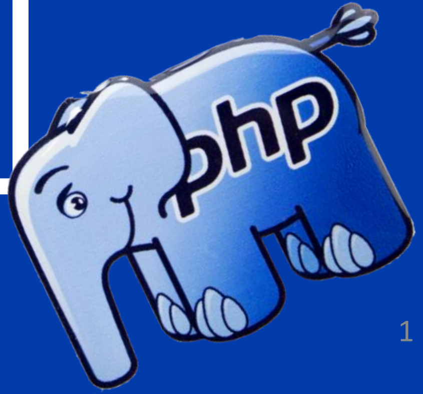
PHP

CHAPTER 5 – MVC & PROJECT STRUCTURE



🥇 OBJECTIVES FOR TODAY 🥇

✓ **Why** do we need **responsibilities** in project ?

✓ Be able to use the **MVC pattern** :
- The model
- The controller
- The view

✓ **Why** do we need a clean PHP project structure

✓ Be able to follow a **clean structure**

DISCUSS

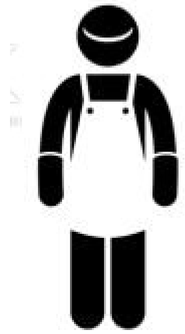
10 MIN

3 **responsibilities** in a restaurant

- ✓ Define their role ? **What do they need to do** ?
- ✓ **Why** do we need to separate those responsibilities ?

CUSTOMER

WAITER



COOK

DISCUSS

10 MIN

What are the responsibilities of this code ?

(many answers possible)

A - **Get data** from database
B - **Decide** what the view **should display** *(depending on the data)*
C - **Display** the view

```
<?php
           $statement = $connection->prepare("select * from posts");
           $statement->execute();
           $posts = $statement->fetchAll();
?>

<?php foreach($posts as $post): ?>
  <li>
           <?= $post['title'] ?> |
           <span><?= $post['description'] ?></span> |
           <a href="controllers/post/post.delete.controller.php?id=<?= $post['id'] ?>" >Delete</a> |
           <a href="views/post/form.edit.view.php?id=<?= $post['id'] ?>">Edit</a>
  </li>

<?php endforeach; ?>
```

EXPLAIN

10 MIN

Why do we need **responsibilities ?**

REAL LIFE PROJECT

- ✓ Participants know **what to do**
- ✓ **Eliminate duplication** of work
- ✓ Clarify a **work flow**
- ✓ **Help to check** if things are done

IT PROJECTS

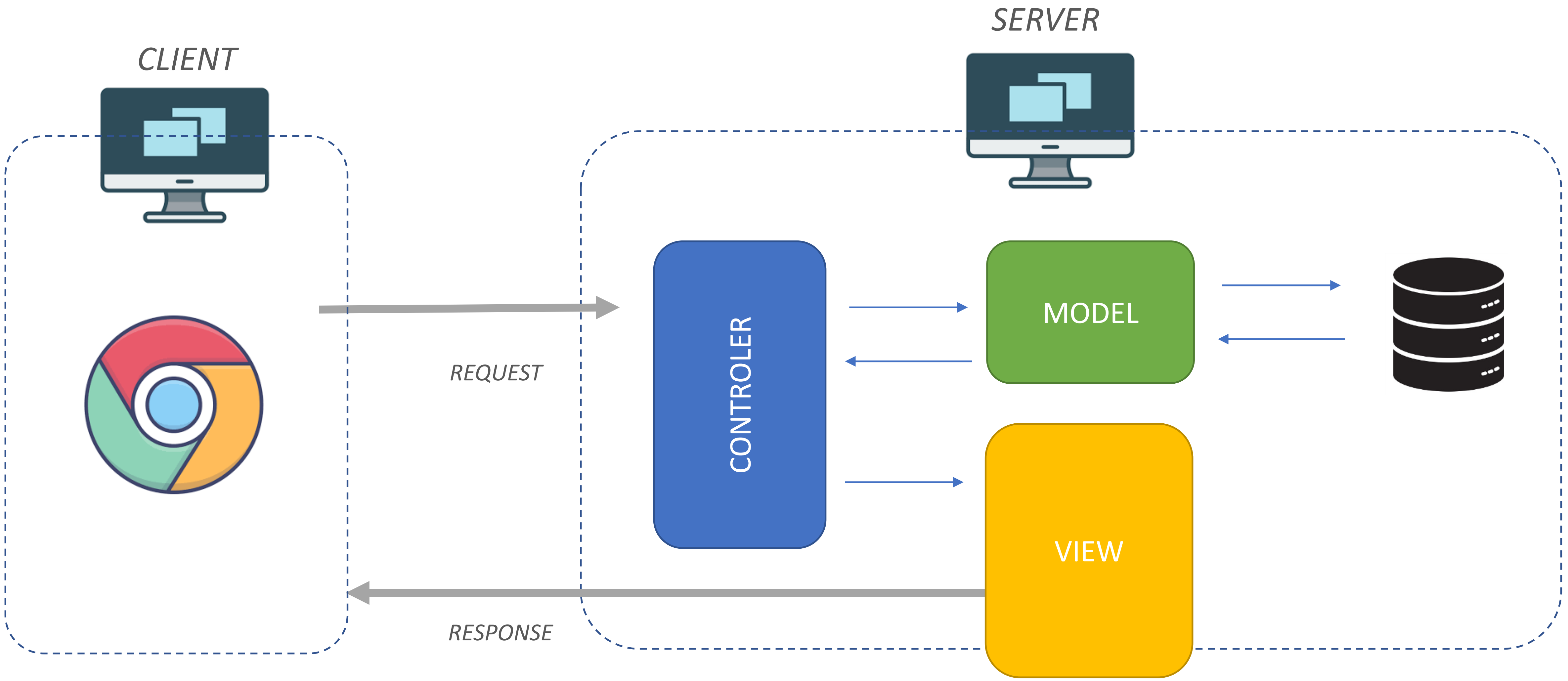
- ✓ Developer know **where to code**
- ✓ **Eliminate duplication** of work
- ✓ Clarify a **work flow**
- ✓ **Help to test** the code

EXPLAIN

10 MIN

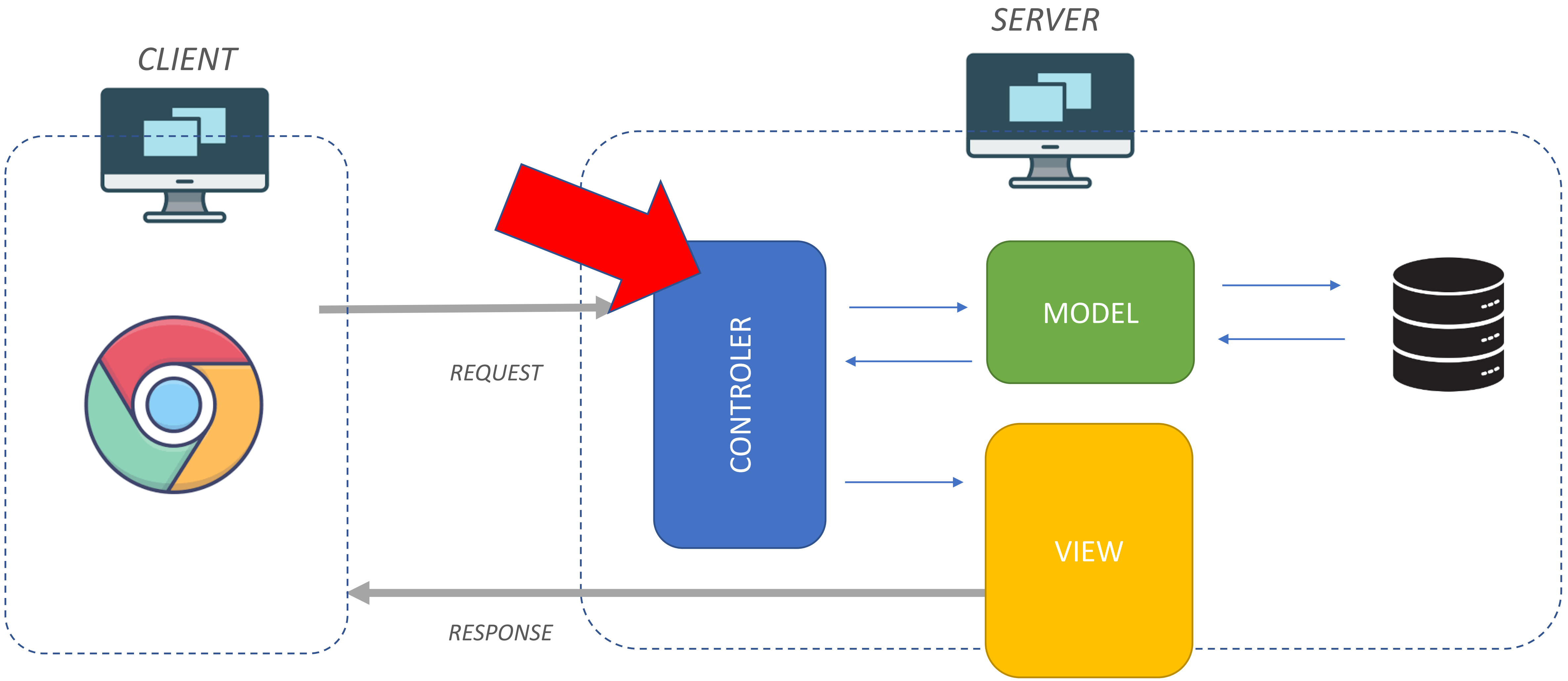
The MVC architecture

- ✓ The CONTROLER Is in charge to **provide the right data** to the view
- ✓ The CONTROLER can also validate user inputs, hide field, display warnings



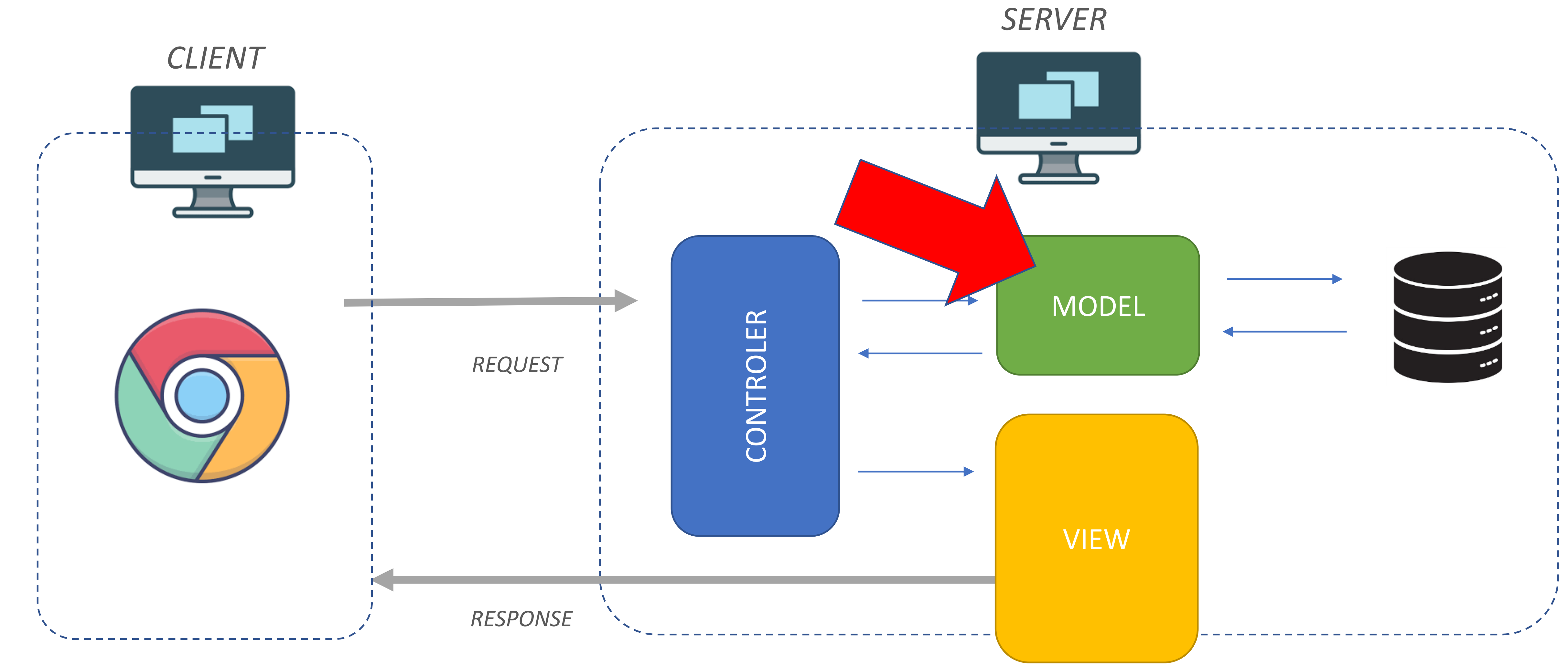
1 – The controller

- ✓ **Provide the right data** to the view
- ✓ Validate user inputs, hide field, display warnings



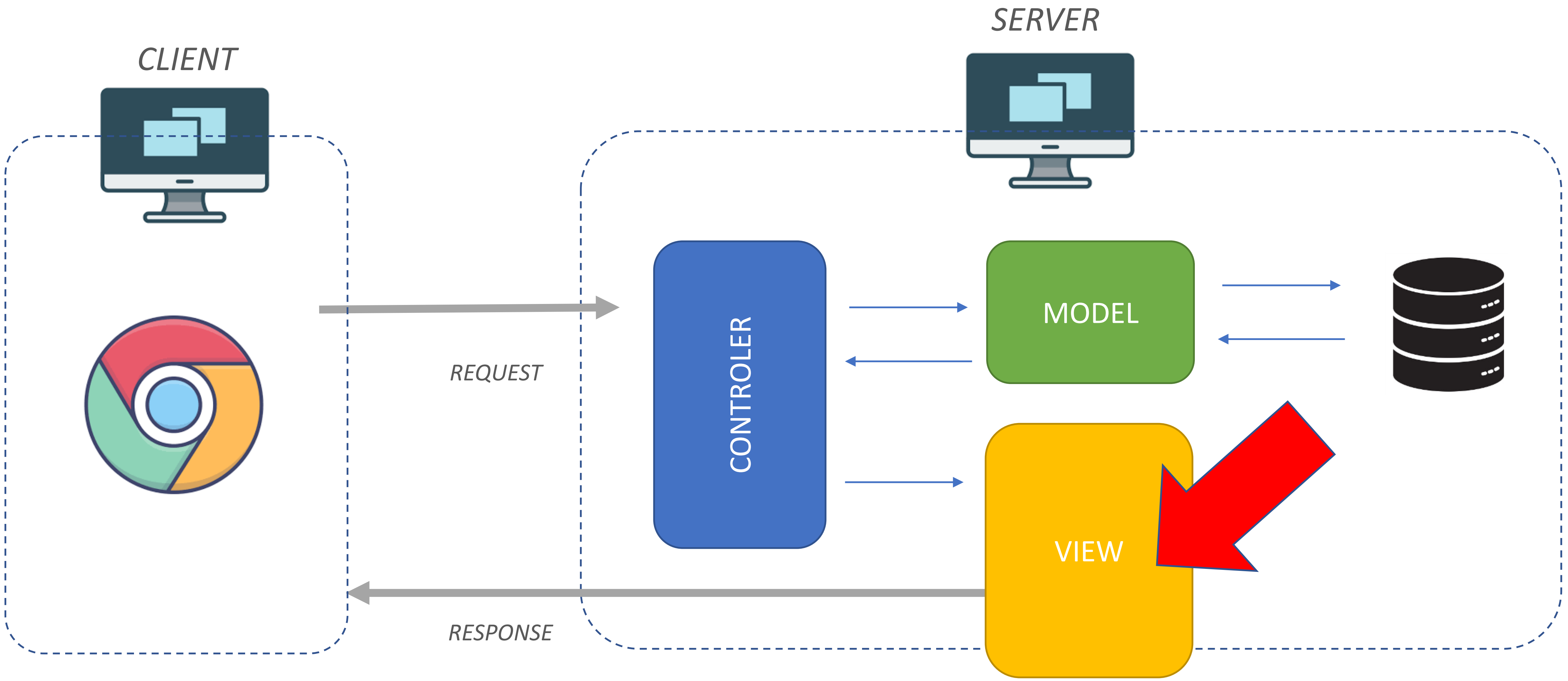
2 – The model

- ✓ **Provide the data needed by the application**
- ✓ **Changing the data** : remove, insert, add..



3 – The view

✓ **Build the user interface** , given the information provided by the controller



DEMO

10 MIN

The MVC architecture

post.**controler**.php

```
$heading = "Post Page";
require 'models/post.model.php';

$posts = getPosts();

require "views/post/post.view.php";
```

1 – Get data

post.**model**.php

```
function getPost(int $id) : array
{
    global $connection;
    $statement = $connection->prepare("select * from posts where id = :id");
    $statement->execute([':id' => $id]);
    return $statement->fetch();
}
```

2 – Call the view

post.**view**.php

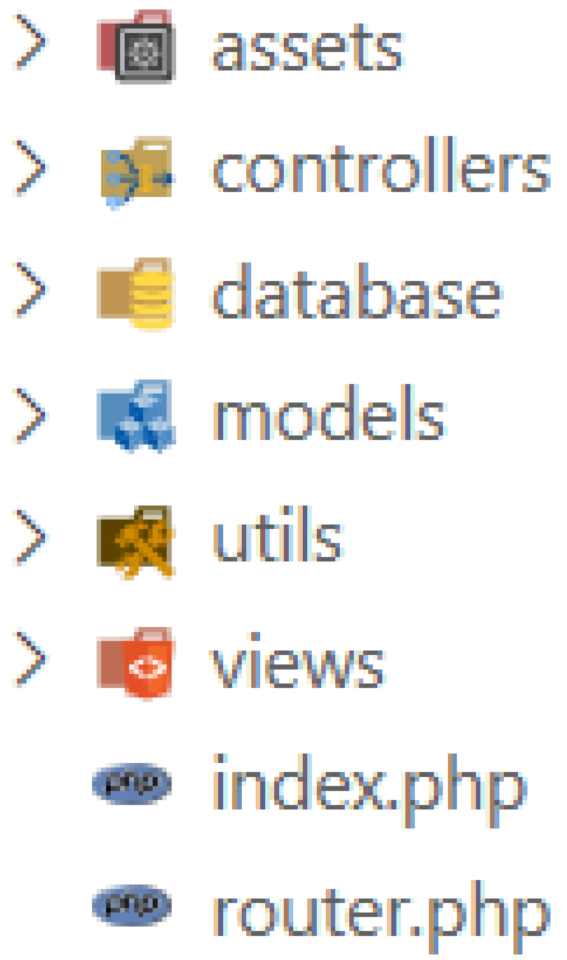
```
<?php foreach($posts as $post): ?>
      <li>
        <?= $post['title'] ?> |
        <span><?= $post['description'] ?></span> |
        <a href="controllers/post/post.edit.controller.php?id=<?= $post['id'] ?>">Edit</a>
      </li>

    <?php endforeach; ?>
```

3 – Create the HTML from the data

EXPLAIN

Clean project structure

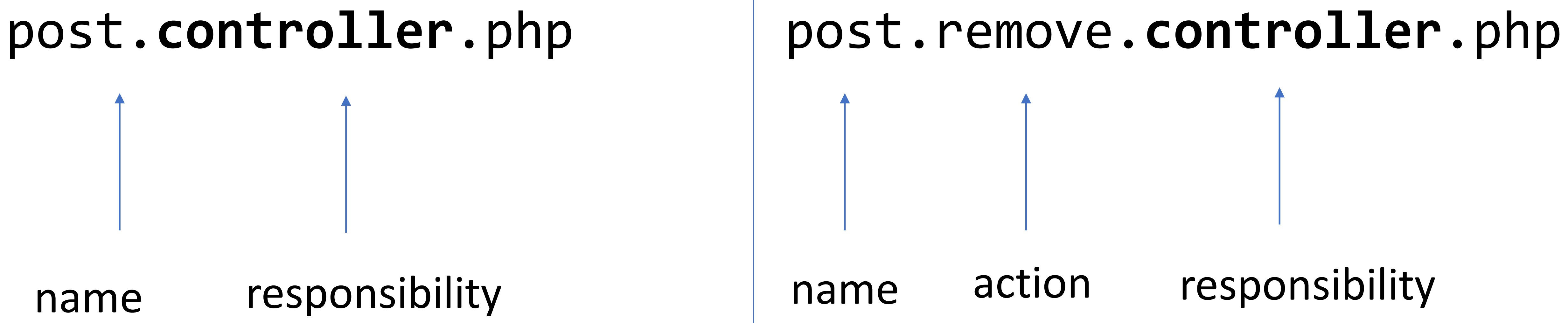


assets	Image, PDF, resources
controllers	**All controllers**
database	Connection to DB
models	**All models**
utils	Utility functions : debug, file, URL operations
views	**All views**
index.php	
router.php	The router to the controllers

EXPLAIN

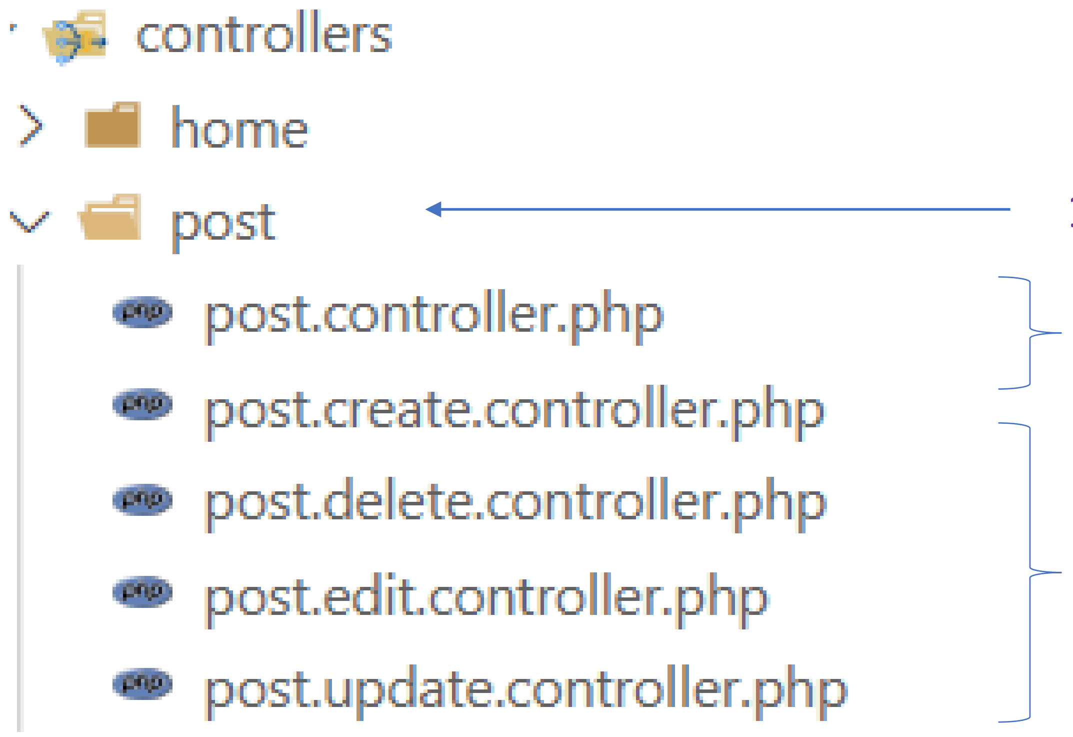
File names

✓ Use the **same syntax** in all project files



EXPLAIN

Controller structure


controllers
home
post
post.controller.php
post.create.controller.php
post.delete.controller.php
post.edit.controller.php
post.update.controller.php

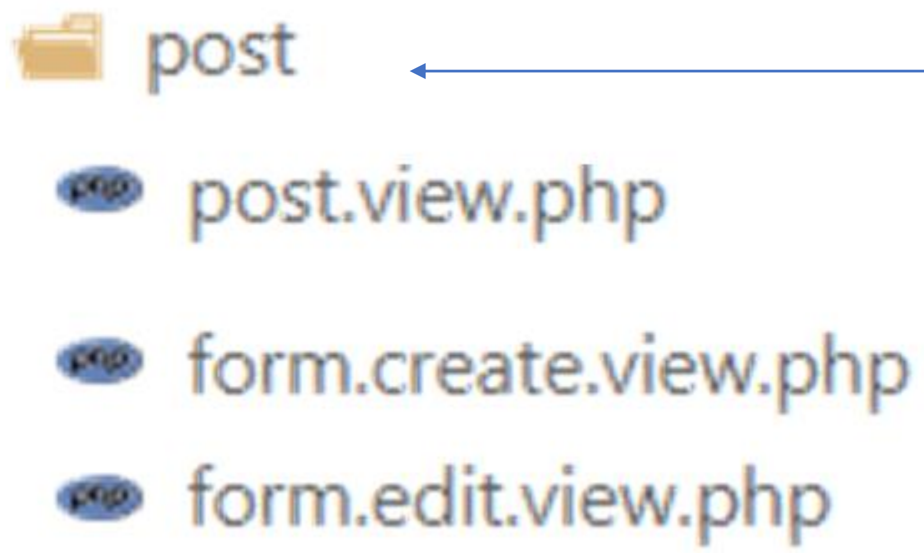
1 folder per **main view**

Main controller

Sub controller (actions or sub views)

EXPLAIN

Views structure



post

post.view.php

form.create.view.php

form.edit.view.php

1 folder per **main view**

Main view

Sub view