MVC Introduction

Model – View – Controller

greenwich.edu.vn

UNIVERSITY of GREENWICH

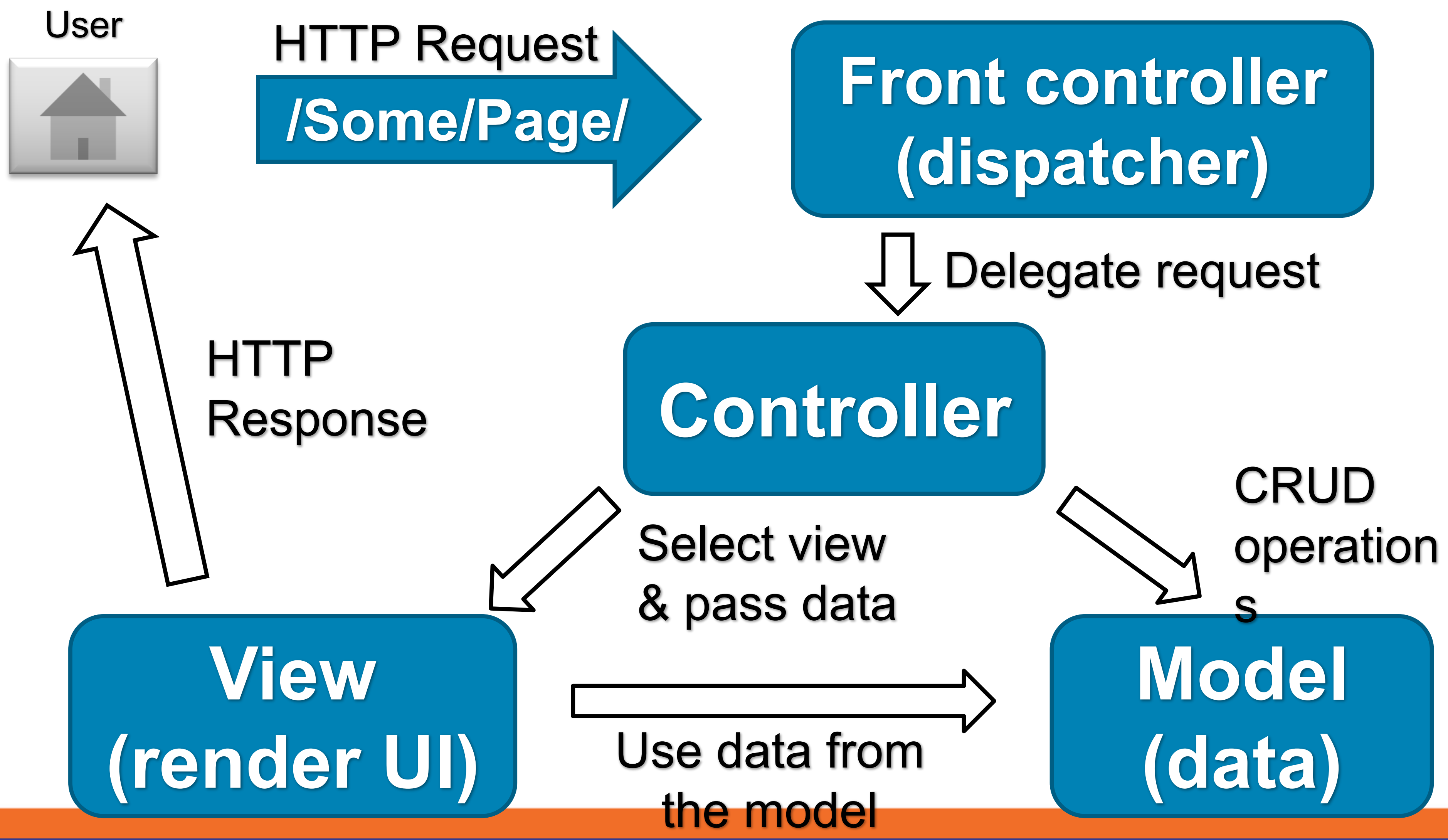
Alliance with FPT Education

Table of Contents

MODEL-VIEW-CONTROLLER

The MVC Pattern

MVC Concept Explained

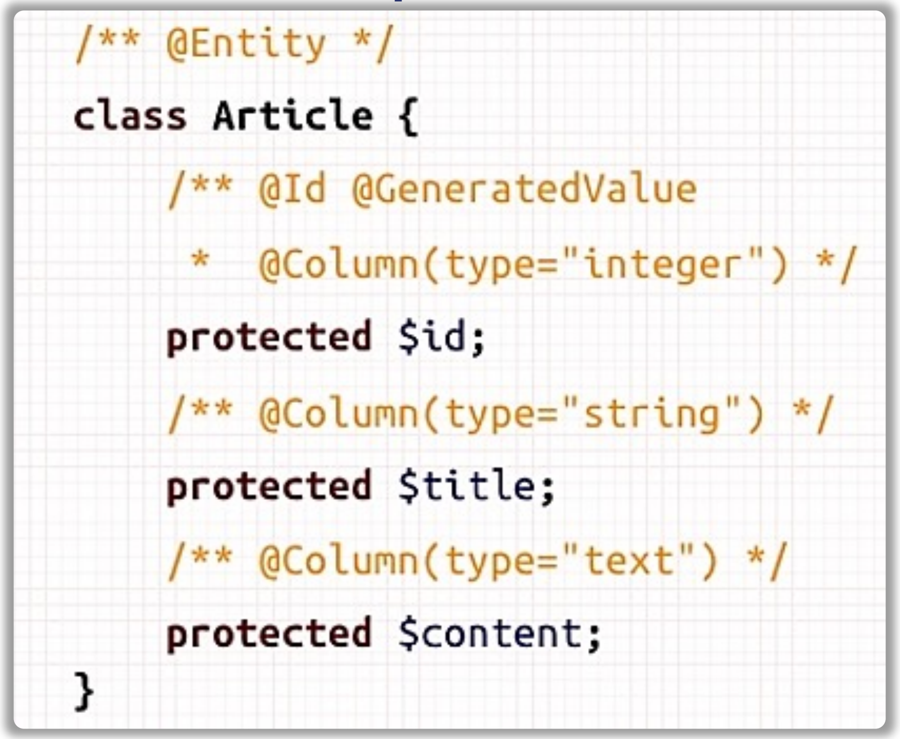


The MVC Pattern

- **Design pattern** with three independent components:
  - **Model** (data)
    - Manages **data** and **database** logic
  - **View** (UI)
    - **Presentation** layer (renders the UI)
  - **Controller** (logic)
    - Implements the **application logic**
    - Processes user request, performs an **action**, **updates** the data model and **invokes** a view to render some UI

Model (Data)

- Set of classes that describes the data we are working with
- Rules for how the data can be changed and manipulated
- May contain data validation rules
- Often encapsulates data stored in a database
  - As well as code used to manipulate the data
- E.g. Data Access Layer of some kind



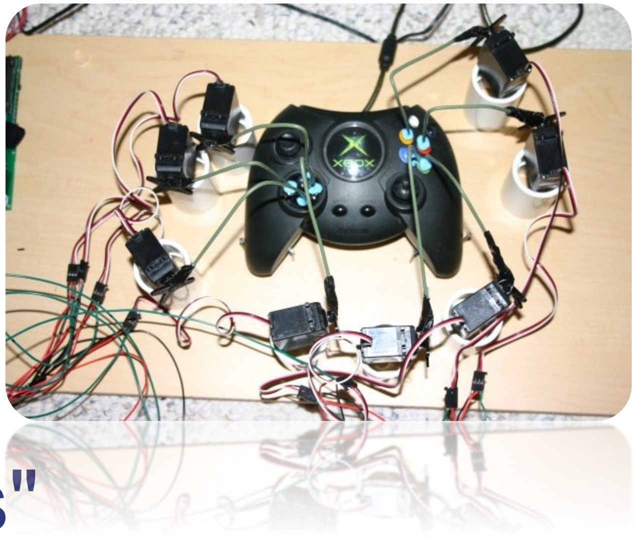
```
/** @Entity */
class Article {
    /** @Id @GeneratedValue
     *  @Column(type="integer") */
    protected $id;
    /** @Column(type="string") */
    protected $title;
    /** @Column(type="text") */
    protected $content;
}
```

View (UI)

- Defines how the application's user interface (UI) will be displayed
- May support master views (layouts)
- May support sub-views (partial views or controls)
- May use templates to dynamically generate HTML

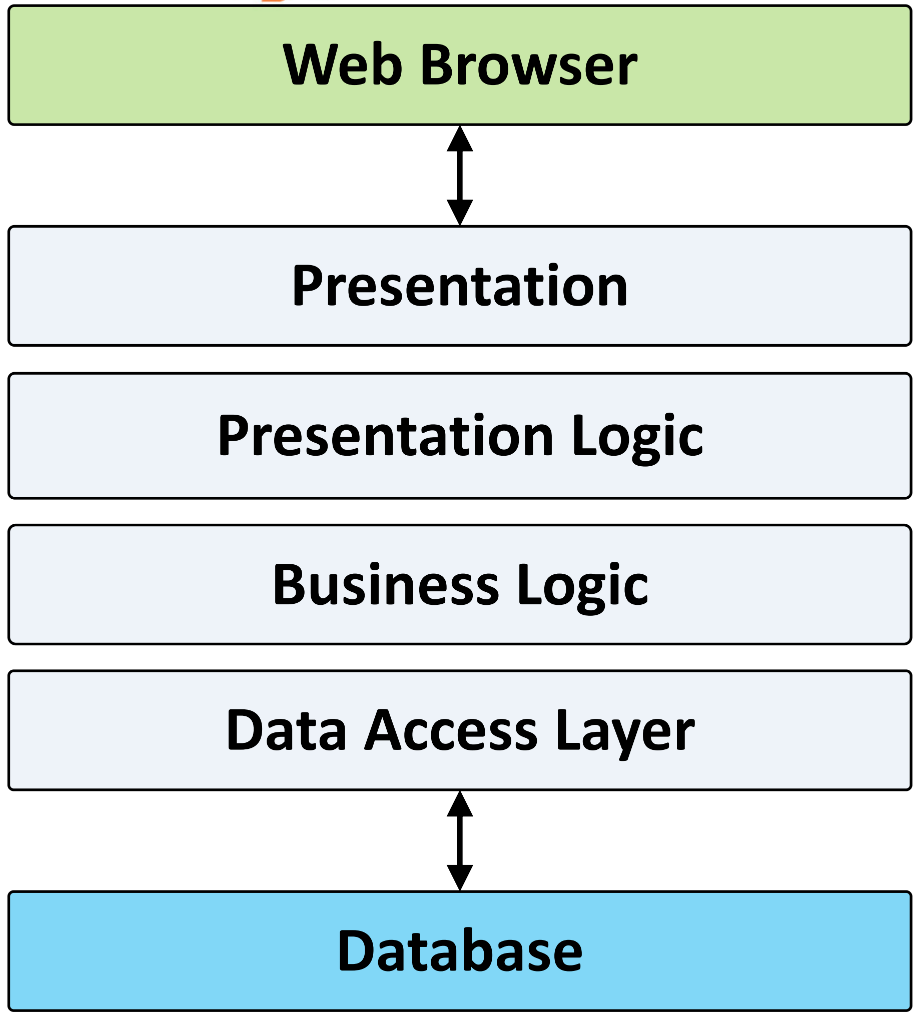
Controller (Logic)

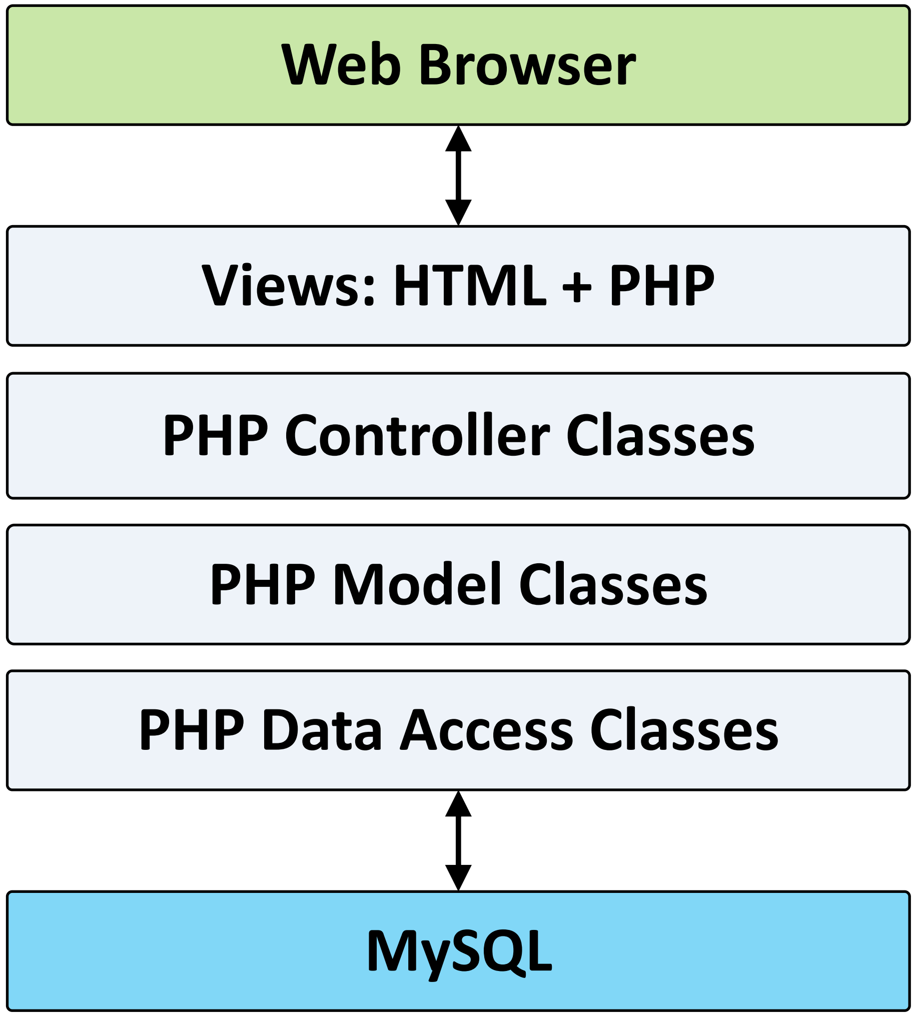
- The core MVC component – holds the logic
- Process the requests with the help of views and models
- A set of classes that handles
  - Communication from the user
  - Overall application flow
  - Application-specific logic (business logic)
- Every controller has one or more "actions"



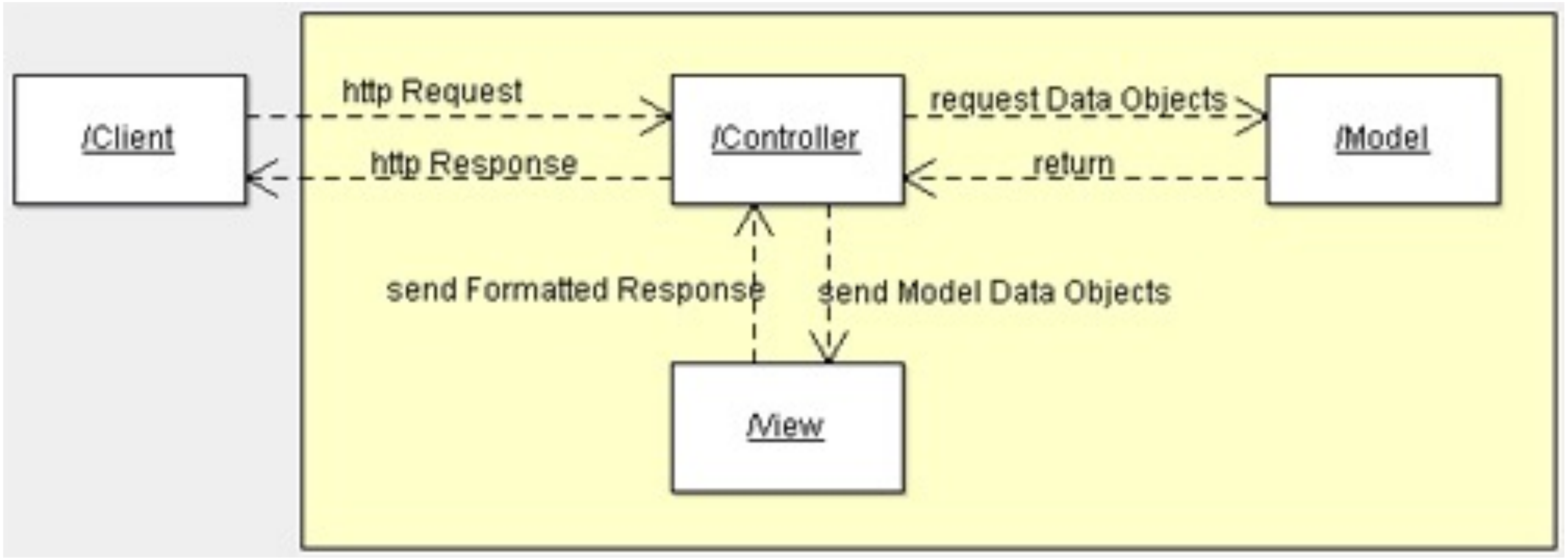
WEB APP STRUCTURE

Web and PHP App Architecture

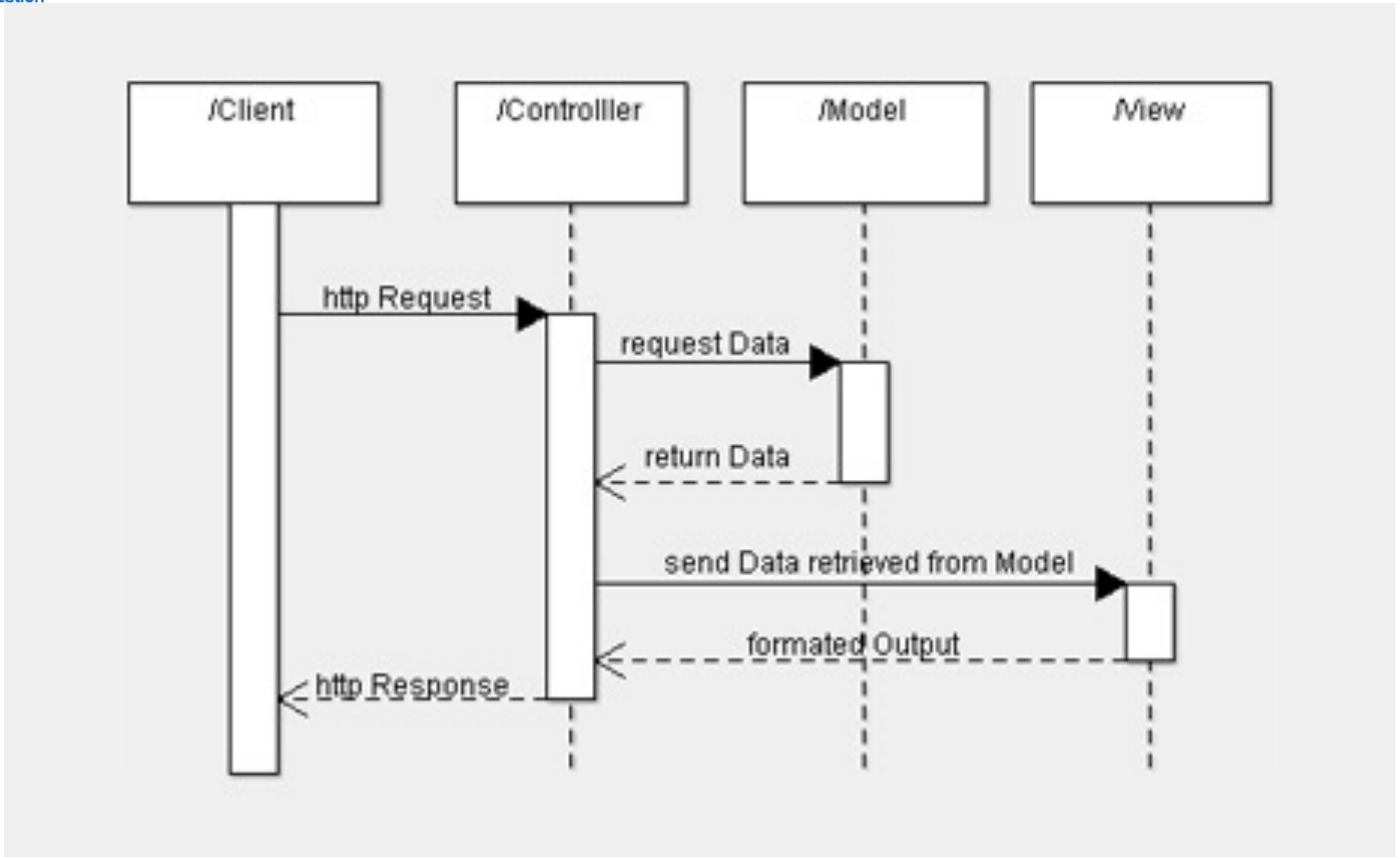




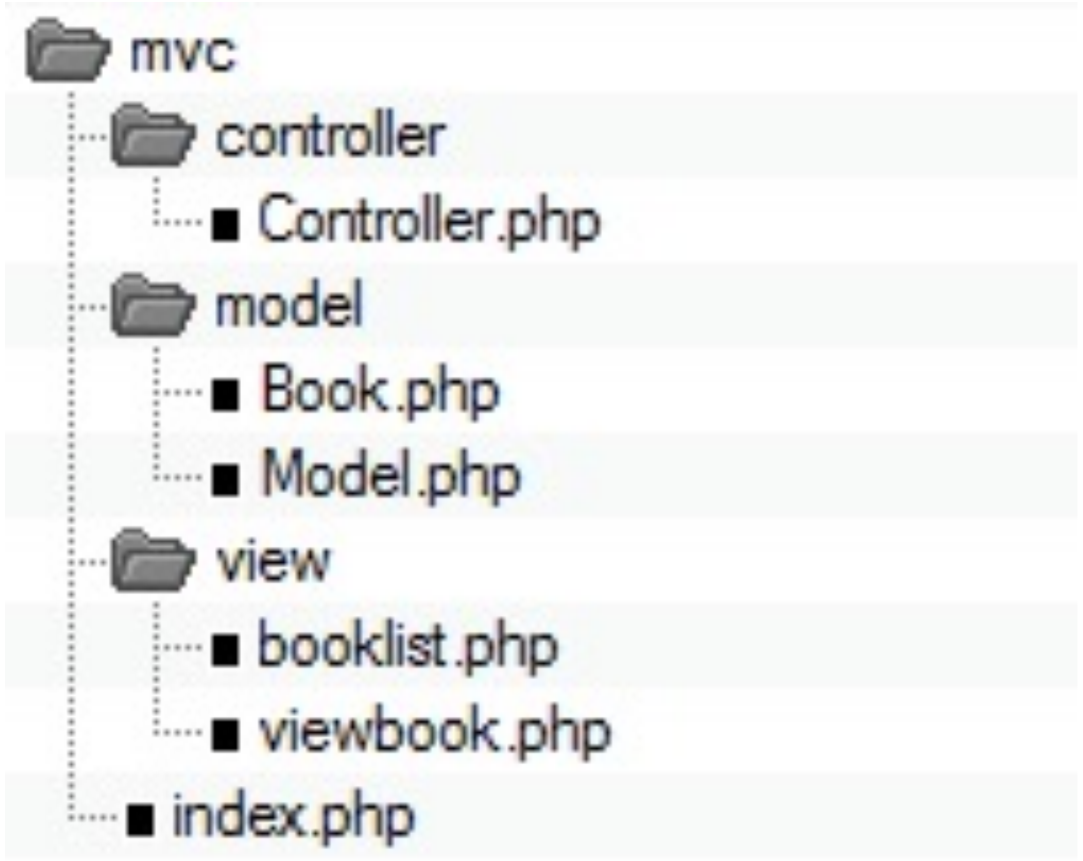
MVC Collaboration Diagram



MVC Sequence Diagram

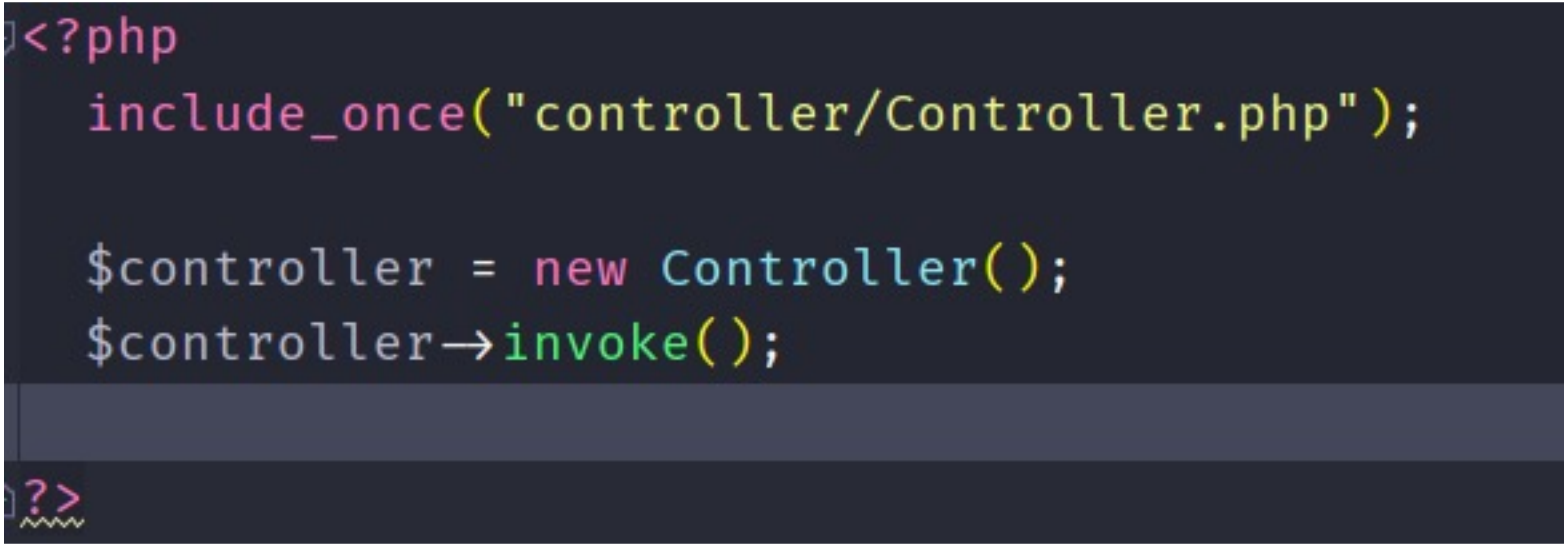


Simple MVC Folder Structure



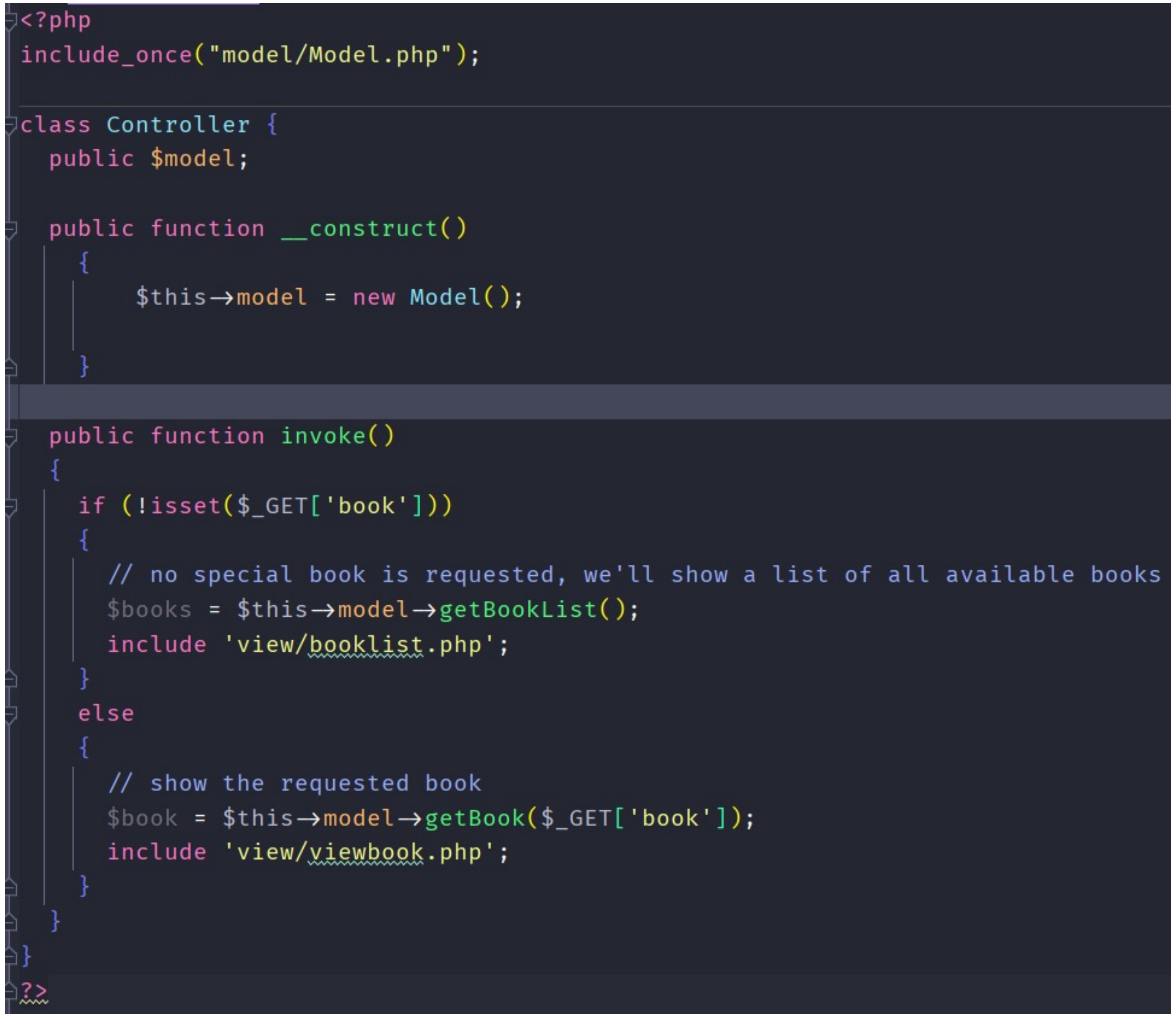
```
mvc
├── controller
│   └── Controller.php
├── model
│   ├── Book.php
│   └── Model.php
├── view
│   ├── booklist.php
│   └── viewbook.php
└── index.php
```

index.php



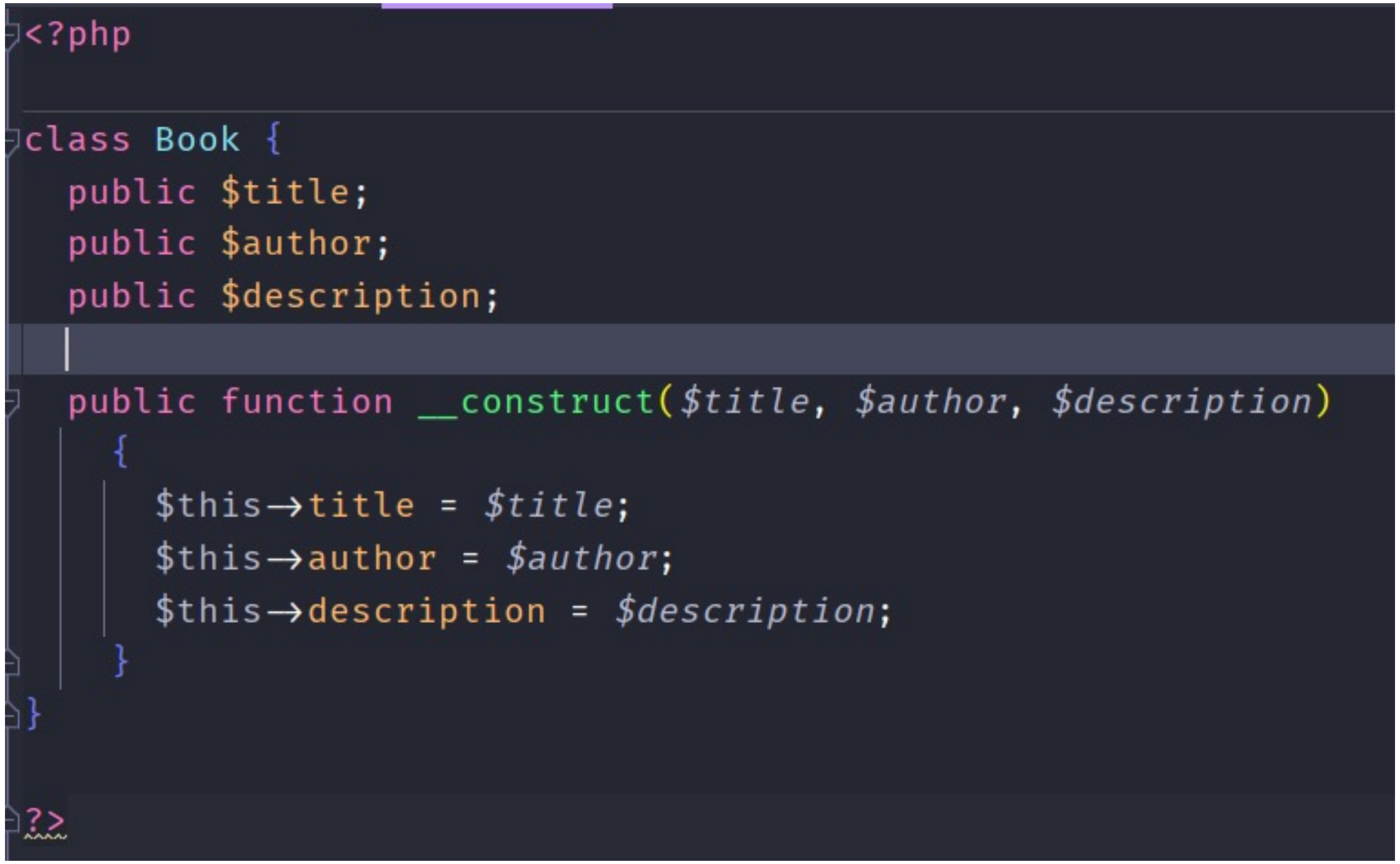
```php
<?php
   include_once("controller/Controller.php");

   $controller = new Controller();
   $controller->invoke();

?>
```

Controller.php



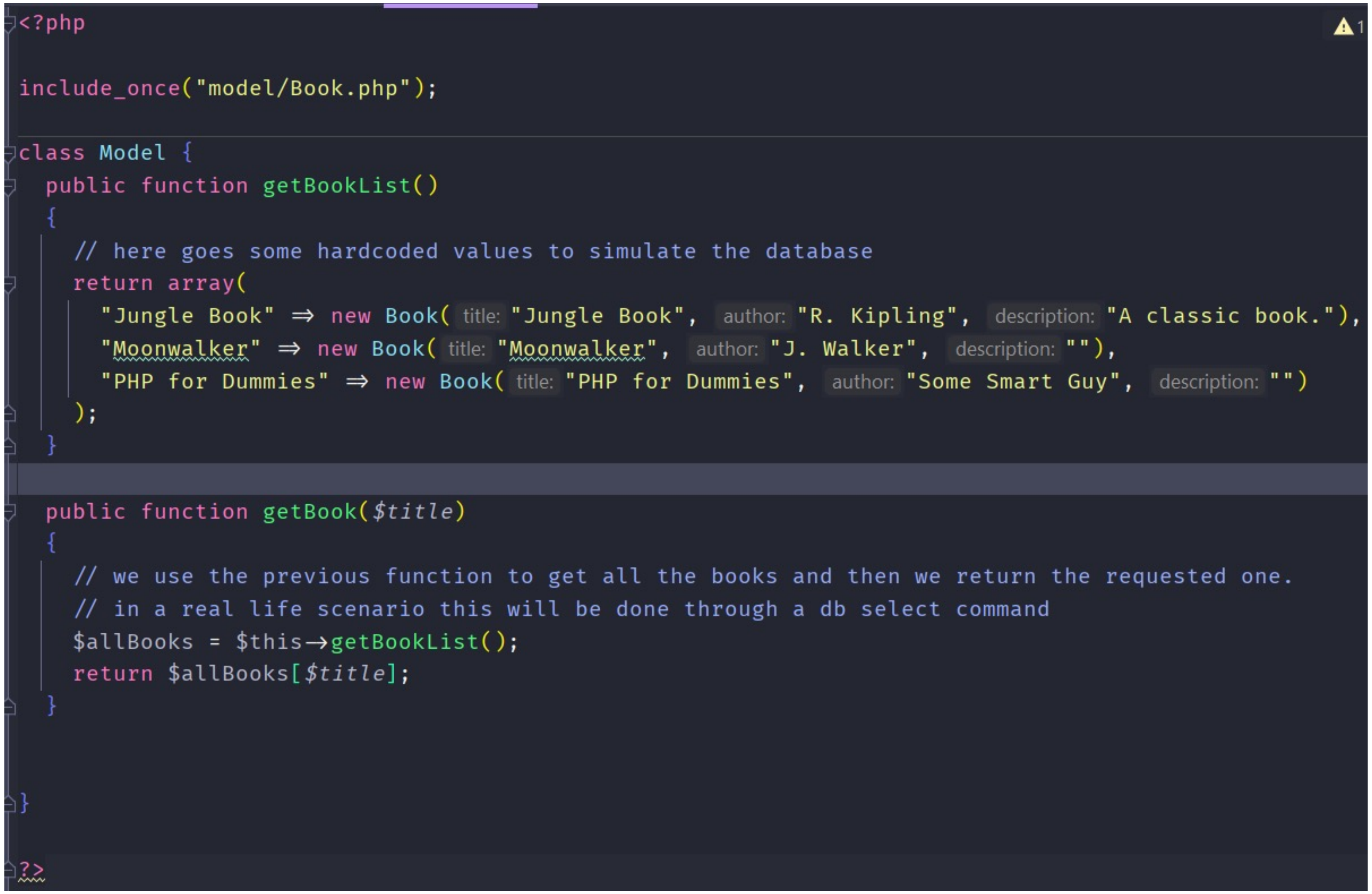
```php
<?php
include_once("model/Model.php");

class Controller {
  public $model;

  public function __construct()
    {
        $this->model = new Model();

    }

  public function invoke()
  {
    if (!isset($_GET['book']))
    {
      // no special book is requested, we'll show a list of all available books
      $books = $this->model->getBookList();
      include 'view/booklist.php';
    }
    else
    {
      // show the requested book
      $book = $this->model->getBook($_GET['book']);
      include 'view/viewbook.php';
    }
  }
}
?>
```

Book.php



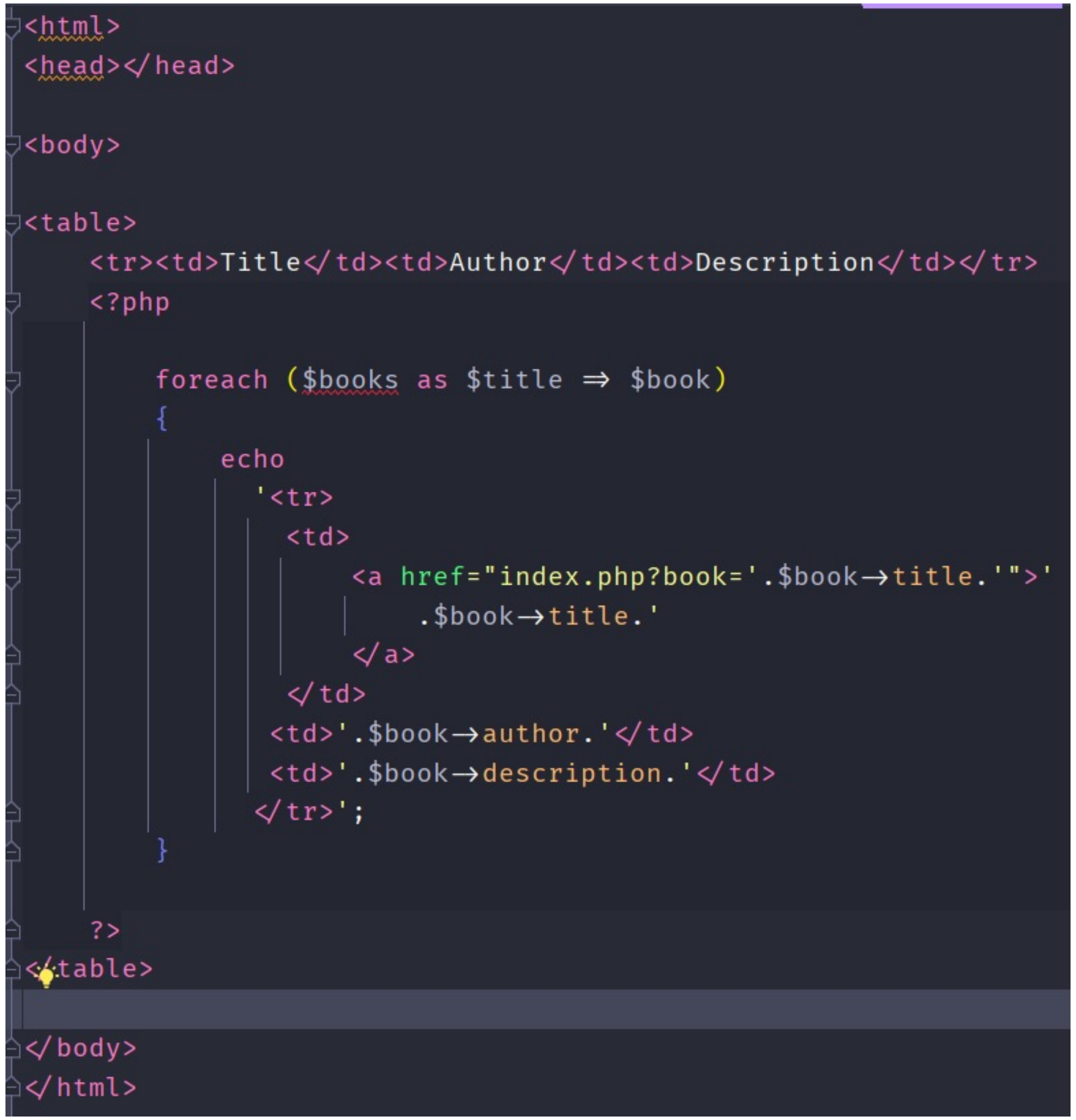
```php
<?php

class Book {
  public $title;
  public $author;
  public $description;

  public function __construct($title, $author, $description)
    {
      $this->title = $title;
      $this->author = $author;
      $this->description = $description;
    }
}

?>
```

Model.php



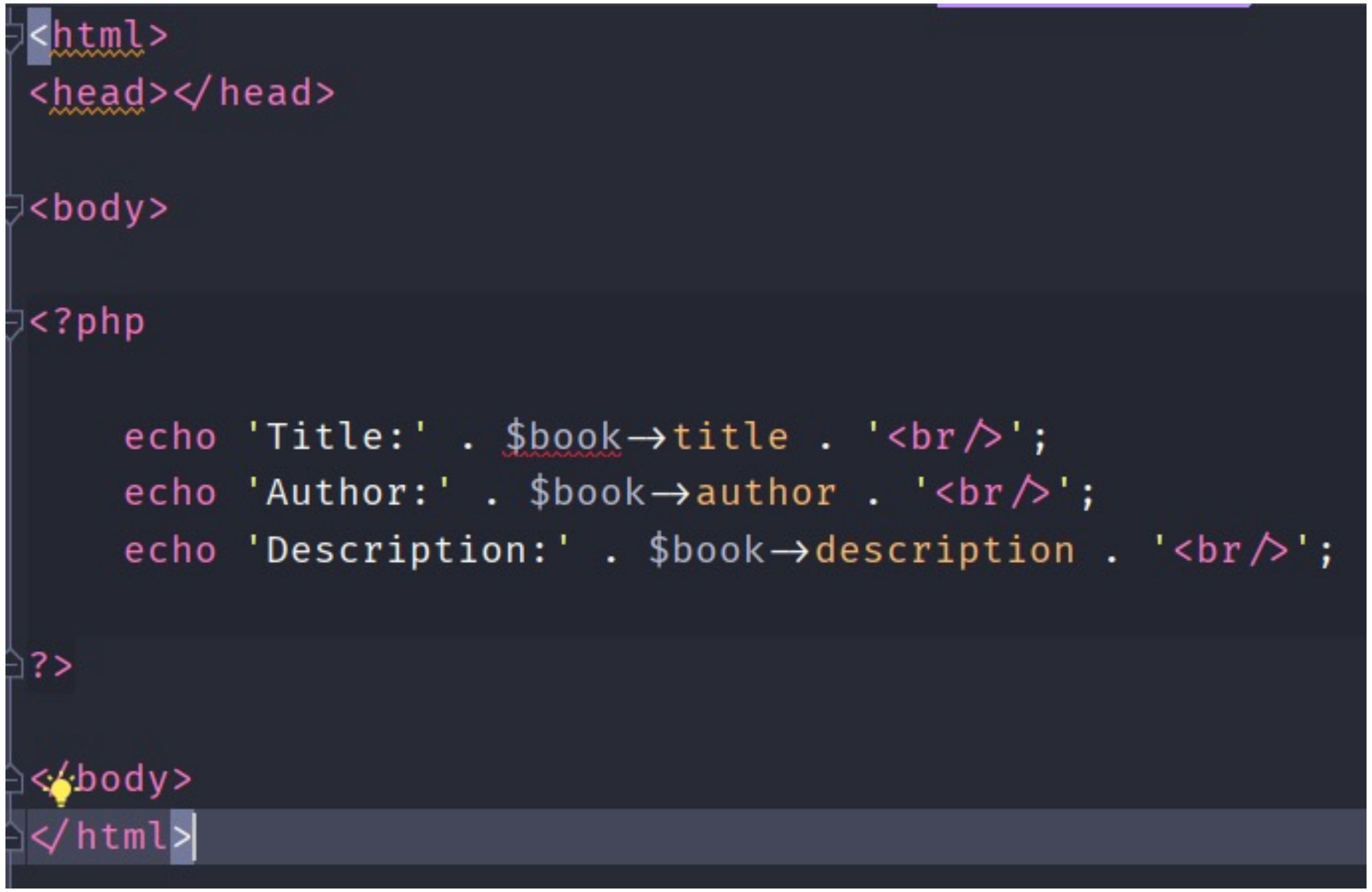
```php
<?php

include_once("model/Book.php");

class Model {
  public function getBookList()
  {
    // here goes some hardcoded values to simulate the database
    return array(
      "Jungle Book" ⇒ new Book( title: "Jungle Book",  author: "R. Kipling",  description: "A classic book."),
      "Moonwalker" ⇒ new Book( title: "Moonwalker",  author: "J. Walker",  description: ""),
      "PHP for Dummies" ⇒ new Book( title: "PHP for Dummies",  author: "Some Smart Guy",  description: "")
    );
  }

  public function getBook($title)
  {
    // we use the previous function to get all the books and then we return the requested one.
    // in a real life scenario this will be done through a db select command
    $allBooks = $this→getBookList();
    return $allBooks[$title];
  }


}

?>
```

booklist.php



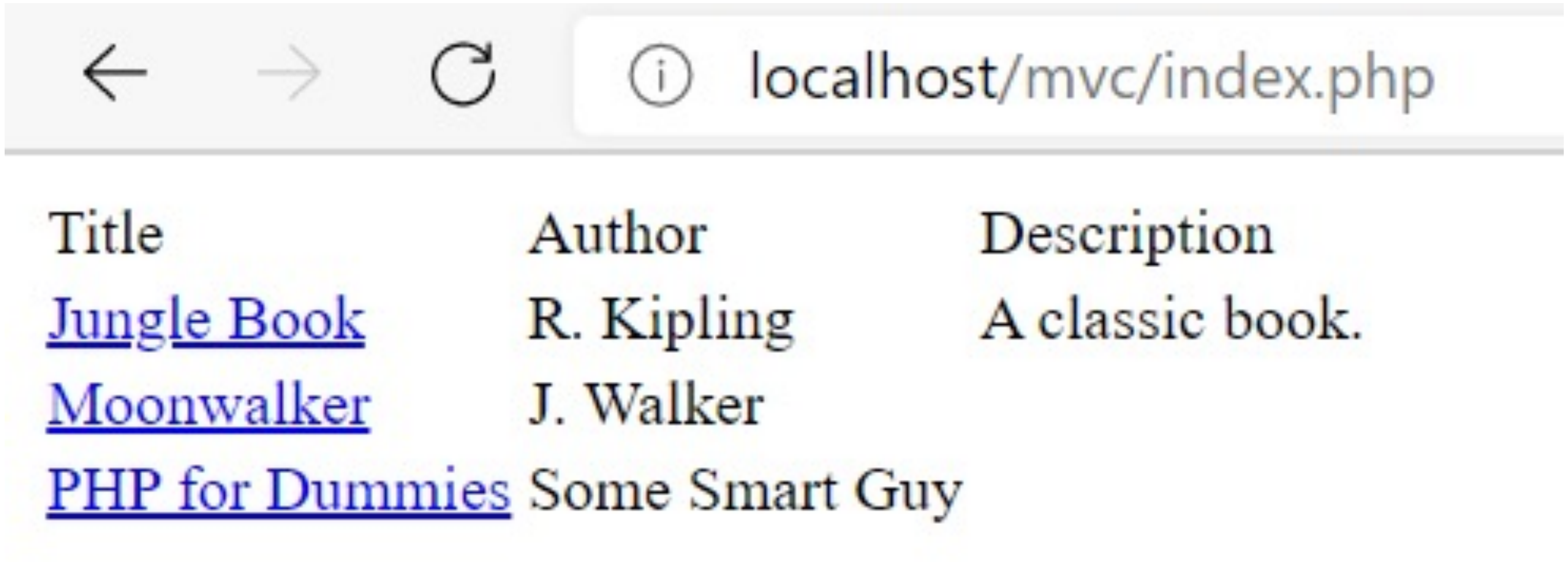
```
<html>
<head></head>

<body>

<table>
    <tr><td>Title</td><td>Author</td><td>Description</td></tr>
    <?php

        foreach ($books as $title => $book)
        {
            echo
              '<tr>
                <td>
                    <a href="index.php?book='.$book->title.'">'
                        .$book->title.'
                    </a>
                </td>
              <td>'.$book->author.'</td>
              <td>'.$book->description.'</td>
             </tr>';
        }

    ?>
</table>

</body>
</html>
```

viewbook.php



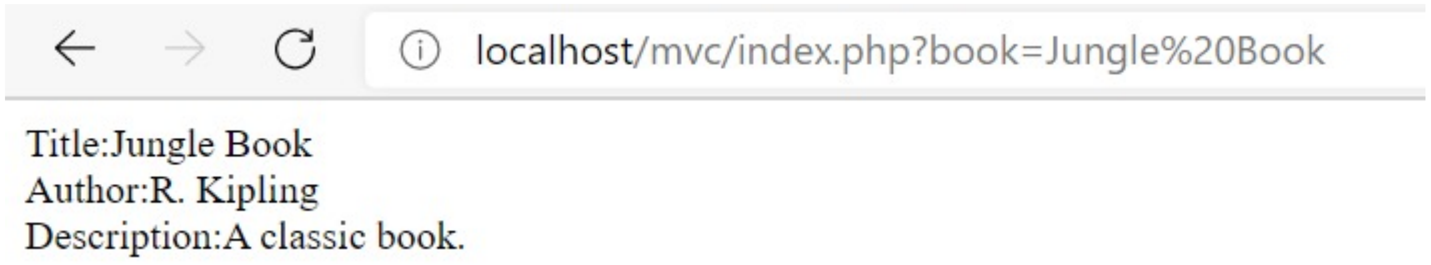
```
<html>
<head></head>

<body>

<?php

    echo 'Title:' . $book->title . '<br/>';
    echo 'Author:' . $book->author . '<br/>';
    echo 'Description:' . $book->description . '<br/>';

?>

</body>
</html>
```

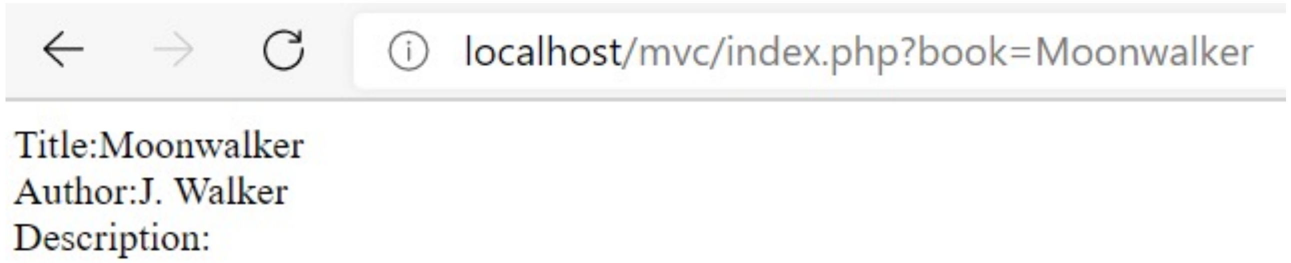
Booklist Page

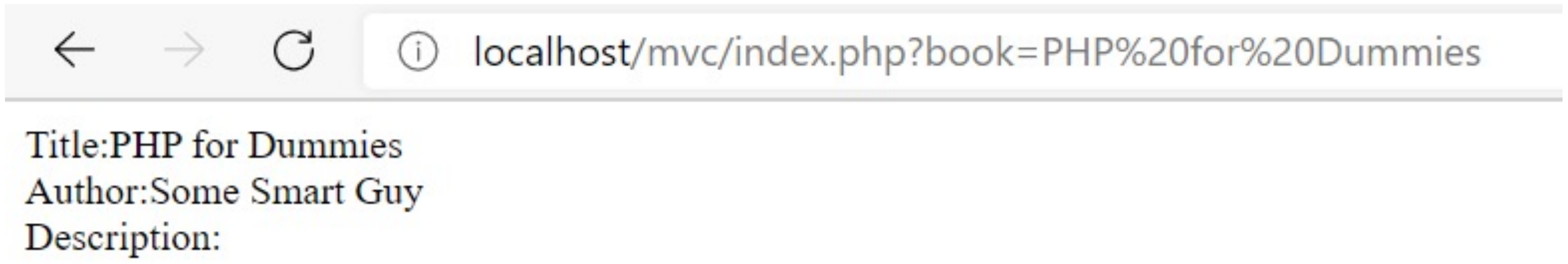

localhost/mvc/index.php

Title	Author	Description
Jungle Book	R. Kipling	A classic book.
Moonwalker	J. Walker	
PHP for Dummies	Some Smart Guy	

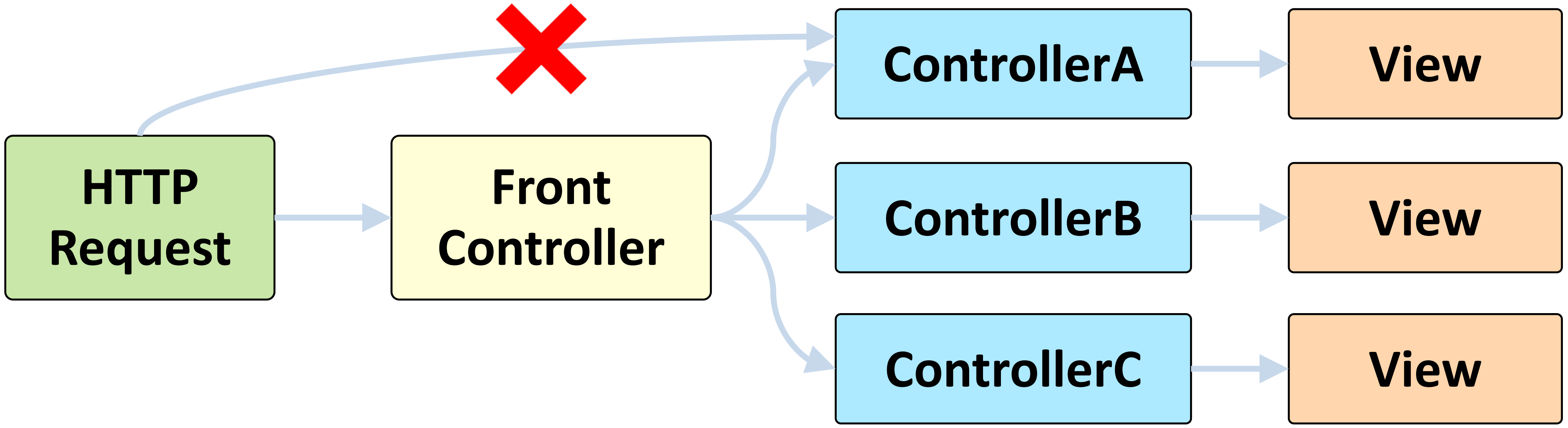
Viewbook page


```
localhost/mvc/index.php?book=Jungle%20Book

Title:Jungle Book
Author:R. Kipling
Description:A classic book.
```


```
localhost/mvc/index.php?book=Moonwalker

Title:Moonwalker
Author:J. Walker
Description:
```


```
localhost/mvc/index.php?book=PHP%20for%20Dummies

Title:PHP for Dummies
Author:Some Smart Guy
Description:
```

Front Controller



Data Access

Database Access Logic

ORM Approach

Map Tables to Classes

Direct Access

Data Access Classes

Data Binding

Data Binding

Form Data

Map Form Data to Objects

View Data

Map Variables to ViewModels

Summary

- MVC Design Pattern
- Web Structure
- PHP Structure