Рефакторинг – стандартне розділення архітектурних рівнів

Дано сторінку, яка відображає деякий книжковий каталог; для кожної книги відображається її автор, назва та рік видання. При цьому книжки, видані після 2010 року, повинні відображатися жирним шрифтом. Використовується PHP:

<html>

<head>

<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=windows-1251">

<title> No template page </title>

</head>

<body bgcolor="white">

<center>

<font color="green">

<h1>BOOK CATALOGUE</h1>

</font>

</center>

<br/><br/>

<?php

class Book {

public $author;

public $title;

public $year;

function \_\_construct($author, $title, $year) {

$this->author=$author;

$this->title=$title;

$this->year=$year;

}

}

$books = array(

new Book("Ivanov","Addicted to murders",2018),

new Book("Petrov","The art of web programming",1901),

new Book("Sidorov","How to cheat",1515)

);

foreach ($books as $book) {

if ($book->year>2010) echo "<b>";

echo $book->author.' - '.$book->title.' - '.$book->year.'<br/>';

if ($book->year>2010) echo "</b>";

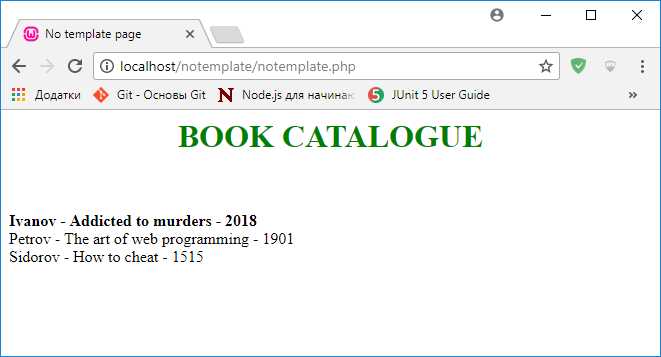
}

?>

</body>

</html>

Цей код відображається наступним чином:



Завдання:

Показати недоліки наведеного рішення. Реорганізувати його в наступних напрямках:

- уникати deprecated-тегів (зокрема, <font>); оформлення сторінки здійснювати на основі таблиць стилів;

- повністю відокремити рівень представлення від бізнес-логіки – зокрема, на сторінці, яка призначена для відображення, не повинно бути PHP-коду; для такого відокремлення використати Smarty або Twig;

- дані про книжки зберігати в базі даних MySQL в таблиці books;

- відокремити бізнес-логіку від рівня даних на основі DAO та VO; всі звернення до даних повинні здійснюватися лише через DAO; повинна здійснюватися нелінива ініціалізація DAO, тобто при кожному зверненні всі дані з бази даних повинні завантажуватися в DAO; в класі DAO мають бути функції ініціалізації даних та отримання всіх книжок.

Бонус. Реалізувати варіант цієї вправи на Laravel.