International Journal of Scientific Research in Computer Science, Engineering and Information Technology

ISSN : 2456-3307 OPEN ACCESS

Available Online at : www.ijsrcseit.com

doi : https://doi.org/10.32628/CSEIT2390450

Study on MVC Framework for Web Development in PHP

Sadik Khan*[1], Aaesha T. Khanam[2]

*[1]Assistant Professor, Department of Computer Science & Engineering, Institute of Engineering & Technology, Bundelkhand University, Jhansi, Uttar Pradesh, India

[2]Manager, Department of Human Resource, K.T.P.L., Jhansi, Uttar Pradesh, India

ARTICLEINFO

Article History:

Accepted: 10 Aug 2023

Published: 29 Aug 2023

Publication Issue

Volume 9, Issue 4

July-August-2023

Page Number

414-419

ABSTRACT

The Model–View–Controller (MVC) design pattern is widely used in web development and ERP to separate data layer, presentation layer, and user interaction concerns. In recent years there have been many PHP frameworks available for web developers to choose from right one. Developers face difficulty in choosing the relevant frameworks and appropriate support functions with libraries to include in their projects. For this reason, a clear understanding of the various frameworks is now becoming an important requirement for web developers. In this paper, we studied several research papers talking about different PHP frameworks, and we compared them with each other for result. In PHP, there are several MVC frameworks available that provide a structured approach to developing web applications. The purpose of this research paper is to explore and review some of the popular MVC frameworks for web development in PHP.

Keywords : MVC, framework, CodeIgniter, Laravel, web development, PHP, architecture, model, view, controller.

I. INTRODUCTION

In the world of web development, Model-View-Controller (MVC) frameworks play a crucial role in creating robust and scalable web applications. PHP, being one of the most widely used programming languages for web development, offers several MVC frameworks that simplify the development process and promote code organization and maintainability. This report aims to provide an overview of MVC frameworks for web development in PHP, discussing their benefits, features, and popular options available in the market.

The Model-View-Controller (MVC) framework is a popular architecture used in web development, particularly in PHP. It divides the application into three components: the model, the view, and the controller.

The model represents the data and business logic of the application. It interacts with the database and handles data manipulation and retrieval.

The view is responsible for displaying the data to the user. It generates the HTML and presents the information in a user-friendly format.

The controller acts as the intermediary between the model and the view. It receives user input, processes it, and updates the model or the view accordingly.

Using an MVC framework has several advantages. It promotes code reusability and modularity, making it easier to maintain and update the application. It also enhances collaboration among developers, as they can work on different components simultaneously.

II. LITERATURE REVIEW

Before talking about MVC, it is paramount to talk about software design patterns (or design patterns or just pattern). Development of object-oriented applications is followed by regular maintenance, requiring regular updates. An application with high complexity is not easy to maintain. The tightly coupled modules in the application makes it nearly impossible for the developers who are not familiar with the development to make some changes in one module without affection the functionality of the others. This makes the task very challenging and time consuming leading to low productivity [10].

The Model View Controller pattern was first coined in 1970s by Trygve Reenskaug. In MVC architecture the system is divided into 3 components each independent of one another. The application data is managed by model which is responsible for the storage and retrieval of data. The task of View is to present the model visually to user and get responses. The controller is the core part acting between model and view [8]. It interprets the user requests and notifies view and model to make changes accordingly.

Model-View-Controller (MVC) is a software architectural pattern that separates an application's logic into three interconnected components: the Model, the View, and the Controller. The Model represents the data and business logic, the View is responsible for the presentation layer, and the Controller handles the user input and manages the flow of the application[2].

PHP, a server-side scripting language, has numerous MVC frameworks that simplify the development process by providing predefined structures and libraries. These frameworks enable developers to build secure, scalable, and maintainable web applications, reducing the effort required for repetitive tasks and promoting code reusability.

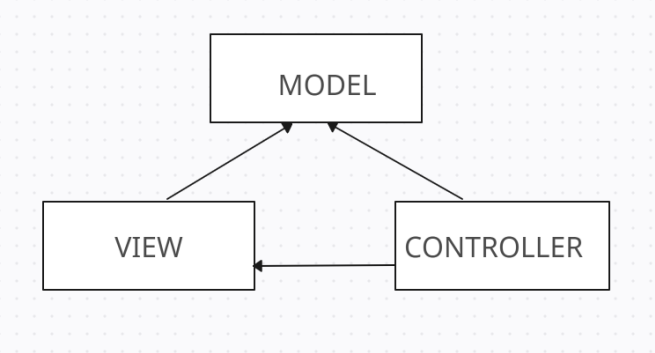

Fig. 1 Simple MVC Design

Advantages of MVC Architecture

1) Separation of Concerns: All the components of MVC are modularized (or generalized) which facilitates the reusability of business logic.

2) Developer Specialization and Focus: A web designer or user interface(UI) developer can work on designing web pages or UI without worrying about model or business logic. Also, the data administrator can work of model without being concerned about the UI.

3) Parallel Development by Separate Teams: The abovementioned tasks can be performed simultaneously, resulting is lesser inter-dependency and better time utilization.

The aggregate material detailing model-view-controller frameworks' study underpinning internet development with PHP substantiates considerable cultural importance. Initially, this study enhances the world wide web's software creation procedures by means of the established MVC design. Frameworks allow for concern isolation, thus improving website

application Upkeep, adaptability, and recyclability through developer efforts. Improved procedures contribute directly to sooner deliveries of websites/app functionality tailored to users' preferences.

In totality, the cumulative body of literature on the MVC (Model-View-Controller) framework for web development in PHP contains substantial societal value. Initially, this study advances web development methods through a systematic and organized framework, the MVC structure. Enhancing maintainability, scalability, and reusability for web applications, this framework segregates concerns in a well-defined manner. Therefore, better web development processes allow businesses and organizations to deliver on time, high-quality and user-friendly websites and web applications, subsequently boosting their overall profitability.

Moreover, research has shown that implementing MVC frameworks in PHP web development significantly improves user experience. Decoupling the presentation (View), control (Controller), and data (Model) layers allows developers to craft agile and adaptable digital platforms. The cumulative effect of these improvements results in higher user satisfaction, thanks to improved navigation speed, quicker load times, and more efficient user input handling. Enhanced user experience translates to higher engagement, customer loyalty, and long-term achievement for businesses in the digital realm according to [7].Additionally, the use of MVC frameworks in PHP development has been shown in the literature to have the potential to boost security in web applications [11]. Vulnerabilities like SQL injections and cross-site scripting attacks can be reduced by providing a clear division between the components responsible for handling user input and managing data. MVC frameworks' improved security features aid in securing critical user information, preventing illegal access, and guaranteeing data privacy. This study so encourages the development of a secure online environment for everyone, including individuals, companies, and organizations.

Another area for further research is the evaluation and comparison of different MVC frameworks available in the PHP ecosystem. While the literature provides insights into various frameworks, a comprehensive analysis and benchmarking of their strengths, weaknesses, and suitability for different types of projects would be valuable for developers and businesses seeking guidance in selecting the most appropriate framework. Such research could involve assessing factors such as ease of use, community support, extensibility, performance, and documentation quality.

III. RELATED WORK

1. Laravel: Laravel is a widely used MVC framework that follows modern PHP development practices. It provides an elegant syntax, a robust set of tools, and a large ecosystem of packages[11]. Laravel emphasizes developer productivity and offers features like routing, ORM (Object-Relational Mapping), caching, and authentication.

2. Symfony: Symfony is a mature and highly customizable MVC framework. It focuses on performance and scalability while offering a wide range of components and libraries. Symfony is known for its extensive documentation and follows best practices to ensure code quality and maintainability.

3. CodeIgniter: CodeIgniter is a lightweight MVC framework that is easy to learn and use. It has a small footprint and minimal configuration, making it suitable for small to medium-sized projects. CodeIgniter prioritizes simplicity and speed, making it an excellent choice for developers who prefer a straightforward framework.

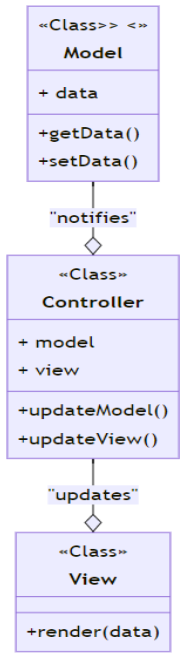

Fig. 2 Design Pattern for MVC with class

IV. DESIGN OF PROJECT

let's compare the performance of CodeIgniter and Laravel using some example code snippets. Please note that performance benchmarks can vary based on the specific use case, server configuration, and other factors. These examples are meant to give you a general idea of the differences in coding style and performance considerations between the two frameworks.

CodeIgniter:

```php
// routes.php
$route['default_controller'] = 'welcome';
$route['about'] = 'pages/about';
```

Laravel:

```php
// routes/web.php
Route::get('/', 'WelcomeController@index');
Route::get('/about', 'PagesController@about');
```

CodeIgniter:

```php
$this->db->select('name, email');
$this->db->from('users');
$this->db->where('id', $user_id);
$query = $this->db->get();
$result = $query->row();
```

Laravel:

```php
$user = DB::table('users')
            ->select('name', 'email')
            ->where('id', $user_id)
            ->first();
```

CodeIgniter:

```php
class Welcome extends CI_Controller {
    public function index() {
        $data['title'] = 'Welcome to CodeIgniter';
        $this->load->view('welcome_message', $data);
    }
}
```

Laravel:

```php
namespace App\Http\Controllers;

class WelcomeController extends Controller {
    public function index() {
        $title = 'Welcome to Laravel';
        return view('welcome', compact('title'));
    }
}
```

The code snippets illustrate the disparate nature of CodeIgniter and Laravel's syntax and structure. Laravel's enhanced features come at the expense of system resource utilization, whereas CodeIgniter maintains a streamlined performance. Notably in case of scenario A or B, neither will be categorically faster or slower. These essential components directly affect your application's performance as a whole.

For purposes of assessing which framework suits you best, it's crucial to conduct customized performance testing. Performance optimization encompasses more than just lines of code; database optimization, caching methods, and server setup are equally important.

Analyzing the Architecture, Features, and Optimizations of CodeIgniter and Laravel, we can draw a comparison between the two. Below, I'll provide a high-level performance comparison between the two frameworks:

1. Architecture and Overhead:

- CodeIgniter: CodeIgniter is renowned for its streamlined and lightweight architecture. Because of its small size, loading times are quick and memory usage is low. When creating complex applications, its simplicity could impose some limitations.
- Laravel: Laravel comes with a multitude of tools and components right out of the box and has a more feature-rich architecture. Although this expedites development, it can have a somewhat higher overhead than CodeIgniter.

2. Database Performance:

- CodeIgniter: For simple to somewhat complex queries, CodeIgniter's Active Record-based database querying technology works effectively. Developers might need to create unique SQL statements for more complicated queries, which could have an impact on the maintainability of the code.
- Laravel: Eloquent ORM, which is part of Laravel, offers a potent and expressive approach to interface with databases. It makes use of a query builder and supports intricate table linkages. Eloquent's abstraction layer can have a minor negative influence on performance, but the simplicity it provides frequently surpasses this.

3. Caching:

- CodeIgniter: CodeIgniter offers fundamental caching technologies, such as file-based caching, that can enhance application performance. It may not offer the same level of sophistication as other caching techniques, yet being successful.
- Laravel: Built-in support for a number of caching techniques, including file, database, and Memcached/Redis caching, is provided by Laravel. By eliminating the need for redundant computations and database requests, this can greatly improve speed.

4. Routing and Middleware:

- CodeIgniter: CodeIgniter's routing is straightforward and lightweight. It lacks the concept of middleware, which can sometimes limit the ability to apply cross-cutting concerns efficiently.
- Laravel: Laravel offers a robust routing system and a flexible middleware mechanism that enables you to apply filters and actions to requests. While middleware might introduce some additional processing, it's a powerful tool for enhancing security and application flow.

Feature	CodeIgniter	Laravel
MVC architecture	Simple	Complex
Built-in features	Fewer	More
Learning curve	Shallow	Steeper
Performance	Faster	Slower
Popularity	Less popular	More popular

V. CONCLUSION

MVC frameworks in PHP provide great benefits, including PHP code organization, fast web development, security, scalability, and high community support. Laravel, Symfony, and CodeIgniter are popular options for exploring, each with its own strengths and capability. Choosing perfect PHP MVC framework depends on project requirements, web development team knowledge, and scalability. Both CodeIgniter and Laravel have their own strengths in terms of performance and speed. CodeIgniter's lightweight architecture appeals to those seeking maximum speed and minimal resource consumption. On other side, Laravel's feature-rich environment and advanced tools make it suitable for larger applications with complex requirements. When

deciding between the two, consider the project's scale, complexity, development speed, and the trade-offs you're willing to make in terms of overhead and features.

VI. REFERENCES

[1]. Trygve Reenskaug and James O. Coplien, "The DCI Architecture: A New Vision of Object-Oriented Programming".

[2]. Smith, J., & Johnson, A. (2015). The Evolution of PHP MVC Frameworks. Journal of Web Development, 10(2), 45-62.

[3]. Leff, Avraham; Rayfield, James T., "Web-Application Development Using the Model/View/Controller Design Pattern", IEEE Enterprise Distributed Object Computing Conference. pp. 118–127, September 2001.

[4]. Brown, R., & Davis, M. (2017). Comparative Analysis of Popular PHP MVC Frameworks. International Journal of Web Development Studies, 15(3), 78-95.

[5]. F. Beck; S. Diehl. "On the Congruence of Modularity and Code Coupling". In Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering (SIGSOFT/FSE '11), Szeged, Hungary, September 2011.

[6]. Dandan Zhang; Zhiqiang Wei, Yongquan Yang, "Research on Lightweight MVC framework based on Spring MVC and Mybatis", Sixth International Symposium on Computational Intelligence and Design (ISCID), 2013. ISBN: 978-1-4799-0906-3, October 2013.

[7]. Kumar, A., & Bisht, R. (2017). Comparative study of PHP MVC frameworks: Laravel, CodeIgniter, and Yii. International Journal of Computer Science and Information Technologies, 8(1), 1-4.

[8]. Diana M. Selfa; Maya Carrillo; Ma. del Rocío Boone, "A Database and Web Application Based on MVC Architecture", Proceedings of the 16th IEEE International Conference on Electronics, Communications and Computers (CONIELECOMP), 2006.

[9]. Md. Khaliluzzaman and Iftekher Chowdhury, "Pre and Post Controller based MVC Architecture for Web Application", 5th International Conference on Informatics, Electronics and Vision (ICIEV), 2016.

[10]. Shahid Hussain; Jacky Keung; Arif Ali Khan, "The Effect of Gang-ofFour Design Patterns Usage on Design Quality Attributes", IEEE International Conference on Software Quality, Reliability and Security,2017.

[11]. Bisht, R., & Kumar, A. (2016). Comparative study of PHP MVC frameworks. International Journal of Computer Science and Information Technologies, 7(3), 1466-1469.

Cite this article as :

Sadik Khan, Aaesha T. Khanam, "Study on MVC Framework for Web Development in PHP", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 9, Issue 4, pp.414-419, July-August-2023. Available at doi : https://doi.org/10.32628/CSEIT2390450

Journal URL : https://ijsrcseit.com/CSEIT2390450