A Custom Framework for Innovative Approach of Teaching Web Development Courses

Patrik Hrkút, Matej Meško, Michal Ďuračík
University of Žilina
Žilina, Slovakia
Patrik.Hrkut@fri.uniza.sk, Matej.Mesko@fri.uniza.sk, Michal.Duracik@fri.uniza.sk

***Abstract* — The development of web applications is very complicated these days, as it requires knowledge of a number of technologies in order to create a full-featured application. Students, especially those who are beginners in this field, have problems in connecting all the necessary technologies and languages together. The core of every web application is its server part. where professional frameworks are often used, which shorten the time needed to create an application for experienced developers, but for students who are just learning, they represent an overly complex system, very difficult to master. Therefore, it is best not to use such frameworks for teaching, but to prepare for them simpler and easier to understand tools that they will use when creating their semester projects. In this article, we deal with the creation and description of experiences with our own framework, which we created for the students of our Internet application development course. In it, we describe what causes the biggest problems for students, how we solved them in our framework, and at the same time we provide an evaluation of how the framework has proven itself over several years of use. Also, part of the paper is an evaluation of how students used the framework. The main criterion for evaluation was how many students preferred our framework to others when developing a semester project. We tested the practical use of the framework in web applications for transport, intelligent transport systems and communication systems between vehicles to prove its functionality, even if it was not primarily intended for creating applications for production.**

I. INTRODUCTION

The subject we teach in the bachelor's study program in computer science is called the development of applications for the internet and intranet, and its main goal is to teach students to create web applications using the latest web technologies. The subject has been taught at the Faculty of Informatics and Management for more than 20 years, which is why we have extensive experience with what causes the biggest problems for students. It is taught in the third year of bachelor's study program and is attended by students with different levels of knowledge, from complete beginners to advanced programmers who work in various companies and often on larger web projects alongside school. Of course, we mainly focus on students who need to orient themselves in the given field and learn how web applications are created. The content of the subject is a cross-section of all languages that are used to develop web applications. We start with the basic characteristics of web applications by comparing them with other types of applications (e.g. desktop). Next, we gradually move from simpler languages such as HTML, through languages and frameworks for defining the design of web applications (CSS) to the design of a complex graphical user interface adapted for desktop computers as well as for mobile devices (responsive design). The second part of the subject is the design of the front-end part of the web application using the JavaScript scripting language using the document object model. (DOM). Knowledge of programming is already required in this part of the subject. The third part of the subject is the creation of the server part of web applications. In this section of subject, mainly for historical reasons, we use the PHP language. In the last part of the subject, we focus on issues of security and elimination of possible vulnerabilities that can get into the application in case of faulty implementation.

Since this subject is very practical, the main component of student evaluation is the development of a semester project, when the student is required to create his own web application, which is fully functional and uses all the technologies covered in the subject.

The core of the semester's work is primarily the creation of the server part, because without it no web application can work. In past years, students used the core PHP language to create semester projects, but nowadays web applications are no longer created this way and the vast majority use different frameworks. Of course, we also had to respond to this trend in our subject, and currently almost 95 percent of all term projects are created using some kind of framework.

II. RELATED WORK

In the available literature, the authors mainly deal with either creating a suitable environment for the development of web applications, or possibly proposing what technologies to use for teaching web application development. Some of the ways of learning in various schools in the Czech Republic were discussed by Kučera in [1]. He found that basic technologies like CSS and HTML are not taught very differently. But the problem is the application logic on the client side (JavaScript and WebAssembly) and the server side (any programming language can be used), while modern web applications use frameworks on both sides. The authors in [2] consider the need to prepare a development and circulation environment to be one of the main problems in teaching web technologies, therefore they propose a solution for centralized and collaborative work of students. Their solution focuses on preparing a development environment that will make it easier for students to start developing web applications. Similarly, authors in [3] suggests using their own heterogeneous system of specialized components for web development. Paper [4] investigates whether a framework for developing projects should be

incorporated into the course. Results showed difficulties using a web framework, although this can assist in implementing the OOP design. Conclusions recommend enlarging the practical part of the course, and the use of frameworks. Authors in [5] presents that, the value of teaching web programming is followed by an extensive listing of the underlying challenges. They summarized in a list view, which helps educators of the field to prepare the respective structure, content, methodology and tools of a web programming course that will serve their needs in a productive way. The author in [6] proposes innovation of the curriculum is the content presentation, which focuses on small incremental steps, spanning most term lab sessions, as well as a teaching and a case study which centres on user login management.

III. WEB DEVELOPMENT

A. Web architecture and architectural design pattern MVC

Web application architecture refers to the design and structure of the components and systems that work together to provide a functional and efficient web application. Currently, the architectural design pattern MVC (Model-View-Controller) [7] is most often used in development.

MVC is an architectural pattern used in the development of various software applications that divides the application into three basic components: a *model*, a *view*, and a *controller*. This pattern is also often used in the design of web applications. The main goal of MVC is to separate the different logical parts of an application to make it more maintainable, modular and scalable. This division achieves better code organization, reduces redundancy, and simplifies maintenance after the application is deployed.

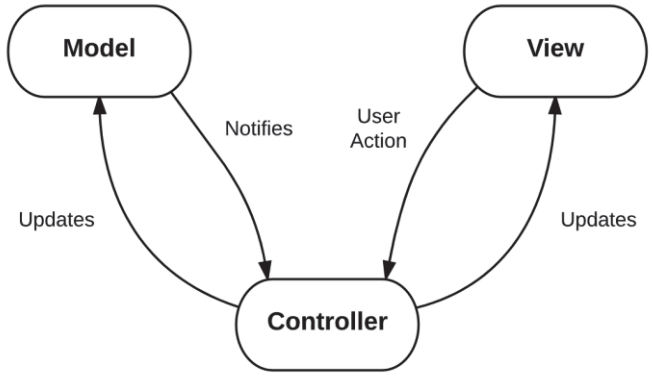

Fig. 1. MVC architectural design pattern

The *model* represents the data layer of the application. This component is responsible for working with data and business logic. The model manages application data, performs operations such as retrieving, storing and updating data in the database. It can also contain data-related rules and validations. If a data change is made in the application, the model is the one to process the change and then update the view to reflect the new data.

The *view* is the component responsible for presenting data to the user. This component displays the user interface and provides a way for users to interact with the application. In the MVC architectural pattern, the view obtains data directly from the model or from the controller and visualizes this data. The view contains minimal business logic, since its main task is to display data in a specific format (HTML, JSON, XML, etc.).

The *controller* is the intermediary between the model and the view. It is the component that handles user requests (such as clicking a button or entering a URL) and decides what operations to perform. The controller communicates with the model to get or change data and then calls the appropriate view to keep the display of data up to date for the user. It basically controls the entire logic of the application to make it functional and interactive.

There are many benefits to using the MVC pattern. Separating the code into three parts makes applications more modular and allows individual components to be developed and tested independently. Developers can easily modify and extend specific parts of the application without disrupting other components. For example, changing the appearance of the application requires modifying only the view, while changes in the data processing logic will only affect the model. This reduces the risk of errors and increases the efficiency of development.

B. PHP frameworks

Currently, there are more than 20 professional frameworks used to develop PHP web applications. It is often very difficult to choose the right framework for creating an application. On many websites there are overviews of frameworks, with a list of their properties and a comparison of individual features of the frameworks[8], [9], [10], [11]. We have selected the 3 most frequently used frameworks by students for our subject and present their basic characteristics.

Three of the most used PHP frameworks are on our subject: Laravel [11], Symfony [12] and CodeIgniter [13]. Laravel is known for its relative simplicity and rich set of tools that simplify development. Symfony is a robust and flexible framework that is often used in large enterprise applications. CodeIgniter, on the other hand, is a lighter and simpler framework, ideal for smaller and less demanding projects.

Laravel has become popular mainly due to its simplicity and user-friendly approach. Its elegant syntax and modern tools, such as Eloquent ORM, Artisan CLI and Blade templating engine, enable rapid application development. Laravel supports complex web applications and offers a large community and a number of packages that developers can take advantage of. It is a framework that supports modern web standards and API development with the help of REST or GraphQL.

On the other hand, *Symfony* is known for its modularity and flexibility. It is ideal for large and complex projects, as it allows developers to select and customize individual components according to the needs of the project. Symfony is also often used in combination with other technologies and frameworks, including Laravel, which inherits some Symfony components. Symfony offers highly scalable solutions and integrates well with other systems, making it the preferred framework for enterprise applications.

CodeIgniter is a framework that is focused on speed and simplicity. It is ideal for smaller projects or applications where

high robustness and complexity are not required. Its low learning curve makes it a good choice for novice developers or those who need simple but functional solutions. Although it doesn't have as much functionality as Laravel or Symfony, CodeIgniter offers good flexibility for projects that don't need extensive structures.

Each of these frameworks has its strengths and optimal use. Laravel is ideal for medium to large projects with a focus on rapid development, Symfony is great for very complex and enterprise solutions, and CodeIgniter is suitable for simple applications or smaller projects where low complexity is required. Choosing the right framework depends on the size and requirements of the project.

As we already mentioned, these frameworks were designed as professional tools for developing web applications and were not intended to be used as teaching tools.

This was a big problem for the students when creating their semester projects, because they were forced to master complicated frameworks in a short time, even if their applications were not complex. Therefore, we decided that it would be good to create our own framework, which, although it will not allow the creation of an application intended for production, but it will be significantly easier to understand and master, and at the same time it will contain similar concepts as advanced frameworks.

C. What causes students the biggest problems

Creating web applications is a very complex process that requires knowledge of many technologies and languages. The existing frameworks are very advanced and relatively complex to master, which causes problems for students, especially beginners. Complex frameworks have a relatively long learning curve and, especially for simpler applications, are quite time-consuming to learn.

Very often, in semester projects, we encountered the fact that students created a monolithic application that did not have separate individual layers, and as a result, students had to change the code in many places of the project in a complicated way when they requested an addition to complete the content. Also, the understanding of what goes where in the web application was often incorrect and the source code of the application was confusing.

If the students chose a professional framework, they often used tutorials available on the Internet when developing a semester project, which they only partially understood, and therefore did not even fully understand how their application works. Furthermore, this superficial knowledge caused them significant problems, especially if we asked them to add or correct something in the application. Although most of the professional frameworks are open source projects and their source codes are available, it was a big problem for students to understand how the framework works from studying these codes.

D. Our framework

This Vaíičko framework (the name is a pun from subject name in Slovak language) is based on the MVC architecture and its goal is to make it easier to understand how the MVC architecture works. It was designed as an auxiliary tool in teaching and in the creation of semestral projects.

The framework is created in the PHP language using the object-oriented programming paradigm. In some parts, it was inspired by the functioning of the Laravel framework. Of course, it is not possible to compare these frameworks, the Vaíičko framework was created for the purpose of teaching and is not intended for the creation of production applications. Its goal is to facilitate the understanding of the principles of creating web applications based on the MVC architecture.

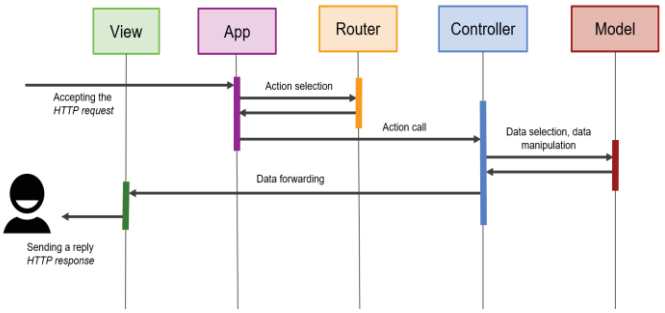

Fig. 2. Framework sequence diagram

The framework uses many conventions that serve to facilitate the work. Some of them can be changed, others are fixed in the source codes. It is usually better to conform to the conventions than to create your own rules and modifications to the existing way the framework works.

Fig. 3. The main page of the framework

The syntax of the framework meets the rules specified in the PSR-12: Extended Coding Style specification [14], which can be an inspiration for how to follow the programming style.

1) Docker support

One of the main obstacles when using any framework is that students have a problem with installation and configuration.

That's why we decided to make the situation easier for them by providing them with a simple docker stack configuration, so installation will be very easy and students will be able to develop right away.

The framework contains the configuration for running in the *Docker* environment. Before that, however, you need to have all software for Docker installed and running (e.g. WSL on Windows). Installation and configuration e.g. Docker desktop can be found many online tutorials. Configuration for Docker contains basic configuration for running and debugging applications. All necessary services are defined in the *docker-compose.yml* file.

2) How does the framework work?

After creating all the containers and entering the URL *http: //localhost/*, the browser will display the initial page of the framework. Even though no address is specified in the URL, the framework displayed the initial page. If no parameters are entered, the following URL will be used by default: *http: //localhost/?c=Home&a=index*.

The steps of processing the request until the response is returned are as follows:

1. When the framework receives a request from the user, it first derives the name of the controller class from the c parameter (*c* as controller) based on it, by adding controller after the name specified in the c parameter. In the above case, we get the *HomeController* class. The fully qualified class name is based on the conventions used in the framework, that means *App\Controllers\HomeController* and this class will be in the *App/Controllers/HomeController.php* file. In this class, it searches for a method according to the content of parameter a (*a* as an action), in the given case the index. A method of a given controller class corresponds to an action.

2. The method in its body executes the defined application logic and selects the view to send in response to the user's request. If you don't specify which view to use, the view found in the *App/Views/* directory in a subdirectory named the same as the controller's name will be used by default. The file is named the same as the name of the action, with the extension *.view.php* added. In this case, the view *App/Views/Home/index.view.php* is used.

3. The required view is integrated into the selected website layout according to the settings in the application configuration (the layouts are **.layout.view.php* files in the *App/Views/* directory) and the response formed in this way is sent to the user as a response to his request.

3) Creating a page using an existing controller

To add your own page to the project, it is possible to create a new action within the existing controller, just create a new method in the controller class, e.g. in the file *App\Controllers\HomeController.php* the method *myPage()*. To display some content as a response to the user (view), it is necessary to create a *myPage.view.php* file in the *App\Views\Home* directory.

To use a custom controller, it must be created first. All controllers must be in the *App\Controllers* directory and must be a child of the *AControllerbase* class. After the controller is created, the necessary actions can be added to it, like in the previous paragraph.

4) Sending data to the view

Not every view is so simple that it doesn't need any data to display properly. If the view needs data, it is the task of the controller to prepare it and send it to the corresponding view. A new method needs to be created in the controller class. We send the data that the view will need to the view via the parameter of the helper method *html()*, which is in charge of finding and calling the corresponding view and selling data to it. In the source file of the view, the data prepared by the controller is available in the *$data* variable, which can be used in the view in the usual way.

5) Creating your own model

If the application is to cooperate with the database, it is necessary to create a model. A model represents an entity that usually has a representation in the database in the form of a table. One application usually contains several models, and the database will also contain several DB tables.

The model class file must be placed in the *App\Models* directory. To ensure that the model can read and write from/to the database, it needs to have the *App\Core\Model* class as its ancestor. A model class must meet the following conventions:

- The model must have a constructor with no parameters (or none),
- Model attributes must be marked protected,
- All attributes must be named the same as the corresponding columns in the DB table.

Other conventions are not mandatory, but they greatly simplify working with the model:

- The model attribute representing the primary key should be named *id*
- In the DB table, the primary key should be marked with the *autoincrement* attribute, if the primary key attribute is called differently, it is necessary to override the *getPkColumnName()* method in the model,
- The name of the class should be in English and in the singular,
- The name of the table in the DB should be in English, plural and lowercase, then the framework is able to automatically link the model and the DB table. If you name the model or table with your own name, it is necessary to override the method returning the name of the table *getTableName()* in the model.

The model should contain all attributes and their *get* and *set* methods. In addition, it may contain other methods whose task is to prepare data for the application logic of the controller. However, it should not contain any application logic.

6) Processing URL GET parameters

The user can also send additional data in the URL address along with the request. These data (also called GET parameters) may contain additional information to the request. In the

framework, they are available in an instance of the *Request* class, which is available in the controller.

To start the *greeting* action, just enter the URL address *http: //localhost/?a=greeting&name=Peter* in the browser. The name of the parameter must match the one found in the URL, in this case name. If there are more parameters, they are separated by the & sign.

7) Creating URLs

It is often necessary to create links in the application, which e.g. they create navigation within the application or redirect the user to another address. The address can be created by combining the name of the controller and the action, or by adding additional URL parameters to be attached to the user's request. However, there is a simpler and more convenient way of creating URLs in the framework.

The *$link* variable is automatically available in every view. The *LinkGenerator* object contains a *url()* method that constructs the corresponding URL. The *$destination* parameter of this method has a simple syntax. For example to build the URL address *http: //localhost/?c=home&a=index*, it is enough to send the string *"home.index"* as a parameter, where the part before the dot corresponds to the name of the controller and the part after the dot corresponds to the name of the action.

If it is necessary to create URLs in the controller, the same *url()* method of the *LinkGenerator* class can be used. To facilitate the work of the developer, the helper method *url()* is available directly in the controller.

8) Authentication

The framework provides a basic implementation of authentication, there is even a view with a login form and a view that is displayed after login. The student can customize the appearance and method of identity verification according to his needs.

There are two views in the *App\Views\Auth* directory: *login.view.php* and *logout.view.php*. Since we don't need to have a menu when logging in, the views use a different page layout, which can be seen by setting *$layout = 'auth';*. The appearance of the form can be adjusted according to your needs. If you want to use your own action for logging in, you can set its URL in the configuration (constant *Config\Configuration::LOGIN_URL*).

There is an *AUTH_CLASS* constant in the *Config\Configuration* application configuration class that specifies which class to use for authentication. Sample simple authentication is implemented in the *App\Auth\DummyAuthenticator* class. This class implements the *App\Core\IAuthenticator* interface. If it is necessary to implement a custom authentication method, e.g. user authentication in the database, it is necessary to implement a custom authenticator class according to the *App\Core\IAuthenticator* interface. You can implement all the methods in your own way according to your needs. Finally, it is necessary to set the new class in the configuration as the one that will be used in the application.

9) Authorization

For some applications, it is necessary to solve the authorization in a more complicated way. Just knowing whether the user is logged in or not is not enough. For example, if individual users have defined roles in the application, you may want the admin user to have different actions available than the normal user. The framework uses the *authorize()* method for this purpose.

In this method, it is possible to implement the authorization logic for all actions of the respective controller. In the entry of the application, e.g. the condition that the post (*Post* model) can be edited and deleted only by the user who created it, but all logged-in users can add a post.

10) Summary

This is a brief description of the core of the framework. This description is not exhaustive, the framework contains several other parts, but they only make it easier for students to develop certain routine parts of the application. The Framework is currently only used at our faculty. The source codes of the framework are freely available in the public *github* repository [15], so students can easily familiarize themselves with the implementation details of the framework and it is already possible to download it and use it for your own purposes. On the *github* server there is also documentation for the framework together with examples [16] that students can use to build a simple application. We also use the framework in several labs and use it to create sample applications so that students learn how to work with the framework.

IV. EVALUATION

We introduced the Framework into the subject in the 2022/23 school year and included lectures and exercises in the syllabus of the subject. After initial mistrust, we noticed an increased number of students who used our framework to develop a semestral project. The data can be seen in the following table:

TABLE I. NUMBER OF STUDENTS USING OUR FRAMEWORK

School year	Number of students
2021/22	5
2022/23	20
2023/24	31

Students can work out their semester work using a language other than PHP. We do not want to limit if they already know some other technology. However, some of them will use the PHP language, so the next table shows how many students use the described framework and how many some other PHP framework.

TABLE II. NUMBER OF STUDENTS USING FRAMEWORK

School year	Our framework	Another PHP framework	Another framework (except PHP)	Sum
2021/22	4.07%	56.91%	39.02%	100%
2022/23	15.15%	40.91%	43.94%	100%
2023/24	25.41%	28.69%	45.90%	100%

The given data shows that the number of students who prepare their semester work using the mentioned framework is growing. However, there is still a significant group that prefers another framework (PHP or another programming language).

At the end of this school year, we therefore asked the students what the reasons were for choosing the framework they will use to develop their semester work. Those who chose our framework cited the following reasons most often:

- steep learning curve
- small scope of source codes, easy study of how the framework works
- plenty of examples in the exercises
- simple and clear wiki pages with documentation
- supporting teachers in solving problems.

Those who chose another framework as their reasons for not choosing our framework stated

- I already had knowledge of another framework,
- The use of the framework is limited to this subject only, I will not use it elsewhere
- I was making a more complex application, the framework was too simple for me,
- Lack of tutorials for creating applications in this framework.

For a better evaluation of the pedagogical impact, it would be good to create a more targeted questionnaire that would give answers to questions in which parts of web development the framework helped them the most, or what remained incomprehensible even with such a simpler framework. In the following years, we therefore do not rule out that we will prepare a more extensive questionnaire focused precisely on pedagogical impacts.

V. Conclusion

We have been using the described framework for 3 years. It is currently in the current version 2.0, where we have fixed several bugs and added a few new features. We are currently working on version 3.0, which again brings several small improvements and mainly expands the options for creating models to make working with the framework even more convenient for students. Despite the fact that the use of the framework is very simple and we would like to leave it in a state so that it is not complicated to use even for beginners, we are preparing another version in which we will add some advanced functions, which, however, will not be necessary to use, but will enable the creation of more complex applications for help with more complex concepts. Although the further development of the framework is practically unlimited thanks to the modularity and object-oriented approach, we carefully consider each new feature. Although the new functionalities make the framework more mature, on the other hand, too complicated concepts can discourage students, and the main task of the framework is to remain a simpler alternative to professional frameworks. From the data presented in the previous chapter, it follows that the use of the framework is gradually growing, and it has found its supporters in the councils of students who use it to prepare their semester work. The creation of several sample applications would certainly help to expand the framework more, where all procedures that students need for semester work would be demonstrated. The source codes of the framework themselves are created in English, as are the comments, but there is no English documentation yet, so we cannot use the framework for teaching Erasmus students yet. We believe that we will be able to solve this shortcoming soon and the framework will serve an increasing number of students.

References

[1] V. Kucera, “Comparative Analysis of Teaching Methods of Making Web Pages,” *Procedia Soc Behav Sci*, vol. 171, pp. 945–949, Jan. 2015, doi: 10.1016/j.sbspro.2015.01.213.

[2] A. Jevremovic, N. Ristic, M. Veinovic, G. Shimic, and N. Stanisic, “WIDE: Centralized and collaborative approach to teaching web development,” *Journal of Internet Technology*, vol. 19, no. 4, pp. 1003–1014, 2018, doi: 10.3966/160792642018081904004.

[3] J. O. Liegle and P. Meso, “Evaluation of a Virtual Lab Environment for Teaching Web-Application Development,” 2007. [Online]. Available: http://isedj.org/5/7/

[4] O. Barzilai and R. Gafni, “Using Web Frameworks in Server Side Programming Courses,” *Journal of Computer Information Systems*, vol. 63, no. 4, pp. 866–876, 2023, doi: 10.1080/08874417.2022.2111378.

[5] S. Xinogalos and T. H. Kaskalis, “The challenges of teaching web programming: Literature review and proposed guidelines,” in *WEBIST 2012 - Proceedings of the 8th International Conference on Web Information Systems and Technologies*, SciTePress, 2012, pp. 207–212. doi: 10.5220/0003960902070212.

[6] F. Maiorana, “Teaching Web Programming An Approach Rooted in Database Principles.” [Online]. Available: http://www.codeavengers.com/

[7] “Model-View-Controller Pattern,” in *Learn Objective-C for Java Developers*, Berkeley, CA: Apress, 2009, pp. 353–402. doi: 10.1007/978-1-4302-2370-2_20.

[8] M. Laaziri, K. Benmoussa, S. Khoulji, and M. L. Kerkeb, “A Comparative study of PHP frameworks performance,” in *Procedia Manufacturing*, Elsevier B.V., 2019, pp. 864–871. doi: 10.1016/j.promfg.2019.02.295.

[9] Jeel Patel, “PHP Framework Comparison – Which is The Best PHP Framework?” Accessed: Sep. 13, 2024. [Online]. Available: https://www.monocubed.com/blog/php-framework-comparison/

[10] A. Niarman, Iswandi, and A. K. Candri, “Comparative Analysis of PHP Frameworks for Development of Academic Information System Using Load and Stress Testing,” *International Journal Software Engineering and Computer Science (IJSECS)*, vol. 3, no. 3, pp. 424–436, Dec. 2023, doi: 10.35870/ijsecs.v3i3.1850.

[11] “Laravel framework documentation.” Accessed: Sep. 13, 2024. [Online]. Available: https://laravel.com/docs/master

[12] “Symfony Documentation,” https://symfony.com/doc/current/index.html. Accessed: Sep. 13, 2024. [Online]. Available: https://symfony.com/doc/current/index.html

[13] “CodeIgniter framework docs,” https://codeigniter.com/userguide3/#. Accessed: Sep. 13, 2024. [Online]. Available: https://codeigniter.com/userguide3/#

[14] “PSR-12: Extended Coding Style specification,” https://www.php-fig.org/psr/psr-12/.

[15] “Github repository of Vaiicko framework,” https://github.com/fri-portal/fri-uniza-laravel.

[16] “Vaiicko framework documentation,” https://github.com/thevajko/vaiicko/wiki. Accessed: Sep. 13, 2024. [Online]. Available: https://github.com/thevajko/vaiicko/wiki