Received Nopember 25th, 2024; accepted December 26th, 2024. Date of publication December 30th, 2024
Digital Object Identifier: https://doi.org/10/25047/jtit.v11i2.5673

Design MicroServer Framework Library with Swoole for Real-time Application Development

MOHAMMAD ROBIHUL MUFID[1], YOGI PRATAMA[2], ARNA FARIZA[3], SANIYATUL MAWADDAH[4], YUNIA IKAWATI[5], DARMAWAN ADITAMA[6], MUHLIS TAHIR[7]

[1,2,3,4,5,6] Politeknik Elektronika Negeri Surabaya, Jl. Raya ITS, Sukolilo, Surabaya, Indonesia
7Universitas Trunojoyo Madura, Jl. Raya Telang, Bangkalan, Madura, Indonesia

CORESPONDING AUTHOR: MOHAMMAD ROBIHUL MUFID (email:mufid@pens.ac.id)

ABSTRACT The PHP programming language is known as a synchronous programming language because the request execution model is carried out sequentially and is very easy to apply in creating systems with simple scenarios. With this model, there will be many challenges in developing real-time applications because sequential model execution can cause bottlenecks because it initializes threads on each request which causes more resource consumption, making it less suitable for handling I/O (Input/Output) operations on an Intense scale. This study aims to implement an asynchronous model in PHP by developing a Xel Async framework that can be used as a foundation for creating micro servers using the Swoole extension. This study will explain the framework developed starting from system modeling, component design, abstraction design that focuses on response time, throughput, and efficient and effective resource usage in handling heavy traffic. To see the performance of the framework developed, an analysis was carried out with other frameworks such as Express Js. And the results show that the Xel Async framework offers significant performance carried out on benchmark tests for 100, 250, 500, 1000 connections and is able to produce a better amount of latency. In addition, an analysis was also carried out for throughput on Xel Async and Express Js, which also produced better performance than Express Js.

KEYWORDS: php, swoole, asynchronous, xel-async, micro server.

I. INTRODUCTION

Hypertext Preprocessor (PHP) is one of the 5 most well known server-side programming dialects. Where PHP is additionally utilized by nearly 75.9% of all existing websites. Figure 1 appears the drift of PHP programming dialect compared to Java, JavaScript, ASP.net, and Ruby programming dialects [1]. Where the chart appears that the assist to the proper, the dialect is used by numerous websites. PHP appears the position on the proper which appears that it is broadly utilized by different websites, but is for the most part utilized for websites with normal activity. In the interim, websites with tall activity for the most part utilize the JavaScript programming dialect [2], [3], [4].

PHP is known as a Synchronous programming language that executes requests sequentially. Although easy to apply in creating a system with a simple scenario, this model can face Bottleneck problems when faced with traffic that does many I/O (Input/Output) operations. This is caused by thread initialization on each PHP request which results in large resource usage rather than reusing threads to handle several processes [5], [6]. Therefore, PHP is less suitable to build real-time applications. One solution to overcome this problem is to utilize a PHP-based library called swoole. Swoole is a PHP-based library that can support asynchronous and concurrent programming capabilities. By utilizing this library, you can build websites using PHP faster and you can also build applications in real time.

There are several studies that try to compare the performance between server programming languages. One of them is a study that discusses the performance between PHP and Node Js [12]. Where one of the testing methods is to use a binary algorithm, a quick sort algorithm, a bubble algorithm, and a heap algorithm that is considered suitable because it has increased time complexity. From the results of testing these two methods, it was found that the PHP programming language has better performance than Node Js in terms of latency and other metrics [13], [14], [15], [16].

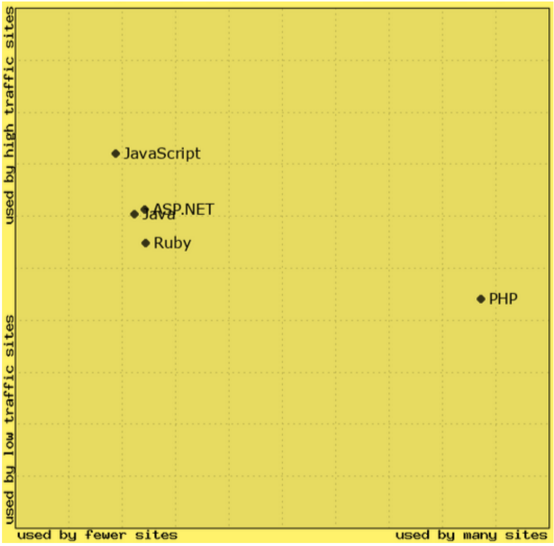

FIGURE 1. Market position for server language programming

In addition, there is also research that builds real-time applications with other frameworks such as research from Vineet K.S. [17] who built a real-time application for streaming and broadcasting built with the Express Js framework, Node Js, React Js, and Mongodb database. The technology applied is by integrating WebRTC and RTMP in the streaming application.

In this paper, the author aims to develop a framework for developing real-time applications based on PHP called Xel Async. The concept of Xel Async is a Micro server Framework library with the Swoole extension that can be used to develop real-time applications in PHP without requiring a traditional web server [18], [19]. Xel Async includes the Xel-Psr7-Bridge library, which provides a bridge between the Xel Async framework and the PSR-7 HTTP message interface standard [20], [21]. By using Xel Async, it is hoped that it can overcome performance problems in PHP and provide better solutions in building real-time applications. The performance benefits of using an asynchronous approach with Swoole have been demonstrated in various studies. Furthermore, researchers have explored architectural patterns and modifications to the traditional Model-View-Controller (MVC) design pattern to better leverage the capabilities of asynchronous programming in PHP. Where this pattern is to divide the framework structure into 3 parts, namely the logic part in the controller, the data structure part in the model, and the interface part in the view, all of which are mutually continuous [22], [23], [24].

Asynchronous programming in PHP has become increasingly popular due to its ability to handle concurrent requests efficiently, which is essential for building high-performance web applications and real-time systems. The Swoole extension, which is at the core of Xel Async, provides an event-driven, asynchronous programming framework for PHP that enables non-blocking I/O operations, coroutines, and efficient concurrency management. By leveraging Swoole's capabilities, Xel Async aims to overcome the limitations of traditional synchronous PHP applications and enable the development of scalable, high-performance real-time applications.

II. METHOD

Xel-Async is a library built on the Swoole extension which provides minimal utilities for building projects flexibly with the focus, providing an asynchronous server development ecosystem in PHP to support the needs of developing real-time applications with high performance along with a small footprint.

Figure 2 shows system design of Xel-Async with execution and integration model between components in the library which starts with several point below.

1. Definition of Bootstrap

Figure 3 shows bootstrapping model of xel and have a function to access the beginning of the system to define component configurations such as servers, container dependency injection, define dynamic routing and middleware, define services from business logic and other components. This bootstrap definition will be injected into the Application class as the main gateway in system configuration.

Figure 4 shows about how the flow of Bootstrap work and contain several processes as follows:

- Instantiate the server class with configuration injection which will be processed serially using another method.
- Next, to handle requests and responses, this is done through the on-Event method which accepts config parameters and container objects.
- Instantiation of Psr-factory to build a bridge adapter that functions as a link between Swoole http server requests and the PSR 7 standard.
- Instantiate the router by injecting instances of the Psr-factory and Container objects.
- Create a server set at the on Request event and connect the Swoole http request to Psr-bridge, map the router by injecting the class config loader as well as the request method and URI that are captured when there is a request. Next, execute the router by injecting the Swoole http server request and response.
- Run the Applications class to start the server program from the Xel-Async library.

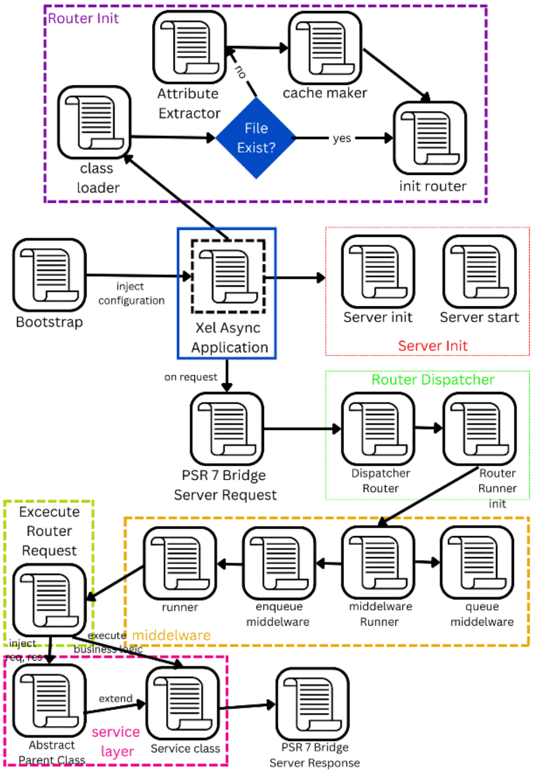

FIGURE 2. System Design of Framework Xel.

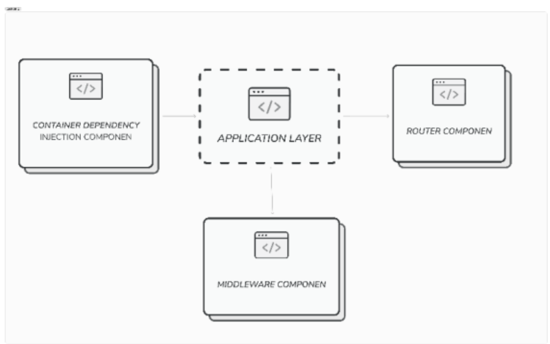

FIGURE 3. Bootstrapping model of Xel.

```
Class Applications
    Define instance of Servers

    Function initialize with config as parameter
        Create new instance of Servers with config
        Return self

    Function onEvent with loader and register as parameters
        Create new instance of PsrFactory with register
        Create new instance of Main with register and PsrFactory

        On 'request' event of instance
            Connect Swoole Http Request to PsrBridge
            Map router with loader, method and path of request
            Execute router with request and response

        Return self

    Function run
        Launch instance
```

FIGURE 4. Flow of Bootstrap work.

2. Server Initialization

Figure 5 shows server setup including 2 modes. In this process, it is used to configure the server according to needs. The xel library focuses on building an http/https server that supports upgrades up to http2. Then, this initialization have their characteristic modes, like Base Mode and Process Mode.

- Base Mode: is a non-blocking asynchronous server mode which has a model that is almost the same as NGINX and Node.js execution because there are no process roles in this mode and each worker initialization will compete for a connection which will then make a TCP connection directly with the client who asked. After that, data will be received and sent via this connection line to the connected worker. Forwarded through the reactor from the main thread.
- Process Mode is a mode that utilizes inter-process, and all TCP client connections are established in the Main process. Because there are many inter-processes for communication, Main Process provides process management and memory protection mechanisms which aim to support complex processes and business logic and ensure server stability when running for a long time.

```php
<?php
return [
    'api_server' => [
        'host' => 'http://localhost',
        'port' => 9501,
        'mode' => 1,
        'options' => [
            "enable_static_handler" => true,
            "document_root" => dirname(__DIR__, 2),
            'worker_num' => swoole_cpu_num(),
//            'task_worker_num'=>16,
            'task_enable_coroutine' => true,
//            'daemonize' => 1,
//            'http_gzip_level' => 9,

```

FIGURE 5. Server mode setup.

Table 1 shows additional information related to the settings on the server initialization side. Where the server side settings will be set starting from the server IP address settings, the port used, server mode, number of workers, coroutines, http compression, and dispatch mode.

TABLE 1. Server Setup Options

Setup	Descripts
0.0.0.0 (default)	IP Address Server
9501 (default)	Default port
Mode 1/2	Server mode for option 1 is single mode and 2 is process mode.
worker_num	The option to add workers for base mode is recommended to setup based on CPU cores, for process

Setup	Descripts
	mode set up a minimum of 35 workers adjusted to capacity.
enable coroutine	To activate coroutines during server initialization
http_compression	For http request size compression
dispatch_mode	Used for the connection distribution algorithm and can only be used in mode 2

3. Dynamic Routing Initialization

Figure 6 shows about dynamic routing flow based on attributes. Initialization routing is carried out dynamically by utilizing several main tools provided by PHP such as reflection API and attribute.

- Reflection API: used to interact with classes that have been registered as services, with the aim of getting the routing value that has been injected into the attribute.
- ATTRIBUTE (Annotation): has a similar function but has been improved with a new, more consistent concept. This attribute is very useful in accommodating routing parameters.

In Figure 6, there are 5 process stages for generating router parameters, including:

- Service class injection functions as a facilitator that holds attributes for router parameter generation. This includes a list of service/controller classes used in this process.
- Checking the availability of files containing generated router parameters which function as static cache handlers.
- The process of extracting router parameters from each service class that is injected via attributes that include the middleware group, middleware per method class.
- The process of bundling router parameters includes URIs, request methods, classes and methods, as well as middleware. Middleware is classified into three layers, namely global middleware, group middleware, and single middleware.
- Then initialize the router based on the results of the parameters generated.

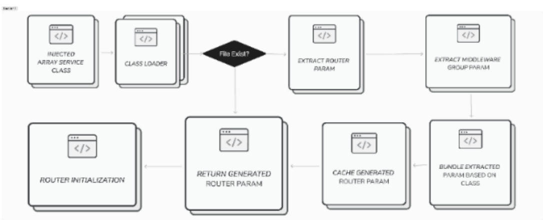

FIGURE 6. Routing dynamic model.

Using reflection API and use attribute for declaring routing parameters, middleware injection will keep development simpler and easier to integrate and debug since the routing will bound with class and method. Detail sample for dynamic routing on figure 7.

```
/**
 * @throws DependencyException
 * @throws NotFoundException
 */
#[GET("/protected", [AuthMiddleware::class])]
public function protectedSource(): void
{
    $this->return
        ->workSpace(function (Responses $responses){
            $responses->json('valid user', false, 200);
    });
}
```

FIGURE 7. Sample attribute routing for get request with middleware.

4. Bridging Swoole Standard Http Requests with Psr17

The bridging request process involves the mapping process of the Swoole http request using abstractions from PSR17 which will be converted to the PSR7 standard. This mapping involves a duplication process regarding header parameters, cookies, query parameters, parsed body, body (raw content), and upload files. Using the bridge method can simplify request processing, because it will only copy data from the Swoole Http Server Request and Response only when used rather than copying and converting directly to the PSR7 standard.

5. Bundling All Components to Skeleton

Based on the previous design components which include server (http server), dynamic router, PSR7 Bridger, Bootstrap will be arranged and integrated in the form of a skeleton to run all components in a distributed manner and submit the bundle as project to packages to make it installable using composer command provided at figure 8.

FIGURE 8. Installable skeleton project.

III. RESULT AND DISCUSSION

In project testing, experiments are carried out with several different test cases. The test is to test the HTTP request method GET and POST with 100, 250, 500, 1000 concurrent requests. To evaluate the performance of the proposed library, we compare it with a popular Node.js framework Express JS using a wrk benchmark tool. The benchmark tests were conducted on a server with following specifications :

- CPU : Ryzen 3 3250u 2.6GHz
- Ram : 8 GB
- Operating System : FEDORA 38

Other tools for benchmarking as follows :

- Wrk benchmarking tools,
- Node JS version 18.
- Express JS
- PHP 8.2 (installed swoole-ext)

Benchmarking test case is Hello World benchmark for GET request and for POST request is to display input form client input.

1. Node JS setup

Logic code server have two request handlers: handleGetRequest and handlePostRequest function and run in port 3000 based on figure 9.

```
const express = require('express');
const app = express();
const PORT = 3000;

// GET request handler
app.get('/', (req, res) => {
  res.send('Hello World');
});

// POST request handler
app.post('/', (req, res) => {
  let body = '';

  req.on('data', (chunk) => {
    body += chunk.toString();
  });

  req.on('end', () => {
    console.log('Received POST data:', body);
    res.send('Hello World');
  });
});

// Start the server
app.listen(PORT, () => {
  console.log(`Server running at
http://localhost:${PORT}/`);
});
```

FIGURE 9. Node JS Setup.

2. Xel server setup

- Install skeleton project using command "composer create-project xel/skeleton".
- Setup Server Using mode 1 with 1 worker to make fair test with Node for non-cluster.
- Figure 10 show Code logic. Based on this setup logic all setup for each component remain default and this server will run on port 9501.

```
    #[GET("/")]
    public function index(): ResponseInterface
    {
//          $this->serverRequest->getMethod();
        return $this->serverResponse->plain("Hello Xel", 200);
    }

    #[POST("/post")]
    public function post(): ResponseInterface
    {
        $data = $this->serverRequest->getBody()->getContents();
        $decode = json_decode($data);
        // var_dump($decode);
        return $this->serverResponse->json([$decode], 201);
    }

    #[POST("/post-text")]
    public function postText(): ResponseInterface
    {
        $data = $this->serverRequest->getBody()->getContents();
        // var_dump($data);
        return $this->serverResponse->plain($data, 201);
    }
```

FIGURE 10. Xel Framework Setup.

3. Benchmarking Result for Xel

In table 2 regarding the Xel test results, an HTTP request was made using the GET and POST method for 60 seconds. The purpose of this test is to test consistency in sending data in the form of JSON "hello world". The test results show that with 100, 250, 500 and 1000 connections, there is an increase in latency, but the handling of data transmission is still relatively good considering the number of connections is different for each test. Apart from that, requests per second also remained stable after being tested with four different scenarios.

TABLE 2. Xel Benchmarking Result

User	Content - type	Latency (90%)	Request per sec	Method
100	application/json	7,48ms	26789.88	GET
250	application/json	13,44ms	28.994,82	GET
500	application/json	24,21ms	30.092,94	GET
1000	application/json	44,91ms	29.999,52	GET
100	application/json	7,4 ms	21.833,44	POST
250	application/json	16,96ms	22.168,76	POST
500	application/json	29,41ms	23.740,84	POST
1000	application/json	62,57 ms	23.308,74	POST

4. Benchmarking Result for Express JS

In table 3 regarding Express JS test results, an HTTP request was made using the GET POST method. The purpose of this test is to test consistency in sending data in the form of "hello world" JSON. The results show that with several connections of 100, 250, 500, and 1000, there is quite a large increase in latency, but the handling of data transmission is still relatively good considering the large number of connections. Additionally, requests per second remain well below Xel.

TABLE 3. Express JS Benchmarking Result

User	Content - type	Latency (90%)	Request per sec	Method
100	application/json	18.57ms	5169.55	GET
250	application/json	49.17ms	5038.29	GET
500	application/json	98.37ms	5025.175	GET
1000	application/json	199.83ms	4990.235	GET
100	application/json	29,44 ms	3260.12	POST
250	application/json	78.09ms	3255.61	POST
500	application/json	156ms	3172.79	POST
1000	application/json	320.29	3109.97	POST

.

In this section, the performance of the proposed Xel framework in this study will be discussed compared to previous studies that used the Express JS framework to create real-time applications. Where the topic to be compared is the latency of each framework. Latency is the total amount of time needed to send data. Where the latency data to be measured is starting from sending data using the GET method and sending data using the POST method. The GET and POST methods are methods in sending data that have differences in terms of data security. Where the POST method has higher data security compared to the GET method.

Figure 12 is a graph of the comparison of latency for sending data using the GET method between the Xel framework and Express JS. Where the graph shows that the latency in the Xel framework has a lower latency than the latency in Express JS. This lower latency indicates that the data sending process in the Xel framework is faster than Express JS. While Figure 13 shows a graph related to the comparison of latency between the Xel framework and Express JS using the POST method. Where the results also show that the latency in the Xel framework has a lower value than Express JS. So it can be concluded that the Xel framework has a faster data delivery time compared to Express JS.

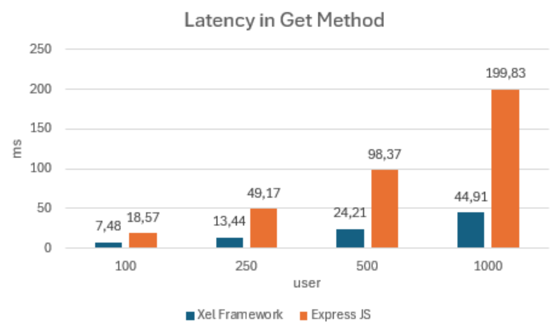

FIGURE 11. Latency in get method.

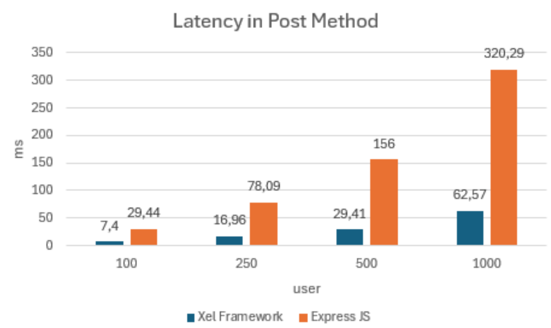

FIGURE 12. Latency in post method.

IV. CONCLUSION

PHP is a synchronous server-side programming language. Where one of the problems for synchronous programming languages when applied to real-time applications is that bottlenecks often occur because they always initialize threads every time a request is made. This study aims to propose a xel framework that implements the asynchronous PHP model using the Swoole extension. Where in this Xel framework, updates are proposed to the PHP framework modeling system, system components, and resource usage efficiency. The test results of the Xel framework show that the Xel framework is suitable for implementation in real-time applications, where the test results compared to other frameworks such as Express JS have better performance in terms of latency and throughput. Then for further research, it is hoped that the development of this Xel framework will focus more on developing data security.

ACKNOWLEDGMENT

The researcher would like to express his thanks to the institution Politeknik Elektronika Negeri Surabaya has become a sponsor from this local research through research centers and community service at the Politeknik Elektronika Negeri Surabaya.

REFERENCE

[1] "Market Position Report of Top 10 Server-side Programming Languages." Accessed: Sep. 06, 2024. [Online]. Available: https://w3techs.com/technologies/market/programming_language/10

[2] F. Sulaeha and I. Yunita, "Design and Build Online Sales on Website-Based Butiq Bilqis Store Using PHP MySQL," *International Journal of Health, Economics, and Social Sciences (IJHESS)*, vol. 6, no. 3, pp. 878–883, 2024, doi: 10.56338/ijhess.v6i3.5828.

[3] S. Sotnik, V. Manakov, and V. Lyashenko, "Overview: PHP and MySQL Features for Creating Modern Web Projects," *International Journal of Academic Information Systems Research*, vol. 7, no. 1, pp. 11–17, Jan. 2023, [Online]. Available: www.ijeais.org/ijaisr

[4] H. Nurrahmi, "Development of a Web-Based Digital Archive Search System Using The Binary Search Algorithm," *International Journal of Engineering Business and Social Science*, vol. 3, no. 1, pp. 14–20, Aug. 2024, doi: 10.58451/IJEBSS.V3I1.188.

[5] H. Ha and S. Yoo, "Comparing Synchronous and Asynchronous Online Programming Classes: Similarities and Differences," *International Journal of Information and Education Technology*, vol. 14, no. 2, pp. 293–301, 2024, doi: 10.18178/ijiet.2024.14.2.2051.

[6] B. Petit and M. Serrano, "Interactive Music and Synchronous Reactive Programming," *The Art, Science, and Engineering of Programming*, vol. 5, no. 1, p. 2020, Jun. 2020, doi: 10.22152/programming-journal.org/2021/5/2.

[7] W. M. C. J. T. Kithulwatta, K. P. N. Jayasena, B. T. G. S. Kumara, and R. M. K. T. Rathnayaka, "Performance Evaluation of Docker-based Apache and Nginx Web Server," *2022 3rd International Conference for Emerging Technology, INCET 2022*, 2022, doi: 10.1109/INCET54531.2022.9824303.

[8] A. Tedyyana and O. Ghazali, "Teler Real-time HTTP Intrusion Detection at Website with Nginx Web Server," *JOIV : International Journal on Informatics Visualization*, vol. 5, no. 3, pp. 327–332, Sep. 2021, doi: 10.30630/JOIV.5.3.510.

[9] I. Putu, A. Eka Pratama, I. Made, and S. Raharja, "Node.js Performance Benchmarking and Analysis at Virtualbox, Docker, and Podman Environment Using Node-Bench Method," *JOIV : International Journal on Informatics Visualization*, vol. 7, no. 4, pp. 2240–2246, Dec. 2023, doi: 10.62527/JOIV.7.4.1762.

[10] L. Mangal *et al.*, "22 International Journal for Modern Trends in Science and Technology As per UGC guidelines an electronic bar code is provided to seure your paper," *International Journal for Modern Trends in Science and Technology*, vol. 8, no. 01, pp. 22–25, 2022, doi: 10.46501/IJMTST0801004.

[11] M. R. Mufid, N. R. Kusuma Saginta Putri, A. Fariza, and Mu'Arifin, "Fuzzy logic and exponential smoothing for mapping implementation of dengue haemorrhagic fever in Surabaya," in *International Electronics Symposium on Knowledge Creation and Intelligent Computing, IES-KCIC 2018 - Proceedings*, Institute of Electrical and Electronics Engineers Inc., Jul. 2018, pp. 372–377. doi: 10.1109/KCIC.2018.8628533.

[12] Q. Odeniran, H. Wimmer, and C. M. Rebman, "Node.js or PhP? Determining the better website server backend scripting language," *Issues in Information Systems*, vol. 24, no. 1, p. 328, Jan. 2023, doi: 10.48009/1_IIS_2023_128.

[13] M. Fariz, S. Lazuardy, and D. Anggraini, "Modern Front End Web Architectures with React.Js and Next.Js," *International Research Journal of Advanced Engineering and Science*, vol. 7, no. 1, pp. 132–141, 2022.

[14] H. Y. Kim *et al.*, "DAPP: automatic detection and analysis of prototype pollution vulnerability in Node.js modules," *Int J Inf Secur*, vol. 21, no. 1, pp. 1–23, Feb. 2022, doi: 10.1007/S10207-020-00537-0/METRICS.

[15] M. F. Falah *et al.*, "Comparison of cloud computing providers for development of big data and internet of things application," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 22, no. 3, pp. 1723–1730, Jun. 2021, doi: 10.11591/ijeecs.v22.i3.pp1723-1730.

[16] J. S. Pasaribu, "Development of a Web Based Inventory Information System," *International Journal of Engineering, Science and Information Technology*, vol. 1, no. 2, pp. 24–31, Mar. 2021, doi: 10.52088/IJESTY.V1I2.51.

[17] V. K. Singh, P. Awasthi, L. Tripathi, S. Thakur, & P. Singh, "Mern (Mongodb, Express-Js, React-Js, Node-Js) Stack Web-Based Streaming and Broadcasting Application," In *2024 International Conference on Electrical Electronics and Computing Technologies (ICEECT)*, Vol. 1, pp. 1-6, August 2024.

[18] Y. Sun, C. Wang, G. Fan, Q. Shi, and X. Zhang, "Fast and Precise Static Null Exception Analysis with Synergistic Preprocessing," *IEEE Transactions on Software Engineering*, pp. 1–14, 2024, doi: 10.1109/TSE.2024.3466551.

[19] D. Tsindeliani, Y. Povstyana, N. Lishchyna, and A. Yashchuk, "Latency Reduction in Real-time GPS tracking in Android and the Web-based GPS Monitoring System," *Proceedings of the 2022 IEEE 12th International Conference on Dependable Systems, Services and Technologies, DESSERT 2022*, 2022, doi: 10.1109/DESSERT58054.2022.10018609.

[20] J. A. N. Hasim, U. Sa'adah, D. I. P. Sari, F. A. Damastuti, and F. N. Koirudin, "Developing Microframework based on Singleton and Abstract Factory Design Pattern," *IES 2022 - 2022 International Electronics Symposium: Energy Development for Climate Change Solution and Clean Energy Transition, Proceeding*, pp. 676–683, 2022, doi: 10.1109/IES55876.2022.9888548.

[21] D. S. Jayalakshmi, D. Hemanand, G. Muthu Kumar, and M. Madhu Rani, "An efficient route failure detection mechanism with energy efficient routing (Eer) protocol in manet," *International Journal of Computer Network and Information Security*, vol. 13, no. 2, pp. 16–28, Apr. 2021, doi: 10.5815/IJCNIS.2021.02.02.

[22] G. Paolone, R. Paesani, M. Marinelli, and P. Di Felice, "Empirical Assessment of the Quality of MVC Web Applications Returned by xGenerator," *Computers 2021, Vol. 10, Page 20*, vol. 10, no. 2, p. 20, Feb. 2021, doi: 10.3390/COMPUTERS10020020.

[23] R. Tarigan *et al.*, "Implementation of MVC (Model-View-Controller) Concept in Develop Management Information System for Transportation Fees (Uang Jalan Operasional) and Revenue in Trucking Companies," 2020.

[24] S. Nahhas, "MVC Architecture from Maintenance Quality Attributes Perspective," *Safia Nahhas International Journal of Computer Science and Security (IJCSS)*, no. 15, p. 169.

Mohammad Robihul Mufid is an academic and lecturer at the Surabaya State Electronics Polytechnic (PENS) specializing in control technology, signal processing, and machine learning. I completed my formal education in informatics and computer engineering, focusing on control systems and intelligent technology. I focus on the development of science, especially in the integration of artificial intelligence into control and automation systems. Outside of academics, I am active in various research and technology development projects that support industrial progress, such as automation and artificial intelligence applications in the manufacturing sector.

Yogi Pratama, is an academic affiliated with the Surabaya State Electronics Polytechnic (PENS), one of the best vocational education institutions in Indonesia. He is actively involved in teaching, research, and technology development, especially in the fields of informatics and electronics. PENS is known as a center of excellence in technology, including informatics engineering, electrical engineering, and multimedia technology, where its lecturers contribute significantly to various innovations and national-level research projects..

Arna Fariza is a lecturer and researcher from Gresik who successfully earned a doctorate from the Sepuluh Nopember Institute of Technology (ITS). Her research focuses on the development of an automatic age estimation system based on panoramic radiography for forensic odontology. This system utilizes dental images to identify a person's age with an accuracy of up to 60.93%. The research aims to help the identification process in situations such as mass accidents or natural disasters, as well as contribute to the field of forensic medicine and rescue teams..

Saniyatul Mawaddah, is a lecturer and researcher in the field of Informatics Engineering at the Surabaya State Electronic Polytechnic (PENS), especially at the Lamongan Regency Campus. She has made significant contributions to technology-based research, one of which is on the classification of rhizome images using the Support Vector Machine (SVM) method, which was published in an international scientific conference in 2022..

Yunia Ikawati is a lecturer and researcher in the field of Informatics Engineering at the Surabaya State Electronic Polytechnic (PENS), Lamongan Regency Campus. Her research and teaching cover the fields of data analysis, web development, Internet of Things (IoT), and wireless networks. In her academic activities, she has published various scientific works that contribute to the development of technology, such as research on student learning styles using the Moodle learning management system.

Darmawan Aditama, S.Kom., M.T., is a lecturer at the Surabaya State Electronic Polytechnic (PENS) who focuses on Multimedia Broadcasting Technology. He is also active in teaching web design at various educational institutions, such as

the Muhammadiyah University of Gresik (UMG). There, he teaches programs such as the Computer Diploma with a specialization in web-based application development and digital technology..

Muhlis Tahir is an academic and practitioner in the field of information technology focusing on computer networks, artificial intelligence, and network security. He is currently affiliated with Trunojoyo University, Madura, where he actively teaches and conducts research. He has an academic background in Informatics Education (S1) and holds a Master of Applied Computer (M.Tr.Kom). Some of his main areas of expertise include Artificial Intelligence, Medical Data Mining, Game Education, and Network Security.