Frontend Framework Consideration for IT Developer

Budi Tjandra Negara*[1], Richardus Eko Indrajit [2], Erick Dazky[3]
[1,2,3] Program Magister Teknologi Infomasi, Universitas Pradita, Tanggerang Indonesia
Scientia Business Park, Jl. Gading Serpong Boulevard No.1, Tangerang, telp: (021) 55689999
e-mail: *[1] **budi.tjandra@student.pradita.ac.id**, [2] eko.indrajit@pradita.ac.id,
[3]erick.dazki@pradita.ac.id

Abstrak

The importance of a framework for a company engaged in application development makes the diversity of choices that a company must choose which is the best. Considering using a programming language framework is important, with the rapid evolution of programming language frameworks and the demands of a target completion time in a project. Each framework has its own purpose, if a project involves many developers, then those developers will spend time to make sure that they don't obscure every line of the script which causes conflict with each other. To analyse what developers consider when choosing a framework, we use primary and secondary data, both qualitative and quantitative. Qualitative data processing using descriptive analysis based on interviews. The collected data is processed using the Analytic Hierarchy Process (AHP) method. The purpose of this research is to examine the implementation of Frontend Language Frameworks Developers. The results of the analysis show that the important elements in Frontend Language Frameworks Developers are Security. The alternative that has the most important role in implementing Frontend Language Frameworks Developers is Vue.js..

Keywords— *Frontend, Framework, AHP*

1. INTRODUCTION

The definition of a good frontend framework does not directly determine the interactivity of the application system, but also knows it with future business needs. In the design process, the concept of "group" was designed in the framework due to the consideration of application integration for multi-systems in the future.[1] If this framework does not exist then the web page will never exist. The basics of all frontend languages include HTML (Hyper Text Markup Language) as the backbone in the web development process. HTML is a pure text file that can be created with any text editor. These documents are known as web pages. An HTML document is a document that is presented in the surfer's web browser. This document generally contains information or system interfaces on the internet. an HTML document is comprised of a string of characters or words.[2], *Markup* is representing a word or sentence in the form of images, tables, videos or other forms of object representation. Cascading style sheets (CSS) is a Web-based style sheet language that is used for the presentation of Web documents.[3] *JavaScript* is a language that has started to be imperative or even base programming where programmers can write commands, in contrast to HTML and CSS which are still based on declarations.

With the development of frontend framework technology, it is easier for programmers to develop web applications, the choice of frameworks is becoming more diverse, among others based on JavaScript including (1) Angular.js, (2) Backbone.js, (3) Ember.js, (4) React.js, (5) Vue.js.[4] by conducting interviews with developers who have more than 5 years of experience, we can establish 6 (six) important criteria for programmers to consider using the framework, including:

1. Community Support

The more popular a framework, the more communities that can provide knowledge and experience sharing in forums that exist in cyberspace. This will be very helpful in solving problems encountered in development.

2. Performance

The speed performance of the frontend framework is a concern for programmers, starting from the development process until the web application is running, bandwidth usage is a concern when this program is implemented.

3. Feature

Each *javascript* language framework has a different approach to handling Document Object Model (DOM) actions, web browser events handling, and accommodating a different experience for a web developer. DSL (Domain specific language) is a special language commonly used by all developers based on the JavaScript framework.[5] Each framework has a variety of different syntax structures, for example *React Framework* was introduced with the use of JSX to write DOM components, while the *Ember framework* used Handlebars.

DSL must be converted into a series of Javascript or HTML structures, the conversion from DSL to DOM structures cannot be read directly in a Web browser is carried out by devices or library archives which are usually included in a framework used. Because the basic concept of the DOM structure on web pages consists of HTML, CSS and Javascript.[5]

The transformation used by the framework has two main benefits, namely:

3.1. Ability to write code using the latest language features and turn it into code that works on everyday devices. For example, a web page developed using JavaScript with the latest up-to-date language features, but the Javascript code used still works in older browsers that don't support these features.

3.2. Option to write code in a completely different language and convert it to a web compatible language.

Although it is possible to build a framework without using DSL, but using frameworks that are commonly implemented now (React, Angular, Vue, etc) it is hoped that writing program code will be simpler and easier to find help from the framework user community.

4. Complexity

Over the last couple decades, web development has progressed significantly. In reality, the tech industry as a whole does. Today's web developers have access to a wide range of powerful tools designed to make their jobs easier and more productive.

JavaScript frameworks and libraries have become an essential part of modern web development. Since the mid-2000s, the JavaScript ecosystem has exploded, spawning a slew of frameworks that let developers build increasingly complex websites and applications. Frameworks have arguably transformed User Interface (UI) development. Allows developers to save time and create feature-rich solutions with significantly fewer resources than previously possible. Growth, on the other hand, does not come without a cost. The evolution of JavaScript frameworks appears to be out of control, leaving many forward-thinking web developers wondering if the era of JS-based frameworks is drawing to a close and new approaches to build websites and applications are on the way.

The JavaScript framework ecosystem is a good example. The first JS framework and library, notably jQuery, was immediately embraced since it provides numerous easy-to-use, straightforward, and automated functionalities across all sections of the web development process. However, as the number of new JavaScript frameworks and libraries has increased over time, the JQuery framework's popularity has waned. Newer frameworks like AngularJS, NodeJS, ReactJS, VueJS, KnockoutJS, EmberJS, NextJS, BackboneJS, and others have replaced it.

Old frameworks soon become obsolete, allowing new frameworks to enable more functionality while increasing complexity and variety. The average JS framework lifecycle is only a few years long, with "rapidly gaining prominence and then gradually declining in popularity as developers adopt newer technologies," according to the report.

This is unavoidable, given the increasing complexity of new frameworks and libraries. This has resulted in several significant problems for the web development community today.

Although the use of JavaScript frameworks for web development has several advantages, the needs and technological solutions are also evolving all the time, making the developer community more and more wondering which framework has the features to accommodate evolving needs and the chosen solution.

The following weaknesses may be encountered in the modern JS Framework ecosystem, namely;

4.1. Increasing complexity of web development

The escalating complexity of new JS frameworks makes developing and maintaining online applications increasingly challenging. The solution frequently reveals itself due to the underlying incompatibility of the various versions of the libraries used and the framework.

4.2. Increasing demand for web developers

It also places more strain on web developers who would continually learn new frameworks as feature additions change or increase, with various versions being utilized by different organizations and on different projects. As a result, many developers grumble about their inability to stay up.

4.3. Reduce innovation

The general use of frameworks and library archives, makes the final product created with the framework much more standard and less original. So it can be said with certainty that the current JS framework limits innovation in web development with a unique originality.

4.4. Decreased website and app performance speed

Although utilizing a JavaScript framework speeds up web development, it has the opposite effect on the performance of the websites and apps created. Due to the process of changing the DSL structure to the DOM structure, a huge quantity of computer code impacts the loading speed of the website on the internet browser, making the website appear slower.

5. Scalable

The capacity of a system, network, or process to cope with higher workloads when extra resources are necessary is defined as scalable or website scalability (usually related to hardware requirements). The ratio of increased system performance to increased resource use can be used to measure scalability. Adding resources to a system with low scalability improves performance just slightly, and at a certain point, adding resources has no effect at all.

Under typical circumstances, program performance varies in direct proportion to the load of the process being run. The strain on the quality of program performance would eventually rise as well. The saturation point has been achieved when there is an anomalous surge in the program's performance process at a specific point. Ramp-up tests are used to determine where the saturation point is. These tests aid in determining the dependability and scalability of web applications, as well as identifying system bottlenecks that impede the overall system process's effectiveness. Which factors, then, influence the web system's scalability?

There are several informations that affect the scalability of a web application, in other words;

5.1. Architecture, web system design as a whole starting from designing the hardware and software requirements that will be used.

5.2. Framework load, selecting the right framework where it predicts the need for additional features and the performance quality of the selected framework.

5.3. Continous Design, the addition of the structure of the algorithm and program code can affect the performance of the results of the web application that is run.

5.4. Application load testing, When application performance testing is carried out, it usually finds bottlenecks that cause the application load to be heavy, Developers must eliminate them by changing some scripts and algorithms

5.5. Hardware limitations. As always, it's not just software that can affect scalability. The integration of third-party components or the hardware used is the most common cause of process performance in applications that are built.

As an example of what can be done to the scalability of the architecture that can be used:

a) **Frontend versus Backend**

The front-end is where these two come together: the project does not sit alone, but within an environment and context which defines the need and context for the project.[6] As in opening a website address, it must appear as quickly as possible and reduce the timing of the background process generated by the Backend system. While the backend is the place where the process in an information system or application runs, where it processes when the data is added, changed and deleted. If there are additional pages or various features being developed, it is necessary to pay attention to the load of the timing process between the Front End and the Backend. A back-end is used to examine the similarity of two identity vectors.[7]

b) **Scalability of multitier architecture**

The Multitier is based on framework and subdivision techniques proposed for network deployment, as well as efficient routing protocols, with the goal of achieving load balancing in the network and converting long-distance communication into shorter multi-hop distance communication, thereby extending the network lifetime.[8] The multi-tier model is a software architecture that consists of three parts: a client, an application server to which the client connects, and a database server to which the program is run. Multi-tier architecture is based on two fundamental ideas; in other words:

- Minimize and simplify component functions on the client side, removing unnecessary functions
- Eliminate queries on the database that should not need to be run.
- Distribute Processing module can be done intelligently, for example with more than one separate server core

Multitier architecture intelligently divides processing modules, which is done in this case on one or more different servers. The application server (Application Layer) serves as a client to the database server and performs server duties such as interacting with users. Furthermore, various application servers can communicate with one another to more precisely divide the system into function blocks with distinct roles.

c) **Vertical versus horizontal scalability**

To enhance overall performance, vertical scalability entails boosting the performance of each system component. As requirements and technology evolve, this means the capacity to replace old computer system components with better and quicker ones. Vertical scaling is the most straightforward since it does not need any changes to the system's application applications. Separating a system into smaller structural components, distributing them across individual physical computers, and increasing the number of servers executing the same job at the same time is known as horizontal scalability. Horizontal scalability refers to the addition of new nodes, servers, and processors to the system in order to increase overall performance. This scaling strategy may necessitate program adjustments in order to fully utilize the additional resources.

Horizontal scalability refers to the network's capacity to expand, allowing more hardware or software entities to be accommodated in the network. Whereas vertical scalability relates to the ability to build the efficiency of existing software or hardware by adding more resources.[9]

The basic principles that must be considered are the scalability of the selection of the framework used for web application development, such as :

a) Good performance

The smooth performance of the framework makes the web application comfortable for users to use. When the performance of the framework runs slowly, it causes delays in the work process carried out by users as well. Therefore, the chosen framework must have the performance speed of transformation from DSL to DOM and the speed of rendering data on DSL.

b) Continuous availability

The selected framework can run at any time on any internet browser. Framework compatibility should support the development of browser technology without having to change the compiled program code.

c) Data access speed

Working with data generally entails plenty of potential performance concerns. The reality is that third-party technology is frequently utilized to store and retrieve data, and merging library archives needed by each component is a difficult process. In the main application and database services section, it is often found that there are glitches or incompatibilities that cause the system to stop.

d) Troubleshooting

The framework's systems must be simple to run, maintain, and upgrade. Furthermore, any issues that may occur during the program's release stage should be simple to locate and resolve. With adequate structuring and division of the application into reasonably independent components, good outcomes can be achieved. As a result, damage to a framework module does not reduce the amount of code in other modules or the amount of modification that must be made.

e) Response speed

The speed of web applications created using a framework is undeniably essential. This includes not just data retrieval from storage or overall performance, but also the speed with which operations and responses are carried out.

f) Security

Computed code may be an enticing entrance for hackers no matter what framework you use, whether it's React.js, Angular, Vue.js, or you're just a front-end developer. As a front-end developer, your primary concerns are generally performance, SEO, and UI/UX. The security aspect, on the other hand, is frequently ignored.

It's astonishing to find that so many frameworks make themselves vulnerable to cross-site scripting (XSS) assaults. There are dangerous methods, such as React's hazardlySetInnerHTML or Angular's Security Trust bypass API. In terms of security, it's important to realize that the front end now has the same responsibilities as the back end or DevOps. Thousands of harmful assaults can be launched from the front end.

Some of the things to understand are the most common and these will cover most types of attacks that might occur to a framework user.

1. Unrestricted File Upload

A system is attacked by uploading a malicious file to the server and then executing it. An unlimited file upload feature in a web application would let attackers to submit files containing malicious code that might be run on the server.[10] Such attacks can include: file system or database overload, complete system takeover, client-side attacks, forwarding attacks to back-end systems, or simple tampering.

2. Clickjacking

The clickjacking attack entices the victim into clicking on a certain feature of a webpage while the victim intended to engage with the content of another website.[11] Users may unintentionally submit credentials or sensitive information, download malware, view malicious web pages, purchase items online, or transfer money as a result of these assaults.

3. XSS Attack

An attack in which a malicious script is injected into a web page as a browser-side script..[12] A flaw in the website allowed this attack to be successful and widespread.

4. SQL injection

An attack in which malicious code is inserted into SQL queries with the goal of destroying the database via a table column input. It becomes the role of the DB administrator to configure secure access credentials. and the role for frontend developers not to insert commands or functions in the field table.[13]

5. Denial-of-service attack (DoS attack)

An attack in which a server or its resources are rendered inaccessible to their intended users by flooding the server with bandwidth traffic.[14] By bombarding requests on a network and server it makes the resource seem unavailable to the user.

6. Man in the middle attack / session hijacking

The attack is carried out by intercepting data packets intended to steal credential data such as passwords and personal data. it is important for developers to mask URLs in changing sessions or encrypting in a data transfer.[15]

2. METHODOLOGY

In selecting the criteria and objects of this paper, we use the Analytic Hierarchy Process analysis tool (AHP). AHP is commonly used and in accordance with the object and purpose of this paper to make qualitative and quantitative decisions based on all aspects it has. (Setiawan, 2014).
These are steps of the AHP method according to Saaty (1991):

2.1. We determine the objectives and detailed criteria and desired objects to solve the problem.

2.2. From a broad perspective, the hierarchical structure's organization (from the top levels to the level where it is possible to intervene to solve the problem). Create a hierarchical structure with broad goals at the top, then sub-objectives, criteria, and viable alternatives at the bottom.

2.3. Create a pairwise comparison matrix for each important element's contribution or effect on each influencing criterion at a higher level. In this matrix, pairs of items are evaluated against a higher-level criteria. Comparison based on the decision maker's "judgment" with the decision maker's judgment estimating the level of relevance of a factor in comparison to other elements.

2.4. In step 3, gather all of the information you'll need to create the matrix set. When there are a lot of individuals engaged, distributing work correctly may make everyone's job easier.

2.5. The priority is pursued and consistency is checked after gathering all pairwise comparison data and inserting the opposite values together with the number 1 entry along the main diagonal.

2.5.1. In this example, the individual opinion matrix is A1, A2,....An is a set of items at one level in the hierarchy. An n x n matrix is formed by quantifying opinion using the outcomes of pairwise comparisons. The aij value, as shown in Table 1, is the value of the opinion matrix of the comparison findings, which indicates the value of Ai's relevance to Aj.

Tabel 1. Individual Opinion Matrix

X	A1	A2	A3	…	An
A1	a11	a12	a13	…	a1n
A2	a21	a22	a23	…	a2n
A3	a31	a32	a33	…	a3n
…	…	…	…	…	
An	an1	an2	an3	…	Ann

The priority is pursued and consistently checked after gathering all of the pairwise comparison data and inserting the inverse values together with the number 1 entry along the main diagonal. In the synthesis, there are two steps of processing:

2.5.2. Horizontal processing, i.e :

a. Row Multiplication

$$zi = \sqrt[n]{\prod_{k-1}^{n} aij} \qquad (1)$$

Zi	=	*vector eigen*
n	=	number of respondent*s*
i,j	=	*1,2,3,....n*

b. The priority vector calculation (eigenvector) is:

$$VPi = \frac{\sqrt[n]{\prod_{k-1}^{n} aij}}{\sum_{1-i}^{n} \sqrt[n]{\prod_{k-1}^{n} aij}} \qquad (2)$$

eVP = priority vector element *1*

c. Calculation of the maximum feature root (eigen value)

VA = (aij) x VP

with VA = (vai)

$$VB = \frac{VA}{VP} \qquad (3)$$

with VB = (vbi)

$$\lambda maks \ = \frac{1}{n} \sum_{i-k}^{n} vbi$$

VBi = *1, 2, 3, n*

VB = *Value Eigen*

VA = *Vector between*

lmax = *Value Eigen maximum*

d. Calculation of the ratio of inconsistencies.

To determine the consistency of respondents' replies, the consistency measurement value is required. This measurement is used to assess the consistency of the responses, which has an impact on the results' validity.

$$CI = \frac{\lambda maks - n}{n - 1} \qquad (4)$$

2.5.3. To find out whether the CI with a certain amount is good enough or not, it is necessary to know the ratio that is considered good, that is, if CR <0.1. The CR formula is

$$CR = \frac{CI}{RI} \qquad (5)$$

2.5.4. RI is a random index value or RI issued by the oarkridge laboratory. Vertical processing is used to prioritize the influence of each element at a certain decision hierarchy level on the main target. This can be seen in the Table 2.

Tabel 2. RI matrix of order 2 s/d 10

Order n	Random Index (RI)
2	0.00
3	0.58
4	0.90
5	1.12
6	1.24
7	1.32
8	1.41
9	1,45
10	1,49

2.5.4. Multiply each consistency index by the priority of the relevant criterion and sum the results to determine the hierarchy's consistency. According to the dimensions of each matrix, these results are split by comparable assertions using a random consistency index. After the matrix has been processed horizontally with Expert Choice 11 computer program, the inconsistency ratio is calculated. If the inconsistency ratio is more than 10%, the information quality must be evaluated and improved, including the way questions are utilized while filling out the questionnaire and how respondents are directed to fill out the questionnaire.

3. RESULTS AND DISCUSSION

Hierarchical Structure

Figure 1 shows the specifications of the hierarchical structure utilized, which consists of three (3) layers. Goals, or the Best Frontend Framework, is the first stage. The goal of the criterion selection process is to establish the different objectives that are used to create fronted applications in a firm.

At level two is an element consisting of six (6) criteria, that are *Community Support*, *Performances*, *Features*, *Complexity*, *Scalable*, *Se*curity. These six criteria are important elements in considering the use of the framework. The third level is alternative actions taken to make it easier for programmers to develop JavaScript-based web applications including Angular.js, React.js, Ember.js, Vue.js dan Backbone.js.

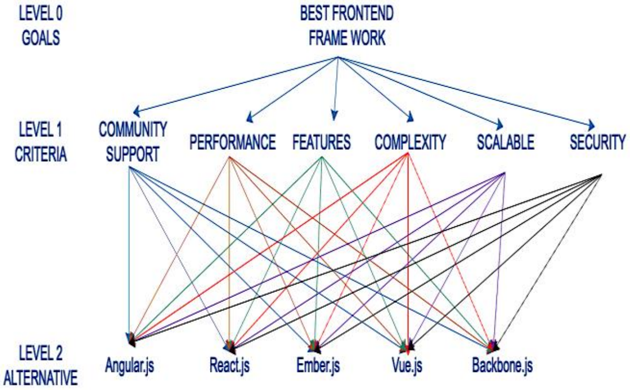


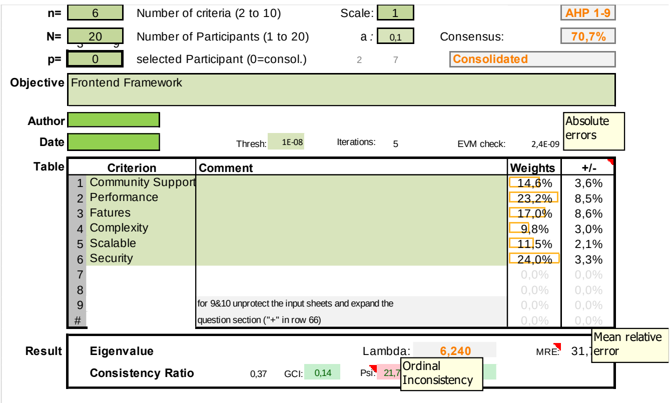
Figure 1. Hierarchical Structure of Best Fronted Frame Work.

Analysis of AHP Processing Results Vertically

Vertical processing is used to compile and view the overall priority of each element at a certain level against the main goal of the hierarchy. Vertical processing is carried out after the combined opinion matrix is processed horizontally and has met the inconsistency requirements of < 10 percent.

Criteria (level-2)

The results of processing at the criterion level can be seen in Table 3 which shows the results of the percentage of filling in the questionnaire by the selected respondents are as follows: *Security (24,0 %), Performance (23,2 %), Features (17,0 %), Community support (14,6 %), Scalable (11,5 %) dan Complexity (9,8 %).*



n= 6 Number of criteria (2 to 10) Scale: 1 AHP 1-9

N= 20 Number of Participants (1 to 20) a : 0,1 Consensus: 70,7%

p= 0 selected Participant (0=consol.) 2 7 Consolidated

Objective Frontend Framework

Author

Date Thresh: 1E-08 Iterations: 5 EVM check: 2,4E-09 Absolute errors

Table

	Criterion	Comment	Weights	+/-
1	Community Support		14,6%	3,6%
2	Performance		23,2%	8,5%
3	Fatures		17,0%	8,6%
4	Complexity		9,8%	3,0%
5	Scalable		11,5%	2,1%
6	Security		24,0%	3,3%
7			0,0%	0,0%
8			0,0%	0,0%
9		for 9&10 unprotect the input sheets and expand the	0,0%	0,0%
#		question section ("+" in row 66)	0,0%	0,0%

Result Eigenvalue Lambda: 6,240 MRE: 31,7 Mean relative error

Consistency Ratio 0,37 GCI: 0,14 Psi: 21,7 Ordinal Inconsistency

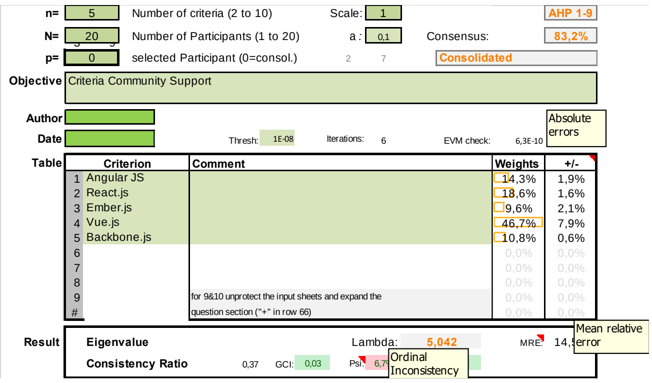
Figure 2. Vertical Processing Criteria Element

From these six criteria, it can be seen that the main criterion that needs to be considered in the best-fronted framework is security because the role of security in the application development process is very important in which there is a crucial database and becomes a guideline for a continuous application progress which refers to security standards.

Analysis of Horizontal AHP Processing Results

Horizontal processing aims to see the priority of an element against a level that is one level above that element in a decision hierarchy structure. The explanation of each element with horizontal assessment in the selection of the framework is explained as follows:

In the community support criteria, the first alternative priority is vue.js with a weight of (46.7%), then React.js with a weight of (18.6%), Angular.js with a weight of (14.3%), Backbone.js with a a weight of (10.8%) and ember.js with a weight of (9.6%). This proves that vue.js can run well with the direction of community support which we know is very much in control of the progress of a framework in meeting customer satisfaction.



n= 5 Number of criteria (2 to 10) Scale: 1 AHP 1-9

N= 20 Number of Participants (1 to 20) a : 0,1 Consensus: 83,2%

p= 0 selected Participant (0=consol.) 2 7 Consolidated

Objective Criteria Community Support

Author

Date Thresh: 1E-08 Iterations: 6 EVM check: 6,3E-10 Absolute errors

Table

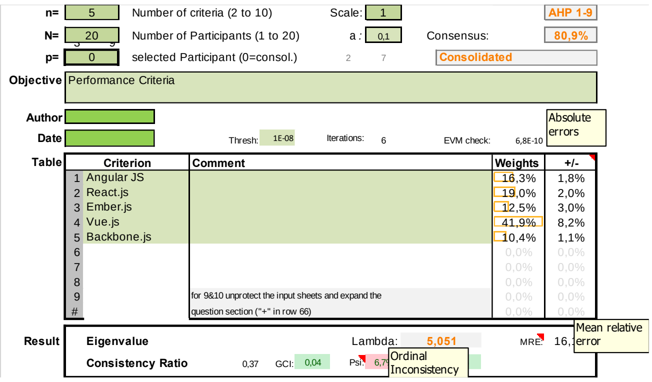
	Criterion	Comment	Weights	+/-
1	Angular JS		14,3%	1,9%
2	React.js		18,6%	1,6%
3	Ember.js		9,6%	2,1%
4	Vue.js		46,7%	7,9%
5	Backbone.js		10,8%	0,6%
6			0,0%	0,0%
7			0,0%	0,0%
8			0,0%	0,0%
9		for 9&10 unprotect the input sheets and expand the	0,0%	0,0%
#		question section ("+" in row 66)	0,0%	0,0%

Result Eigenvalue Lambda: 5,042 MRE: 14,5 Mean relative error

Consistency Ratio 0,37 GCI: 0,03 Psi: 6,7% Ordinal Inconsistency

Figure 3. Community Support for Level 2 Alternatives

On the performance criteria, the first alternative priority is vue.js with a weight of (41.9%), then React.js with a weight of (19.0%) , Angular .js with a weight of (16.3%), ember.js with a weight of of (12.5%) and backbone.js with a weight of (10.4%) This proves that vue.js shows the best speed performance so that it is the choice of programmers.



n= 5 Number of criteria (2 to 10) Scale: 1 AHP 1-9

N= 20 Number of Participants (1 to 20) a : 0,1 Consensus: 80,9%

p= 0 selected Participant (0=consol.) 2 7 Consolidated

Objective Performance Criteria

Author

Date Thresh: 1E-08 Iterations: 6 EVM check: 6,8E-10

Absolute errors

Table

	Criterion	Comment	Weights	+/-
1	Angular JS		16,3%	1,8%
2	React.js		19,0%	2,0%
3	Ember.js		12,5%	3,0%
4	Vue.js		41,9%	8,2%
5	Backbone.js		10,4%	1,1%
6			0,0%	0,0%
7			0,0%	0,0%
8			0,0%	0,0%
9		for 9&10 unprotect the input sheets and expand the	0,0%	0,0%
#		question section ("+" in row 66)	0,0%	0,0%

Result Eigenvalue Lambda: 5,051 MRE: 16,1

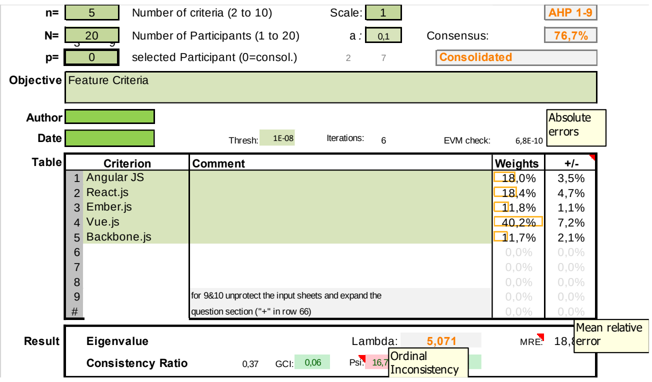
Mean relative error

Consistency Ratio 0,37 GCI: 0,04 Psi: 6,7

Ordinal Inconsistency

Figure 4. Performance Against Level 2 Alternatives

On the features criteria, the first alternative priority is vue.js with a weight of (40.2%), then React.js with a weight of (18.4%) , Angular .js with a weight of (18.0%), ember.js with a weight of of (11.8%) and backbone.js with a weight of (11.7%) This proves that vue.js has far more features and is better than its competitors.



n= 5 Number of criteria (2 to 10) Scale: 1 AHP 1-9

N= 20 Number of Participants (1 to 20) a : 0,1 Consensus: 76,7%

p= 0 selected Participant (0=consol.) 2 7 Consolidated

Objective Feature Criteria

Author

Date Thresh: 1E-08 Iterations: 6 EVM check: 6,8E-10

Absolute errors

Table

	Criterion	Comment	Weights	+/-
1	Angular JS		18,0%	3,5%
2	React.js		18,4%	4,7%
3	Ember.js		11,8%	1,1%
4	Vue.js		40,2%	7,2%
5	Backbone.js		11,7%	2,1%
6			0,0%	0,0%
7			0,0%	0,0%
8			0,0%	0,0%
9		for 9&10 unprotect the input sheets and expand the	0,0%	0,0%
#		question section ("+" in row 66)	0,0%	0,0%

Result Eigenvalue Lambda: 5,071 MRE: 18,8

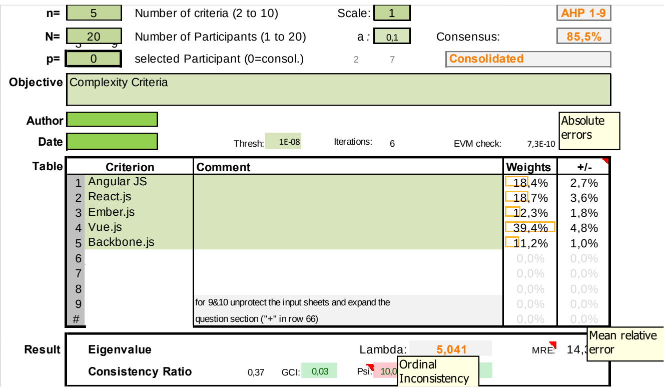
Mean relative error

Consistency Ratio 0,37 GCI: 0,06 Psi: 16,7

Ordinal Inconsistency

Figure 5. Features of Alternative Level 2

On the complexity criteria, the first alternative priority is vue.js with a weight of (39.4%), then React.js with a weight of (18.7%) , angular.js with a weight of (18.4%), ember.js with a weight of by (12.3%) and backbone.js with a weight of (11.2%) This proves that vue.js can answer the complexity challenges of various applications that are developing today.



n= 5 Number of criteria (2 to 10) Scale: 1 AHP 1-9
N= 20 Number of Participants (1 to 20) a : 0,1 Consensus: 85,5%
p= 0 selected Participant (0=consol.) 2 7 Consolidated
Objective Complexity Criteria
Author
Date Thresh: 1E-08 Iterations: 6 EVM check: 7,3E-10 Absolute errors

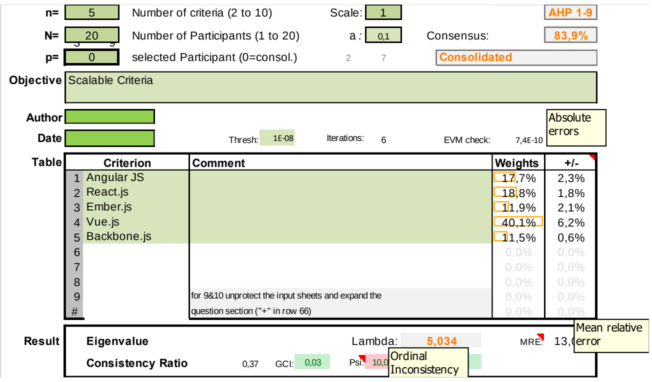
Table

	Criterion	Comment	Weights	+/-
1	Angular JS		18,4%	2,7%
2	React.js		18,7%	3,6%
3	Ember.js		12,3%	1,8%
4	Vue.js		39,4%	4,8%
5	Backbone.js		11,2%	1,0%
6			0,0%	0,0%
7			0,0%	0,0%
8			0,0%	0,0%
9		for 9&10 unprotect the input sheets and expand the	0,0%	0,0%
#		question section ("+" in row 66)	0,0%	0,0%

Result Eigenvalue Lambda: 5,041 MRE: 14, Mean relative error
Consistency Ratio 0,37 GCI: 0,03 Psi: 10,0 Ordinal Inconsistency

Figure 6. Complexity of Level 2 Alternative

On the Scalable criteria, the first alternative priority is vue.js with a weight of (40.1%), then React.js with a weight of (18.8%), angular.js with a weight of (17.7%), ember.js with a weight of of (11.9%) and backbone.js with a weight of (11.5%) This proves that vue.js can answer the challenges of a larger application workload compared to other frameworks.



n= 5 Number of criteria (2 to 10) Scale: 1 AHP 1-9
N= 20 Number of Participants (1 to 20) a : 0,1 Consensus: 83,9%
p= 0 selected Participant (0=consol.) 2 7 Consolidated
Objective Scalable Criteria
Author
Date Thresh: 1E-08 Iterations: 6 EVM check: 7,4E-10 Absolute errors

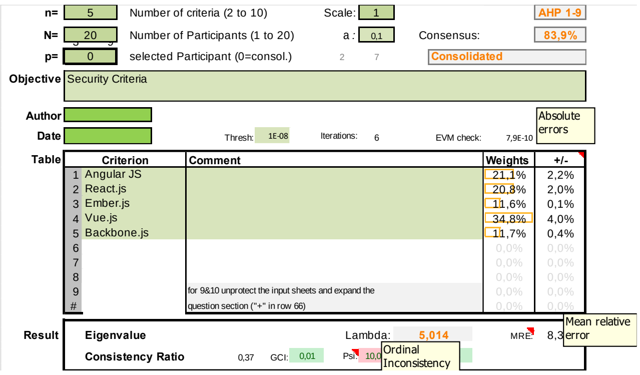
Table

	Criterion	Comment	Weights	+/-
1	Angular JS		17,7%	2,3%
2	React.js		18,8%	1,8%
3	Ember.js		11,9%	2,1%
4	Vue.js		40,1%	6,2%
5	Backbone.js		11,5%	0,6%
6			0,0%	0,0%
7			0,0%	0,0%
8			0,0%	0,0%
9		for 9&10 unprotect the input sheets and expand the	0,0%	0,0%
#		question section ("+" in row 66)	0,0%	0,0%

Result Eigenvalue Lambda: 5,034 MRE: 13, Mean relative error
Consistency Ratio 0,37 GCI: 0,03 Psi: 10,0 Ordinal Inconsistency

Figure 7. Scalable of Level 2 Alternative

In the security criteria, the first alternative priority is vue.js with a weight of (34.8%), then Angular.js with a weight of (21.1%), React.js with a weight of (20.8%), backbone.js with a weight of of (11.7%) and ember.js with a weight of (11.6%) This proves that vue.js is able to provide the highest level of security, so that the security of every application developed is guaranteed to be secure.



n= 5 Number of criteria (2 to 10) Scale: 1 AHP 1-9
N= 20 Number of Participants (1 to 20) a : 0,1 Consensus: 83,9%
p= 0 selected Participant (0=consol.) 2 7 Consolidated
Objective Security Criteria
Author
Date Thresh: 1E-08 Iterations: 6 EVM check: 7,9E-10 Absolute errors

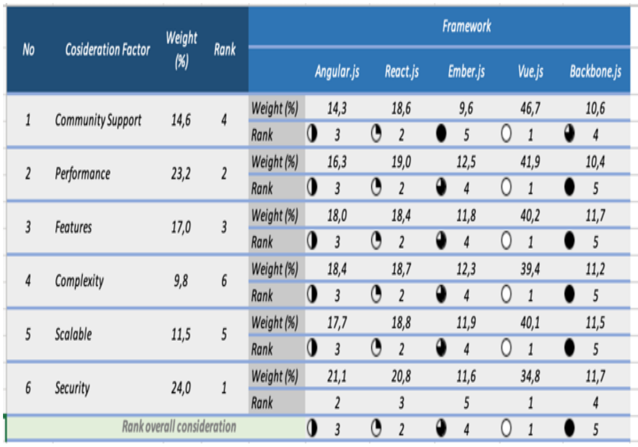
Table

	Criterion	Comment	Weights	+/-
1	Angular JS		21,1%	2,2%
2	React.js		20,8%	2,0%
3	Ember.js		11,6%	0,1%
4	Vue.js		34,8%	4,0%
5	Backbone.js		11,7%	0,4%
6			0,0%	0,0%
7			0,0%	0,0%
8			0,0%	0,0%
9		for 9&10 unprotect the input sheets and expand the	0,0%	0,0%
#		question section ("+" in row 66)	0,0%	0,0%

Result Eigenvalue Lambda: 5,014 MRE: 8,3 Mean relative error
Consistency Ratio 0,37 GCI: 0,01 Psi: 10,0 Ordinal Inconsistency

Figure 8. Security of Level 2 Alternative

Of all the criteria that have been chosen, it turns out that 'security' has the most role from developer considerations and from all the selected frameworks it turns out that vue.js ranks first among all its competitors, not only on the criteria of community support, performance, features, complexity and scalable, but even the most important criteria prove that vue.js is indeed the best framework.



No	Cosideration Factor	Weight (%)	Rank		Framework				
					Angular.js	React.js	Ember.js	Vue.js	Backbone.js
1	Community Support	14,6	4	Weight (%)	14,3	18,6	9,6	46,7	10,6
				Rank	◑ 3	◔ 2	● 5	○ 1	◕ 4
2	Performance	23,2	2	Weight (%)	16,3	19,0	12,5	41,9	10,4
				Rank	◑ 3	◔ 2	◕ 4	○ 1	● 5
3	Features	17,0	3	Weight (%)	18,0	18,4	11,8	40,2	11,7
				Rank	◑ 3	◔ 2	◕ 4	○ 1	● 5
4	Complexity	9,8	6	Weight (%)	18,4	18,7	12,3	39,4	11,2
				Rank	◑ 3	◔ 2	◕ 4	○ 1	● 5
5	Scalable	11,5	5	Weight (%)	17,7	18,8	11,9	40,1	11,5
				Rank	◑ 3	◔ 2	◕ 4	○ 1	● 5
6	Security	24,0	1	Weight (%)	21,1	20,8	11,6	34,8	11,7
				Rank	2	3	5	1	4
Rank overall consideration					◑ 3	◔ 2	◕ 4	○ 1	● 5

Figure 9. Result of Comparison of Criteria Factor Values and Framework Selection Alternatives

4. CONCLUSION

For companies engaged in IT Consultants, the opinion of this developer is a consideration for the company's management to uniform the framework for all developers in building the frontend and from these results can describe the medium-term strategy in determining a framework.

5. SUGGESTION

Saran-saran untuk untuk penelitian lebih lanjut untuk menutup kekurangan penelitian. Tidak memuat saran-saran diluar untuk penelitian lanjut.

References

[1] K. Li, Y. Ding, D. Shen, Q. Li, and Z. Zhen, “The design and research of front-end framework for microservice environment,” *Proc. - 2020 Int. Conf. Comput. Inf. Big Data Appl. CIBDA 2020*, pp. 124–127, 2020, doi: 10.1109/CIBDA50819.2020.00036.

[2] C. Opara, B. Wei, and Y. Chen, “HTMLPhish: Enabling accurate phishing web page detection by applying deep learning techniques on HTML analysis,” *arXiv*, 2019.

[3] J. G. Ndia, G. M. Muketha, and K. K. Omieno, “a Survey of Cascading Style Sheets Complexity Metrics,” *Int. J. Softw. Eng. Appl.*, vol. 10, no. 03, pp. 21–33, 2019, doi: 10.5121/ijsea.2019.10303.

[4] S. Delcev and D. Draskovic, “Modern JavaScript frameworks: A Survey Study,” *2018 Zooming Innov. Consum. Technol. Conf. ZINC 2018*, pp. 106–109, 2018, doi: 10.1109/ZINC.2018.8448444.

[5] L. Shen, X. Chen, R. Liu, H. Wang, and G. Ji, “Domain-Specific Language Techniques for Visual Computing: A Comprehensive Study,” *Arch. Comput. Methods Eng.*, vol. 28, no. 4, pp. 3113–3134, 2021, doi: 10.1007/s11831-020-09492-4.

[6] T. Williams, H. Vo, K. Samset, and A. Edkins, “The front-end of projects: a systematic literature review and structuring,” *Prod. Plan. Control*, vol. 30, no. 14, pp. 1137–1169, 2019, doi: 10.1080/09537287.2019.1594429.

[7] Z. Bai, X.-L. Zhang, and J. Chen, “Partial AUC Metric Learning Based Speaker Verification Back-End,” no. November, pp. 380–384, 2020, doi:

10.21437/odyssey.2020-53.

[8] A. Shukla and S. Tripathi, “A multi-tier based clustering framework for scalable and energy efficient WSN-assisted IoT network,” *Wirel. Networks*, vol. 26, no. 5, pp. 3471–3493, 2020, doi: 10.1007/s11276-020-02277-4.

[9] A. Gupta, R. Christie, and R. Manjula, “Scalability in Internet of Things: Features, Techniques and Research Challenges,” *Int. J. Comput. Intell. Res.*, vol. 13, no. 7, pp. 1617–1627, 2017.

[10] J. Huang, Y. Li, J. Zhang, and R. Dai, “UChecker: Automatically Detecting PHP-Based Unrestricted File Upload Vulnerabilities,” *Proc. - 49th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Networks, DSN 2019*, pp. 581–592, 2019, doi: 10.1109/DSN.2019.00064.

[11] K. Jamwal and L. Sen Sharma, “Clickjacking Attack: Hijacking User’s Click,” *Int. J. Adv. Netw. Appl.*, vol. 10, no. 01, pp. 3735–3740, 2018, doi: 10.35444/ijana.2018.100110.

[12] S. Goswami, N. Hoque, D. K. Bhattacharyya, and J. Kalita, “An unsupervised method for detection of XSS attack,” *Int. J. Netw. Secur.*, vol. 19, no. 5, pp. 761–775, 2017, doi: 10.6633/IJNS.201709.19(5).14.

[13] B. Appiah, E. Opoku-Mensah, and Z. Qin, “SQL injection attack detection using fingerprints and pattern matching technique,” *Proc. IEEE Int. Conf. Softw. Eng. Serv. Sci. ICSESS*, vol. 2017-Novem, pp. 583–587, 2018, doi: 10.1109/ICSESS.2017.8342983.

[14] L. Liang, K. Zheng, Q. Sheng, W. Wang, R. Fu, and X. Huang, “A denial of service attack method for iot system in photovoltaic energy system,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10394 LNCS, pp. 613–622, 2017, doi: 10.1007/978-3-319-64701-2_48.

[15] A. Mallik, A. Ahsan, M. M. Z. Shahadat, and J. C. Tsou, “Man-in-the-middle-attack: Understanding in simple words,” *Int. J. Data Netw. Sci.*, vol. 3, no. 2, pp. 77–92, 2019, doi: 10.5267/j.ijdns.2019.1.001.