Bachelor Degree Project

Evaluating JavaScript frameworks

Author: Josefin Salomaa
Supervisor: Jesper Andersson
Semester. VT 2020
Subject: Computer Science

Abstract

Web application frameworks have become a common part of software development, with each programming language having its own specific frameworks. However, choosing the right framework can be a daunting task when there are so many frameworks on the market and factors to consider. Previous research has attempted to solve this problem by evaluating frameworks based on several different metrics. While the results offer a way to rank the frameworks, they do not tell us much about the actual process of evaluating a framework, or why certain metrics matter and in what situation. This project aimed to answer these questions by conducting a literature review to gather information about the existing evaluation methods and metrics. Based on these results, an evaluation criteria was developed to provide an overview of when to evaluate what. Furthermore, a user survey was done to measure the usability of three popular JavaScript frameworks. Lastly, a controlled experiment was conducted to test the performance of the three frameworks. The results show that different frameworks have different strengths and weaknesses as they received different scores from the methods. Moreover, the results show that when choosing a framework, one has to consider which factors matter most as it may not always be possible to find a framework that has it all.

Keywords: JavaScript, Angular, React, Vue, literature review, user survey, controlled experiment, usability, performance, evaluation criteria.

Contents

1 Introduction

Web frameworks are a common part of web development and there are many frameworks for different programming languages. In the jungle of frameworks, it can be difficult to know which framework to pick. What factors should be considered when choosing a framework, and do the factors change depending on what situation we want to use the framework for? This project aims to provide a better understanding of how to evaluate frameworks, and then evaluate three popular JavaScript frameworks based on these findings.

1.1 Background

Frameworks as a technology were created with the purpose of realizing established software concepts and software designs and implementations to lower the price and enhancing the software quality [1]. The point of using a framework is to obtain a way to develop applications that are essentially already half-finished. Fayad and Smith state that the main advantages of a framework are: “modularity, reusability, extensibility, and inversion of control”. [1]

While there is more than one type of framework on the market, this project focuses on web application frameworks (or simply web frameworks). A web application framework can be described as a blueprint which provides developers with a standardized way of creating web applications. The frameworks can offer interaction with data persistence, security, authentication, templates etc. [2]

Two of the most common types of frameworks in web development are server-side and client-side frameworks. The server-side frameworks run on a web server while the client-side frameworks run in the web browser (which are normally JavaScript). Server-side are essentially backend and client-side are frontend [3].

JavaScript is a popular high-level interpreted programming language that is mainly used when creating web applications and dynamic web pages. It offers support for functional, imperative, as well as object-oriented programming [4].

1.2 Related work

Many attempts have been done to evaluate JavaScript frameworks in the past. One of the most common ways to evaluate the frameworks is by benchmarking them using different metrics. An example of this was done by Gizas, Christodoulou, and Papatheodorou who tested a few popular JavaScript frameworks by looking at the performance, quality as well as the validation of them. This was done by using specific test tools for this purpose [5].

Benchmarking frameworks, however, might not give a fair view of the frameworks as research has shown that JavaScript benchmarks tend to be small and act in a different way than the real applications. Moreover, benchmarks do not always display the same behaviors that can be found in real applications [6].

Evaluating frameworks can also be done by conducting surveys that were done by Pano, Graziotin, and Abrahamsson [7]. Their survey aimed to find the aspects which made the developers choose a JavaScript framework. The survey was based on the expectations of the performance and exertion, the effects of the social aspects, among other things.

The examples above demonstrate the need and methods for evaluating frameworks, however, when assessing related research, the author found that there was a shortage of information about when one should evaluate what.

1.3 Problem formulation

The goal of this project is to gain a better understanding of how one can evaluate frameworks depending on different factors, which aspects should be tested, how can they be tested, and then use this knowledge for evaluating three popular frameworks (Angular, React.js, and Vue.js). The problem can be summed up into two main questions:

- *How can the frameworks be evaluated?*
- *What should be evaluated (e.g. which metrics)?*

1.4 Objectives

In order to answer the questions mentioned in 1.3 Problem formulation, the following objectives have been set:

- O1: An investigation of how to evaluate frameworks will take place with the purpose of identifying certain metrics and methods that stand out in terms of how often they occur.
- O2: An evaluation criteria will be created to demonstrate what should be evaluated when evaluating frameworks (e.g. which method should

be used for what metric/factor). This will essentially work as a guide or reference model one can follow when evaluating frameworks.

- O3: A few selected frameworks will be tested based on the metrics in the evaluation criteria, in order to show how the criteria can be used and how an evaluation process can look like.

1.5 Scope/Limitation

Because of the limited time and size of the project, only three frameworks will be evaluated. Furthermore, only one programming language will be used. When selecting the metrics that will be measured for evaluating the frameworks, there will not be as many as there could be if it was a bigger project. The project will only focus on a few metrics, whereas a bigger project could investigate many more. The metrics that are investigated in this project were chosen based on their popularity (how often they occur in related research), and the metrics the author found would be more interesting for someone who is trying to choose between frameworks. Furthermore, the application that is being tested is very small compared to complex, real-life applications.

1.6 Target group

The results from this project can be useful for anyone who is trying to decide what frameworks to pick. They are also useful for those who are looking to evaluate frameworks but do not know how to do it.

1.7 Outline

Chapter 2 explains the methods that have been used for this project. Chapter 3 contains a literature review of evaluating JavaScript frameworks as well as information about the frameworks that are evaluated in the following chapter. Chapter 4 continues with the evaluation of the frameworks with both implementation and results. Chapter 5 covers the analysis of the results and a discussion about the results in relation to the problem formulation and related work. Chapter 6 consists of a conclusion of the paper and a few suggestions for future research.

2 Method

In order to answer the question of what should be evaluated (which metrics), a qualitative literature review was conducted to gain knowledge of the metrics that are commonly evaluated. The literature review further aimed to find the answer to the question of how to evaluate frameworks, by gathering data from previous evaluations.

For the question of how to evaluate frameworks, a controlled experiment was also performed with the purpose of demonstrating how an evaluation process can look like based on the evaluation criteria. For the same question, a quantitative user survey was created to get information from developers' own opinions of the frameworks.

2.1 Literature review

In order to gain a better understanding of how evaluation of frameworks works, it was essential to investigate this further. This was done by conducting a literature review which was done during the period of March-May 2020. Relevant information was gathered from Google Scholar, IEEE Xplore Digital Library, and DiVA portal.

The search terms that were used were:

- Evaluating
- Evaluate
- JavaScript
- Frameworks
- Benchmarking
- Benchmark
- Testing
- Assessing
- Web frameworks
- Web application frameworks
- How to

They were also used in different combinations:

- Evaluating JavaScript frameworks
- How to evaluate JavaScript frameworks
- Benchmarking JavaScript frameworks
- How to benchmark frameworks
- Benchmarking web frameworks

- How to evaluate web frameworks
- How to evaluate web application frameworks
- Benchmarking web application frameworks
- Assessing JavaScript frameworks
- Testing JavaScript frameworks
- Testing frameworks

50% of the sources being used were scientific and academic publications. The other half were websites with information about the frameworks that were later evaluated. In order to choose between the sources, an inclusion and exclusion criteria was used which is shown below:

Inclusion criteria:

- The included articles must be in English.

  When searching for the articles, the language was filtered to show only articles in English. When that was not possible, a quick look at the title and abstract was sufficient enough to know if the article would be in English or not. All articles had to meet this criteria, otherwise they were not chosen.
- The included articles must contain qualitative, quantitative or mixed methodology.

  Usually the introduction or method chapter tells the reader which methods will be used later in the paper. However, if an article did not contain these specific methods, it could still be chosen if they met the rest of the inclusion criterias.
- The included articles must be dissertations, journal articles, book chapters, or trusted web pages.

  When searching for the articles, it was possible to pick the format (journal articles, dissertations or book chapters). As for trusted web pages, the overall impression the author got from the web page was usually enough to determine if it was trustworthy or not (information about for example the framework Vue on Vue’s own web page seem highly trustworthy). For pages that did not seem as trustworthy, the text was compared to other results from searches to confirm that it had the same or similar information. All articles had to meet this criteria, otherwise they were not chosen.
- The included articles must cover JavaScript, web frameworks, benchmarking or other methods for evaluating software.

  When searching for the articles with the search terms above, the information needed to meet this criteria was usually in the title of the article or the abstract. When it was not, the author would start to read the article to determine whether there was anything of value in the text. All articles had to meet this criteria.

Exclusion criteria (all of the following criterias must be met)

- It uses a language other than English
- The web page does not seem trustworthy or up to date
- The article lacks a source
- The article lacks a date/year
- The article covers other types of frameworks instead of web frameworks and no general information about evaluating software

The articles were used to gain knowledge about the common ways of evaluating software. The author looked at metrics that were used in evaluating frameworks, which factors are commonly looked at, and what web developers find important when choosing a framework.

From conducting this literature review, the result was an answer to the question "how to evaluate frameworks?".

2.2 Controlled experiment and user survey

By analyzing the findings from the literature review, an evaluation criteria was created as a procedure of what to evaluate (e.g. specific metric). Since there are many different ways to evaluate a framework, only a few were included in this criteria. It was mainly based on how often they occur in previous research and how useful they seemed for someone who is looking to evaluate them (different types of metrics to show different methods instead of choosing several metrics that were very similar to each other). This was done to help in the next step of the project which was to evaluate three popular JavaScript frameworks (React, Vue, Angular, [8]). These frameworks were tested based on the evaluation criteria. It was done by a controlled experiment where the performance of the three frameworks was evaluated by benchmarking three different applications made up of the frameworks. The performance metric was chosen because of how common it is to test the performance of software, and to also demonstrate different types of methods. In this case, the controlled experiment would offer a more objective way of evaluating the frameworks. Since the focus of the experiment was to show how an evaluation process could look like, the choice of frameworks for this project was not too important. They were essentially randomly chosen based on popularity.

Lastly, a user survey was conducted to further demonstrate how to evaluate frameworks, and to discover differences between the selected frameworks. It consisted of quantitative questions and a Likert scale was used as the answer type. Furthermore, the survey targeted professional developers and students that use JavaScript.

2.3 Reliability and validity

Reliability means that the results are reliable, in other words if others try to repeat the work, they will get the same results

To ensure reliability for the literature review, the search terms and databases used to gather the relevant information has been described in the previous section. This is done in order to give an understanding of the literature review's context and to make it easier to compare it to other, similar, studies. Furthermore, it enables others to search for the same things.

To ensure that the controlled experiment will be reliable it is important to offer a detailed description of the environment (i.e. the setup of the experiment). This is to give others the chance to replicate the experiment more easily. There also needs to be detailed documentation on the experiment itself in terms of what is being tested and how it is being tested. While conducting the tests, it is important to ensure that the source code will hold the same standard no matter who is carrying out the experiment. This is of course not always possible to guarantee, however, there are steps that can be taken to reduce the risk of different results due to badly maintained code. By following the general guidelines of the frameworks it is more likely that the experiment produces the correct results.

For the survey, it is essential that it holds consistency and that the questions in the survey evoke the same kind of information every time they are asked. The questions in the survey used in this project were consistent by providing the same questions for each of the different frameworks.

Validity, on the other hand, means that the conclusions that are drawn are backed up by the work that has been done and the data it has produced. The conclusions must be valid. There are two types of validity; internal and external. The internal validity is about whether the results and conclusions are backed up by the data. The external validity is about if we can generalise the results.

When looking at the internal validity it is important to look at the risk of bias. For the literature review, there are different kinds of sources that are being used. While most of them are scientific and academic papers, there are also a few books and websites.

To ensure internal validity for the controlled experiment, it is important to explain what is being evaluated. For example, simply stating that a certain framework has the best performance is both vague and misleading, since other people might not agree with that statement. When it comes to external validity, one must consider the limitations of the test environment. Since the experiments in this project are only being measured on one browser, it would be disingenuous to claim that the results apply for other browsers as well. Therefore, the actual browser being used (and other factors for the test environment), is clearly stated.

When it comes to the survey, the proper set of questions need to be asked. This means that the questions will provide answers that are useful for the conductor of the survey. The survey in this project aims to measure the usability of the frameworks, which means that the questions need to be centered around the respondents' own experience.

There is also the risk of bias for the survey. For example, if the conductor of the survey lets their own experience and preferences of the frameworks interfere with the questions and how they are conveyed. To avoid this, it is essential to keep the questions as objective as possible, ask the same questions for all the frameworks, and draw conclusions from the participants' answers only. Moreover, it is not possible to take the survey more than once with the same account.

Lastly, when it comes to external validity of the survey, one must look at the participants and what generalisations can be drawn from this group. The survey in this project is focused on professionals and students who are familiar with JavaScript and JavaScript frameworks. This means that the results and conclusions apply for professionals and students.

2.4 Ethical Considerations

The only ethical considerations of this project are regarding the survey as it is important that the privacy of the people participating in the study must not be violated. The survey aims to be anonymous, which means that there are no questions regarding names, location, gender, or age. This is to ensure that there is nothing that can be used to identify anyone. To conduct the survey, the participants needed to have a Google account, however, the email addresses were not collected or accessible to the creator of the survey. The survey furthermore states its purpose so the participants know what their answers will be used for.

3 Frameworks

As mentioned in the first chapter, a framework is meant to offer developers a reusable way to develop applications that are more or less already half-finished. One can also think of frameworks as a blueprint for a group of applications. Developers are then able to build finished applications by applying this blueprint. Each programming language has its own set of frameworks and this is because the frameworks provide definite structures, methods etc for specific programming languages, as opposed to, for example, design patterns. The main purpose of using frameworks is to lower the price and time, and strengthen the software quality [1].

This chapter reports on the literature review about the evaluation of JavaScript frameworks. It covers information about the three frameworks that were chosen for the evaluation in chapter 4, considerations when choosing a frameworks as well as two common metrics that are explained in more detail.

3.1 Frameworks used in this project

Angular (also referred to as Angular 2+) is an open-source JavaScript framework that is used when developing web pages, desktop applications, and mobile applications. This framework is used alongside TypeScript, and it is a superset of JavaScript (similar to HTML and CSS). The part of the code that is done in TypeScript runs to JavaScript, and the browser then renders it [9].

React.js, also known as React or React JS, is an open-source JavaScript library made for creating user interfaces. It was developed by Facebook and can be used for developing both single-pages and mobile applications. React is especially useful when building big web applications that quickly modify the data with the page not needing to reload [10].

Vue.js is an open-source JavaScript framework for building user interfaces. It is targeted on the view layer and is mainly used for single-page applications and mobile applications [11].

	Angular	React	Vue
First release	2016	2013	2014
Type	MVC framework	JavaScript library	MVC framework

Scalable	Yes	Yes	Yes
Companies/websites that use it	Freelancer, Santander, Forbes	Facebook, Instagram, Khan Academy	9gag, Nintendo, Laravel
Latest version	9.0.0 Feb 2020	16.13.1 March 2020	2.6.11 Dec 2019

Table 3.1: information about the different frameworks [12], [13], [14], [15].

3.2 Considerations when choosing a framework

Pano, Graziotin, and Abrahamsson's [7] research showed that there are a few aspects that play a role when it comes to choosing a JavaScript framework. These aspects were considered by a mix of several people such as the client, developer, group, and group leader. Pano, Graziotin, and Abrahamsson stated that the aspects they found were "usability (attractiveness, learnability, understandability), cost, efficiency (performance, size), and functionality (automatization, extensibility, flexibility, isolation, modularity, suitability, updates)." [7].

Satrom [16] lists a number of considerations to evaluate when choosing the right framework. These considerations are focused on the environment of the framework, tools, business aspects, and the frameworks themselves. When looking at the environment of a framework, one looks at the framework's background and how long is has been on the market, as well as how long it may be around for, but also how popular it is, if a corporation is supporting it, and its community. The tools consideration look at the tools needed for the frameworks to work such as the UI and libraries, IDEs and other support for the tools. The business considerations center around the license of a framework, how support and improvements of the framework look like, and security of the frameworks. Lastly, for the frameworks themselves, one should look at the process of getting started, the skills needed, and the size and performance of the framework.

Through a thematic analysis, four themes that affect the choice of a framework were found. Duvander and Romhagen identified them as "demand, usability, community and reputation" [17]. Demand is related to the skills needed for the frameworks, how demand can be influenced by the location, and what impact the requirements of a project can have on the process of picking a framework. Usability in this study refers to the aspects that improve the development process, such as the learning period, documentation, and structure. It also includes the time it takes to start a project with a framework and the time it takes for the total time it takes to use a framework. Community refers here to the community of a framework. With an active community, it enhances the ability to search for solutions on online communities such as Stack Overflow. Maintaining the code is easier when there is a community made up of many users since it is easier to find developers to work with certain frameworks. Finally, reputation is about view of the frameworks and where this view comes from. The company behind a framework is sometimes a factor here, big and well-known companies such as Google or Facebook can offer a sense of reliability.

3.3 Performance

When evaluating a framework, performance is often one of the first things that come to mind. Performance can be split into two sections: (1) the time it takes to finish a task, and (2) at what rate the system can perform a task. From a user’s perspective, the time it takes to execute a program is the most crucial part when it comes to performance. This is because users do not care about what happens in the background or the specific of things, they mainly just want no delay. The management’s perspective, however, is centered around the rate the system can perform a task. Here the focus is on questions such as the number of operations a system is able to process or the number of requests a web server can process per second [18].

Testing the performance of a system is commonly known as benchmarking the system. Benchmarking essentially means assessing an application’s performance under specific conditions against the performance of another application. There are specific test tools designed for benchmarking. Not only can benchmarking offer a more objective way of comparing different applications or solutions, but it can also be beneficial for the evolution of software in more general terms by offering a standard way of measuring and assessing software [19].

A benchmark that was conducted by Gizas, Christodoulou, and Papatheodorou [5] demonstrates a common way to benchmark. They used the SlickSpeed Selectors test framework (code.google.com/p/slickspeed/) to test the performance of the frameworks. They tested them on seven browsers and four operating systems. They looked at the overall time it took to run the frameworks on the browsers, and to estimate this they ran each framework five times to find the average time.

3.4 Quality model

Another common way to evaluate a framework is by looking at various static qualities. Multiple quality attributes and values define a quality model that may be used to evaluate and rank multiple frameworks. Quality can refer to several things, but when it comes to evaluating frameworks, the focus is usually centered around how big the framework is, how complex it is, and how easy or difficult it is to maintain. One can look at it by both examining the code in general but also the individual functions. Metrics for examining the size are Lines of Code (LOC), the total amount of statements, the total amount of lines that consist of comments, as well as the percentage of lines that consist of code and the lines that consist of comments. For complexity, metrics such as Cyclomatic Complexity (CC) and Halstead Complexity can be used. With regards to maintainability, Maintainability Index is an example of a metric that can be used [5].

In 1976, the Cyclomatic Complexity metric was created, with the goal of counting the number of potential paths through a system. It was meant to limit the code complexity and determine the number of test cases required for a method [20]. Today this metric is integrated into many metric tools and static analysis tools.

Halstead Complexity emerged in 1977 [21] with the purpose of determining qualities of the software and their connection to each other that could be measured. By knowing the amount of unique operators, the amount of unique operands, the total amount of operators, and the total amount of operands it is possible to calculate the length of a program, its difficulty, effort, the time it takes to write it, the volume, among other measures.

Finally, the Maintainability Index [22] is a metric that is meant to show the maintainability of the code. Its calculation is a factored form comprised by LOC, CC, and Halstead (volume). It exists in various metric tools.

4 Framework Evaluation and Results

This chapter contains an evaluation criteria, the evaluation of the frameworks, and the results.

4.1 Evaluation criteria

The following criteria/model is a suggestion on what to evaluate and how to evaluate it based on the literature study. Anyone who would like to evaluate frameworks can look at this and follow accordingly. Important to note is that it is used to collect data to use for comparison. The criteria lack levels such as one can find in a quality model where the quality is specified.

Question/situation	**Metric**	**Method**	**Example**
Which framework is the fastest in terms of page load/render time/execution time?	Performance [5], [19]	Benchmark the frameworks using a performance test tool.	Lighthouse [23], Chrome DevTools [24], js-framework-benchmark [25]
Which framework is the smallest? Which framework is the largest? How complex is the framework? How maintainable is the framework?	Complexity & Maintainability [5], [20], [21], [22]	Analyze size, complexity and/or maintainability using a test tool.	Microsoft Visual Studio [26]
How easy will it be to learn the framework? Which framework will save the developer time when using it?	Usability [7], [16], [17]	User survey, interview, and/or look at the documentation of each framework.	Online survey, in-person interviews

Which framework can give the developer the highest chance of solving a problem by searching for it online? Which framework will give the employer the largest pool of candidates?	Community [7], [16], [17]	Look at the frequency of posts on Stack Overflow, and/or analyze Github repositories and popularity [7].	N/A
Which framework has the strongest security aspects? What are the possible upgrades for the framework? What does support look like?	Enterprise [16]	Look at licensing, support, upgrade, and/or security.	N/A

Table 4.1: Evaluation criteria.

4.2 Usability

For this project, one of the two metrics that were chosen was usability because of its qualitative aspect. As mentioned in the literature review (chapter 3), usability can refer to the aspects that improve the development process, such as the learning curve, documentation, and syntax. It also means the time it takes to start a project with a framework and the time it takes for the overall usage of a framework. While there are many ways to measure usability of a system (usability test, interviews, use case scenarios), a survey with a usability scale seemed the most fitting considering the time and size of this project. A survey saves the conductor and participants time and allows anonymity in a way most of the other methods do not. Since the survey revolves around usability, the questions were focused on the participants' own perceived experience on usability, such as the learning curve, time management, ease of usage etc. The survey was sent to Kalmar Science park, students at Linnaeus University, as well as other people who the author knows and has experience in JavaScript.

There were 78 participants, 33 of them were familiar with Angular, 56 of them were familiar with React, and 29 of them were familiar with Vue. Below are a few of the questions and answers, the full survey with results can be found in Appendix 1.

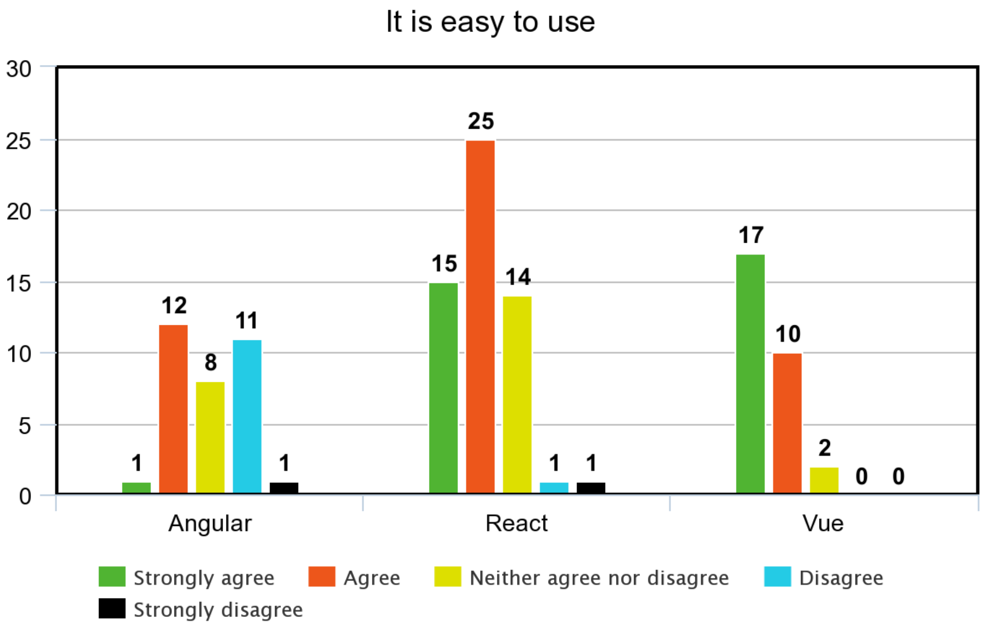


Figure 4.1: Bar chart displaying the answers to the statement “it is easy to use”.

The first statement aimed to gain a better understanding of how easy the frameworks are to use since it is one of the most important aspects when it comes to usability. The results can be seen in Figure 4.1.

Another important factor for usability is how fast it takes to learn a new framework. The results are shown in Figure 4.2.

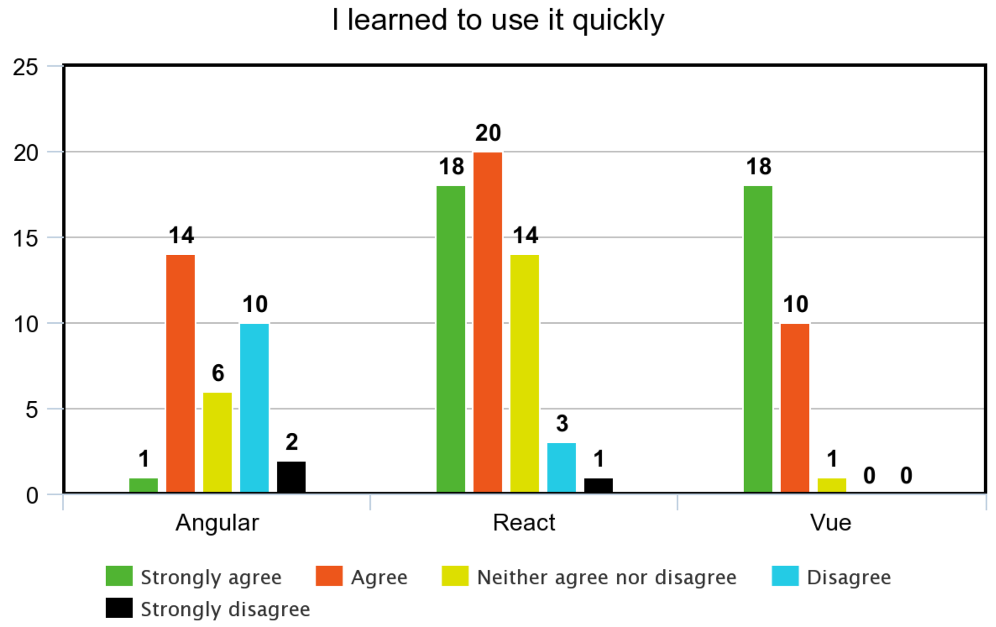


Figure 4.2: Bar chart displaying the answers to the statement “I learned to use it quickly”.

When assessing the usability of a framework, the productivity of using it is also worth investigating. Vue and React both received a good score from this statement, while Angular got a much lower result, as seen in Figure 4.3.

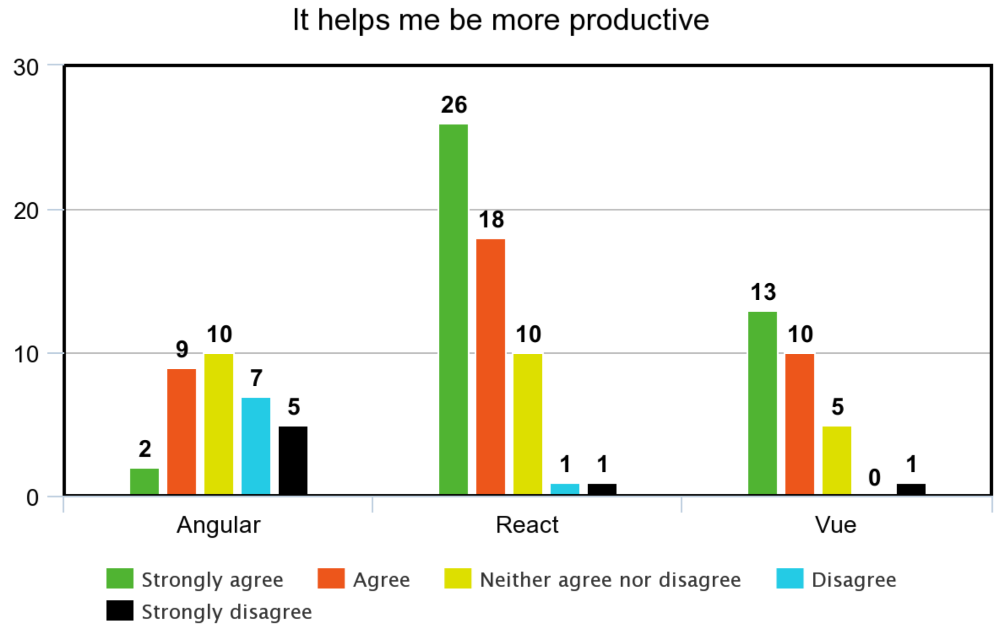


Figure 4.3: Bar chart displaying the answers to the statement “it helps me be more

productive”.

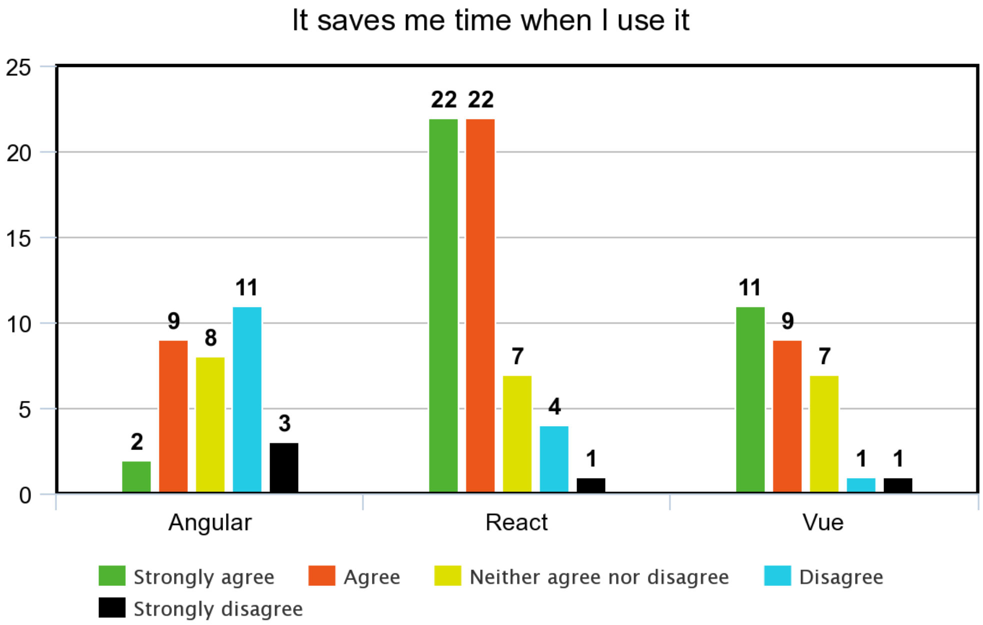


Figure 4.4: Bar chart displaying the answers to the statement “it saves me time when I use it”.

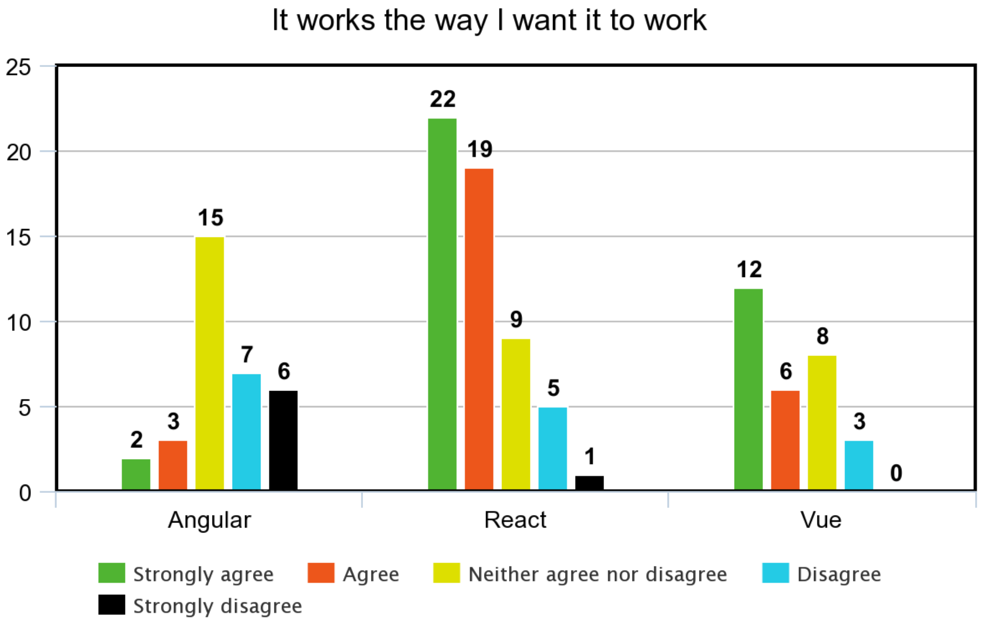


Figure 4.5: Bar chart displaying the answers to the statement “it works the way I want it to work”.

Lastly, the participants were asked which framework they thought had the best performance and speed and the results are presented below.

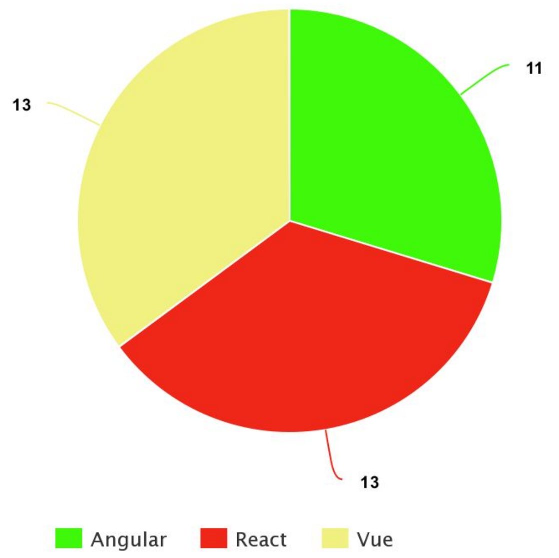


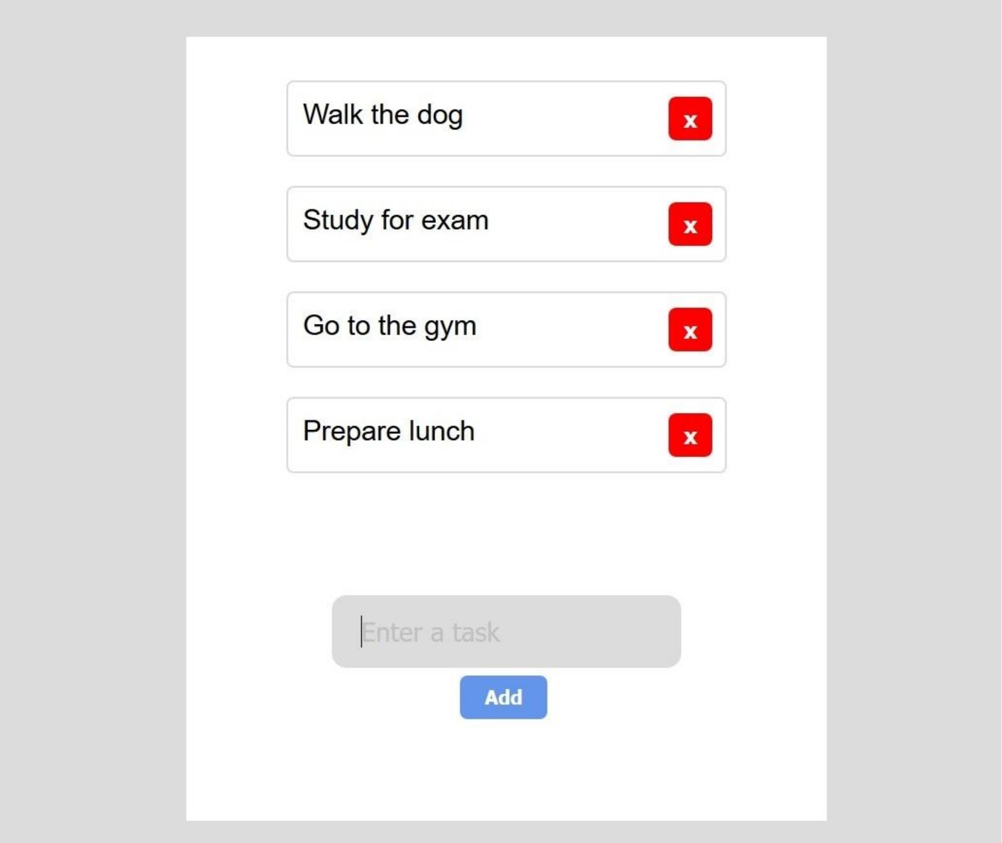
Figure: 4.6: Pie chart of the answer to the question "Which of the following JavaScript framework do you think is the best in terms of performance and speed?".

4.3 Performance

While the user survey aimed to get more qualitative results, the performance test was meant to offer an objective way of evaluating frameworks, which is why this metric was chosen. Moreover, benchmarking, as mentioned in Chapter 3, is an important and common way of evaluating frameworks.

4.3.1 Setup

The experiment was conducted by creating and testing three applications made up of each of the frameworks. The application was a simple To-Do application with basic CRUD functionality (Create, Read, Update, Delete). It was possible to add(create), read, edit(update), and delete tasks. Figure 4.7 gives an example of what they look like. The source code to the applications can be found at https://github.com/Josefin4/angular-todo-app, https://github.com/Josefin4/react-todo-app, and https://github.com/Josefin4/vue-todo-app.



Walk the dog

Study for exam

Go to the gym

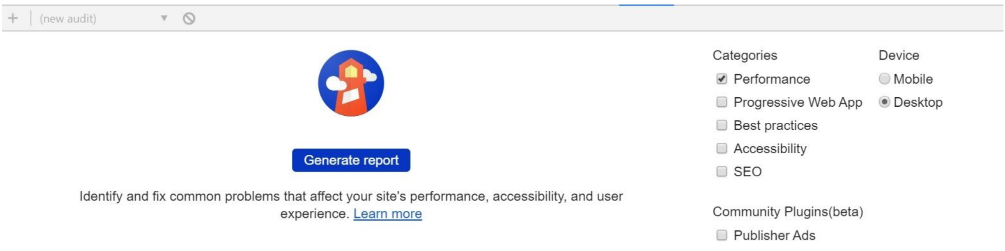
Prepare lunch

Figure 4.7: Example of the application.

The operating system being used for this experiment was Windows 10 x64 version 1903. The applications were tested in Google Chrome (version 81.0.4044.138), since Chrome is by far the most popular browser [27], and it also has built-in tools that can be used to test performance. In this project, the tool Lighthouse [23] was used as a Chrome extension in Chrome DevTools. The reason Lighthouse was chosen was that it offers a different range of factors that can be assessed when testing the performance. When using Lighthouse, it is possible to run it in Chrome DevTools, from the command line, or as a Node module. For this experiment, it was running in Chrome DevTools. The way Lighthouse works is that it audits the application and then generates a report. It is possible to choose different factors that should be assessed which are shown in Figure 4.8. Once the report is finished, the results are displayed. An example of what this can look like can be seen in Figure 4.9.

There are six metrics that are measured when evaluating the performance from the Lighthouse audit;

- First contentful paint [28], which measures the total rendering time of the initial DOM content by the browser once a user visits the page. The content that counts as DOM content is pictures, elements that are not white, and lastly SVGs.
- First meaningful paint [29], measures the time that it takes for the first-page content to be displayed for a user.
- Speed index [30], measures the time it takes for the content to be visible when the page is loading. It works in which Lighthouse grabs a video of the page that is loading in the browser, and then estimates the visible development between the frames.
- First CPU idle [31], measures the time it takes until a page first begins to be interactive.
- Time to interactive [32], measures the time it takes until a page is completely interactive.
- Max potential first input delay [33], measures the maximum first input delay that could possibly happen to a user who visits the page. It estimates the time it takes from the moment the user starts interacting with the page until the browser can react to it.



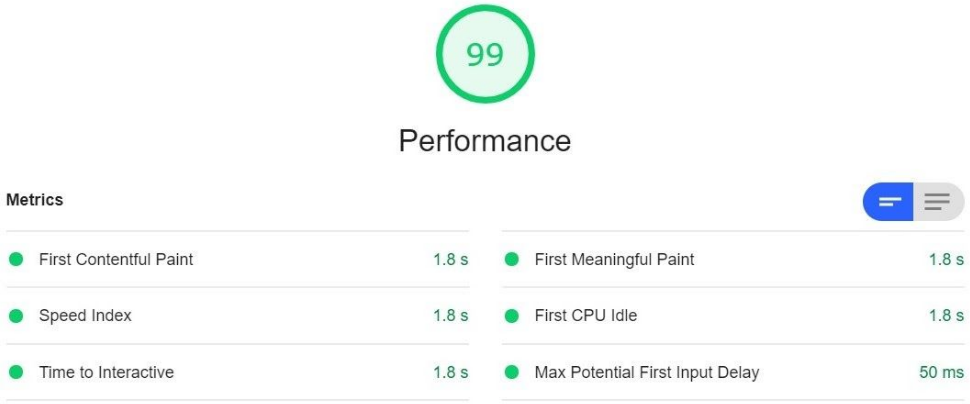
Identify and fix common problems that affect your site's performance, accessibility, and user experience. Learn more

Categories
☑ Performance
☐ Progressive Web App
☐ Best practices
☐ Accessibility
☐ SEO

Device
○ Mobile
● Desktop

Community Plugins(beta)
☐ Publisher Ads

Figure 4.8: Example of what aspects can be assessed from a Lighthouse audit.



99

Performance

Metrics

● First Contentful Paint 1.8 s

● First Meaningful Paint 1.8 s

● Speed Index 1.8 s

● First CPU Idle 1.8 s

● Time to Interactive 1.8 s

● Max Potential First Input Delay 50 ms

Figure 4.9: Example of the results from a Lighthouse audit report.

4.3.2 Experiment

The Lighthouse tool was used to evaluate Performance, with the device Desktop being chosen. For each application/framework, the audit was run 5 times in total. The results are shown in Table 4.2 as the average time in seconds, except for Max Potential FID which is shown in milliseconds, and finally the overall performance score. The maximum (and best) performance score is 100.

	Angular	React	Vue
First Contentful Paint	3.1	1.5	1.8
First Meaningful Paint	3.1	1.5	1.8
Speed Index	1.9	1.4	1.8
First CPU Idle	3.1	1.5	1.8
Time To Interactive	3.1	1.5	1.8
Max Potential FID	36	26	30
Overall Performance Score	90	100	98

Table 4.2: Results from the Lighthouse audit.

5 Analysis and Discussion

This chapter covers the analysis of the results from the previous chapter, as well as a discussion.

5.1 Evaluating Frameworks

The results from the literature review show that there are many considerations to keep in mind when evaluating and ultimately choosing a framework. Furthermore, it shows that there are several different factors to assess and some of them are more common than others such as the performance and quality of the frameworks, and the usability and community. This suggests that evaluating frameworks is not a straightforward process as it largely depends on which factors matter more when picking the framework. If the overall execution time for a framework is the most important factor, then the performance of the framework is the most essential aspect to focus on and evaluate. Whereas if one is looking to find the easiest framework to learn then one should focus on the usability. The literature review results furthermore show that there are different ways to evaluate different aspects such as conducting user surveys, benchmarking the frameworks, and analyzing the complexity among other things. Moreover, one can assume that the stakeholders have a large impact on which factors and metrics will be evaluated. A developer may care more about the usability and performance of a framework than the price or license. Even factors all stakeholders have an interest in may look different, for example, the community of a framework is important to both a developer and an employer. The developer may want to use a framework that has a large online community for searching for solutions in their work, while the employer may want to pick a framework that has a large number of developers they can hire. Picking a framework with a small community would mean fewer candidates when trying to hire a new developer. Additionally, based on the different considerations and metrics to keep in mind, it also likely to assess trade-offs when picking a framework. It may not be possible to evaluate every single metric with a framework nor find a framework that scores the best in all tests/experiments. In such a case one would have to choose which factors matter the most and then evaluate based on them.

From the literature review, an evaluation criteria was created in an attempt to offer an overview of how a framework evaluation can be conducted. Depending on the problem or situation the conductor is faced with, the criteria can give them an answer to which metric and method should be used. The criteria itself shows that, while there are many considerations to keep in mind and different viewpoints that influence the choices, there is a way to make the evaluation process more standardized and certain methods work best (or only) with certain metrics. For example, performance would be better to evaluate by testing the frameworks rather than conducting a user survey since a benchmark can give a more precise and objective answer instead of asking people.

5.2 Evaluation techniques

The evaluation was done based on the evaluation criteria and was made up of two methods; user survey and experiment.

5.2.1 User survey

The user survey attempted to answer the question regarding usability and was meant to further demonstrate how one can evaluate frameworks as well as providing a better understanding of the usability of the frameworks chosen for this project.

The majority of the 78 participants in the survey were familiar with React (56 participants), followed by Angular (33 participants), and finally Vue (29 participants). No framework got the best results in all every question, but a few received an overall higher score than others. When it came to the statement "it is easy to use", Vue got the best results as 93% of the 29 participants who were familiar with Vue either strongly agreed or agreed. React received 71% from its participants and Angular landed at 42%. This was the trend for all five questions, except for the statement "it saves me time when I use it", where React got the best result (79% strongly agreed or agreed), while Vue came second with 69%. React got its lowest score from the statement "I learned to use it quickly" as it got only 68%. That was also Vue's highest score at 97%. When it came to the productivity of the frameworks, Vue and React received the same score (79%) while Angular got 33%. With the last statement "it works the way I want it to work", Angular received its lowest score with 15%, while React got 73% and Vue 62%. Based on these results, one can assume that the participants were the happiest with Vue and React, and enjoyed working with Angular the least. At the same time, one can also assume that Vue and React have different strengths and weaknesses when it comes to usability. Based on the results, Vue is the best framework when it comes to how easy the frameworks are to use as well as its learning curve. From this survey, Vue would be the fastest framework to learn. The frameworks that would help the developer be more productive would be either Vue or React. When it comes to the framework that saves the developer the most time, it would be React. When it comes to the developer's experience of which framework works most in the way they want it to work, React would be the choice. This relates to the findings from the literature review: choosing the best framework based on these results would require the person to assess which factors matter the most as no framework received the highest score in each question. If learnability is important then the survey results suggest that Vue would be the most fitting framework, while React would be better for saving the developer time. Overall Vue scored best even though it has certain weaknesses which as shown in the statement "it works the way I want it to work".

It is worth mentioning that the sample size for the survey was limited, which means that the results might not be entirely trustworthy or they might have looked different with a larger group of participants. Many of the participants were students at the same university and same country. This of course could affect their answers, depending on different courses that may have used any of these frameworks, or professors and teachers who hold strong opinions of the frameworks or of frameworks in general. Moreover, programmers in general may have other opinions of these particular frameworks compared to this group of participants, especially in countries or cities where for example Angular is the most popular framework.

5.2.1 Experiment

The experiment was centered around the performance of the frameworks and had the same purpose as the user survey; demonstrating how an evaluation process could look like when following the evaluation criteria.

While Vue received the best overall score in the user survey, the framework that performed best in this experiment was React. This shows how two different metrics (usability and performance) can give different results. While one might offer more usability, the other has a better performance. For this experiment, React was able to get the highest scores in all 6 areas (First Contentful Paint, First Meaningful Paint, Speed Index, First CPU Idle, Time to Interactive, Max Potential FID), multiple times and was consistently fast. However, Vue was not far behind in these same areas. This suggests the difference in performance might not be very noticeable at all between these two frameworks. Lastly, Angular got slowest response in all areas. The conclusion that can be drawn from this experiment is that React and Vue are faster while Angular is the slowest. This matches the participants' from the user survey answer to the question "which of the following JavaScript framework do you think is the best in terms of performance and speed?", as 13 participants guessed Vue and 13 participants guessed React. 11 answered Angular. This could mean that developers may be able to perceive the performance of a framework in a way that is fairly accurate, since Vue and React both received a good score. However, a fairly large part of the participants expected Angular to be the best when it comes to performance and speed. The results from the experiment do not reflect this assumption. This suggests that testing frameworks via benchmarks or other, more objective, methods, is more suitable than making an estimated guess or asking developers.

5.3 Findings

The purpose of this project has been to provide a greater understanding in how to evaluate frameworks and which metrics should be used when evaluating them. The literature review looked at the previous related research that has been done in the past in order to find certain metrics and methods that are more common in order to create the evaluation criteria. The findings from the literature review suggest that there are various factors to consider when choosing a framework and which metrics should be used for what factor (e.g. benchmarking a framework is used for evaluating the performance). The user survey further answers the question "how to evaluate frameworks" as it shows one way of evaluating them by focusing on the metric "usability". From the user survey results, one can choose the framework that receives the best results overall or the best results in the questions that are most important for whoever reads the results. Likewise, the experiment also demonstrates how to evaluate frameworks by focusing on the metric "performance" via benchmarking the frameworks.

As mentioned in the previous sub-chapters, the results from each method suggest that one framework can receive a great result in one metric or method and a not so great result in another. Additionally, one framework can receive the best result in one metric while a different framework gets the highest score in another metric. The question then becomes which metric is the most important for the stakeholder, which means that trade-offs might be inevitable in many cases. Lastly, the results from the user survey and the experiment show that different methods will give different results. This further shows how different problems/questions need different methods and metrics.

The findings from this project are similar to the related research that has been conducted in the past, such as benchmarking the performance of a framework that has been proven to be both important and common when evaluating frameworks.

6 Conclusion and Future Work

This project has shown how to evaluate frameworks, it has found certain considerations to keep in mind, which metrics that are commonly used to evaluate frameworks and why they are important, and what metrics should be picked for a specific problem/question (See Chapter 3.2, 3.3, 3.4, and Chapter 4.1). Furthermore, the project has demonstrated an implementation of how to evaluate frameworks based on this knowledge and the metrics that have been evaluated are usability and performance. The conclusion of the findings are there are many different ways to evaluate a framework and there are various factors to evaluate, but it is possible to find methods and metrics to evaluate based on previous research and specifically which aspects of a framework that matter the most to developers and other professionals in the field. Moreover, the conclusion that can be drawn from the evaluation conducted in this project is that trade-offs may be needed depending on what the results are. If one framework receives a better result when it comes to performance but another framework receives the best result with usability, then trade-offs could be necessary. Which aspect is the most important for the stakeholder and how big is this difference in the results?

The findings from this project can be used for professionals in the industry who are looking to evaluate frameworks and are in need of some guidance as to how to conduct an evaluation. Additionally, the results can lead to further research with frameworks and how to evaluate them and best practices. The results can be used for both JavaScript frameworks as well as frameworks of other programming languages since metrics such as performance and usability are of importance in other languages as well.

In order to get even better results, more metrics could have been evaluated to get a bigger overview and more than three frameworks. The survey could have been expanded to ask questions regarding the developers' reasons for their choice of framework(s).

This project has provided a better understanding of how to evaluate frameworks but there is still research needed for how to pick a framework. While this project discusses the different considerations and the difficulty in finding one framework that has it all, there might be a method or methods that could give a more straightforward answer. As for benchmarking the performance, this project is testing a fairly simple To-Do app, but future research could test a larger application to emulate a real-world application to get an even better result. Other browsers or different operating systems could also be used when testing the performance of the frameworks.

References

[1] M. Fayad, and D. Schmidt. (1997) “Object-Oriented Application Frameworks”. Communications of the ACM. 40. 10.1145/262793.262798.

[2] Web.archive.org. (2015). *Web application framework - DocForge*. [online] Available: https://web.archive.org/web/20150723163302/http://docforge.com/wiki/Web_application_framework

[3] Mozilla. (2019) “Server-side web frameworks” [Online] Available: https://developer.mozilla.org/en-US/docs/Learn/Server-side/First_steps/Web_frameworks

[4] Mozilla. (2019) “About JavaScript.” [Online] Available: https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript

[5] A. Gizas, S. Christodoulou, and T. Papatheodorou. (2012) “Comparative Evaluation of JavaScript Frameworks.” WWW '12 Companion Proceedings of the 21st International Conference on World Wide Web, pp. 513-514.

[6] P. Ratanaworabhan, B. Livshits, and B. G. Zorn. (2010) “JSMeter: Comparing the Behavior of JavaScript Benchmarks with Real Web Applications”. WebApps '10: Proceedings of the 2010 USENIX conference on Web application development. [Online] Available: https://www.usenix.org/legacy/event/webapps10/tech/full_papers/Ratanaworabhan.pdf

[7] A, Pano, D. Graziotin, and P. Abrahamsson.(2016) “What leads developers towards the choice of a JavaScript framework?” [Online] Available: https://www.researchgate.net/profile/Pekka_Abrahamsson/publication/303221742_What_leads_developers_towards_the_choice_of_a_JavaScript_framework/links/58834fa7aca272b7b443d49a/What-leads-developers-towards-the-choice-of-a-JavaScript-framework.pdf

[8] M. Kamaruzzaman. (2020) “Top 10 JavaScript Frameworks to Learn in 2020” Towardsdatascience.com [Online] https://towardsdatascience.com/top-10-javascript-frameworks-to-learn-in-2020-a0b83ed3211b

[9] A. Goel. (2019) “Why should you learn Angular?” Hackr.io. [Online] Available: https://hackr.io/blog/why-should-you-learn-angular-in-2019

[10] Wikipedia. (2015) “React (web framework).” [Online] Available: https://en.wikipedia.org/wiki/React_(web_framework)

[11] Vuejs.org. (2014) “Introduction - Vuejs.” [Online] Available: https://vuejs.org/v2/guide/

[12] Stackshare.io. (2018) “React” [Online] Available at: https://stackshare.io/react

[13] Stackshare.io. (2018) “Vue-js” [Online] Available at: https://stackshare.io/vue-js

[14] Madewithangular.com. (2020) “Made with Angular” [Online] Available at: https://www.madewithangular.com/

[15] Vuejs.org. (2015) “Comparison with Other Frameworks” [Online] Available at: https://vuejs.org/v2/guide/comparison.html

[16] B. Satrom. (2018) “Choosing the Right JavaScript Framework for Your Next Web Application” [Online] Available: https://www.telerik.com/docs/default-source/whitepapers/telerik-com/choose-the-right-javascript-framework-for-your-next-web-application_whitepaper.pdf

[17] J. Duvander, and O. Romhagen. (2019) “What affects the choice of a JavaScript framework: Interview with developers” Jönköping University, oai:DiVA.org:hj-46268.

[18] M. Utting and B. Legeard, *Practical Model-Based Testing -- A Tools Approach*. Elsevier Inc, 2007.

[19] S. Sim, S. Easterbrook, and R. C. Holt, “Using benchmarking to advance research: A challenge to software engineering”, in *Proceedings of the 25th International Conference on Software Engineering*, 2003, pp. 74–83.

[20] T.J. McCabe, “A Complexity Measure” *IEEE Transactions on Software Engineering,* vol. 2, pp. 308-320, 1976.

[21] M.H. Halstead, *Elements of Software Science.* Elsevier Science Inc, 1977.

[22] P. Oman, and J. Hagemeister, “Metrics for assessing a software system’s maintainability, in *Proceedings Conference on Software Maintenance,* 1992, pp. 337-344.

[23] Developers.google.com. (2020) “Lighthouse” [Online] Available at: https://developers.google.com/web/tools/lighthouse

[24] Developers.google.com. (2019) “Chrome DevTools” [Online] Available at: https://developers.google.com/web/tools/chrome-devtools

[25] Github.com. (2017) “Js-framework-benchmark” [Online] Available at:https://github.com/krausest/js-framework-benchmark

[26] Docs.microsoft.com. (2018) “Code metrics values” [Online] Available at: https://docs.microsoft.com/en-us/visualstudio/code-quality/code-metrics-values?view=vs-2019

[27] Gs.statcounter.com. (2020) “Browser Market Share Worldwide” [Online] Available at: https://gs.statcounter.com/browser-market-share

[28] Web.dev. (2019) “First Contentful Paint” [Online] Available at: https://web.dev/first-contentful-paint/

[29] Web.dev. (2019) “First Meaningful Paint” [Online] Available at: https://web.dev/first-meaningful-paint/

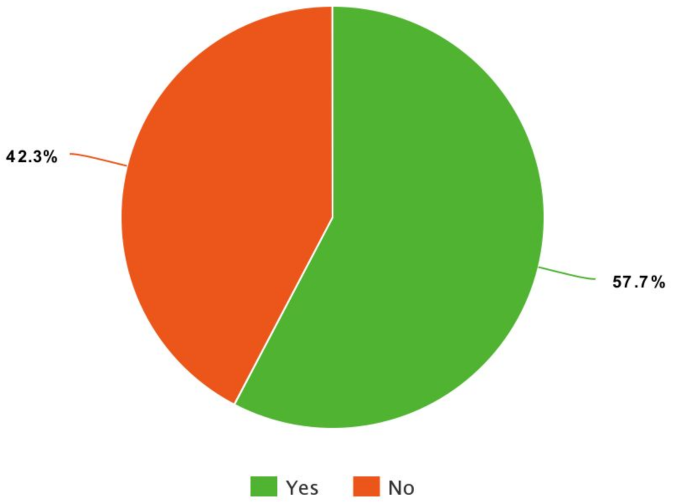
[30] Web.dev. (2019) “Speed Index” [Online] Available at: https://web.dev/speed-index/

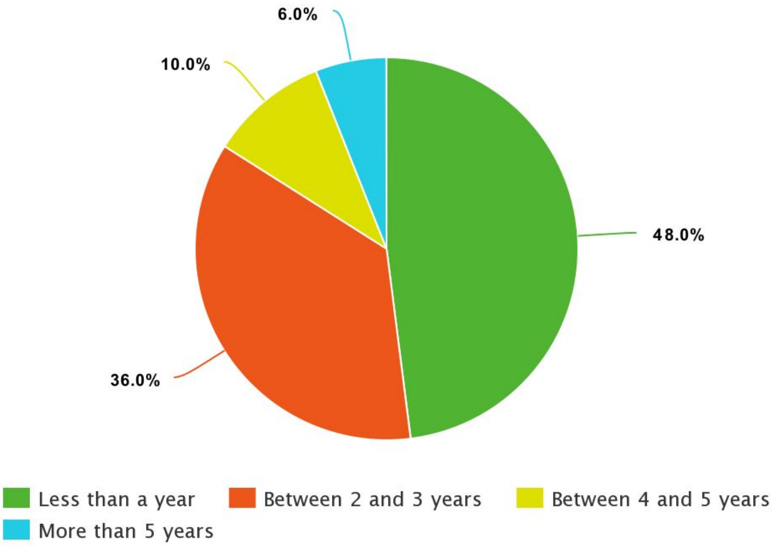
[31] Web.dev. (2019) “First CPU Idle” [Online] Available at: https://web.dev/first-cpu-idle/

[32] Web.dev. (2019) “Time To Interactive” [Online] Available at: https://web.dev/interactive/

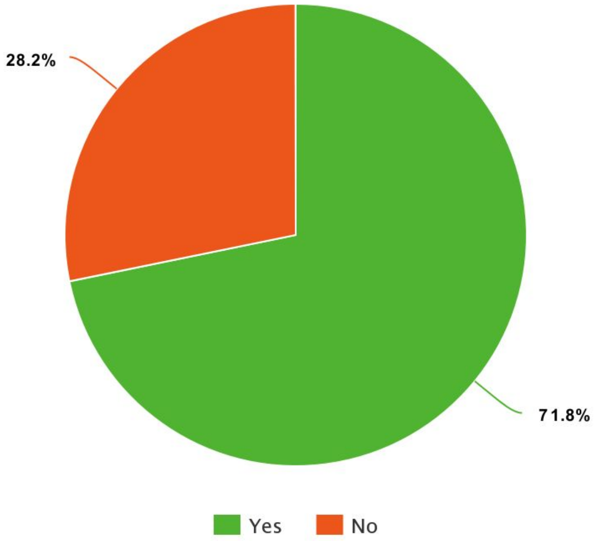
[33] Web.dev. (2019) “Max Potential First Input Delay” [Online] Available at: https://web.dev/lighthouse-max-potential-fid/

A Appendix 1

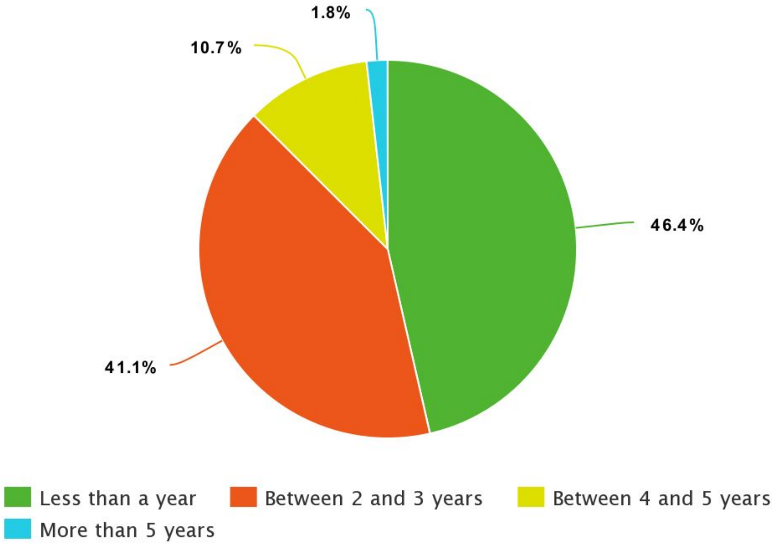

Are you familiar with Angular (Note that the question is about Angular2+ and not AngularJS)


For how many years have you been using Angular?

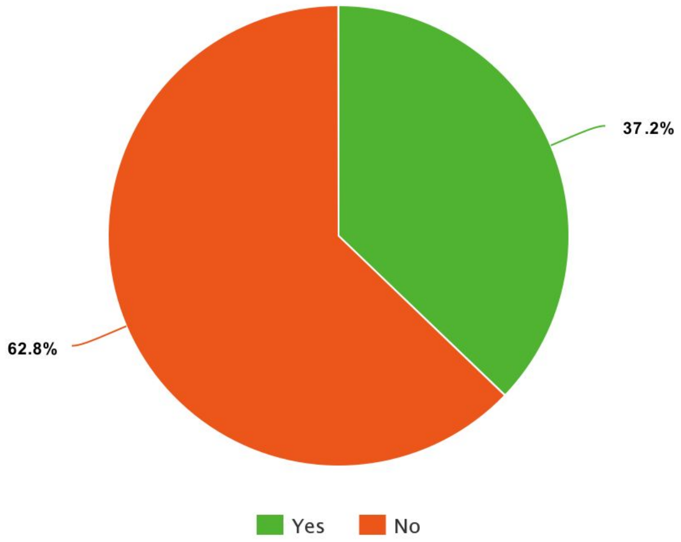
Are you familiar with React.js?



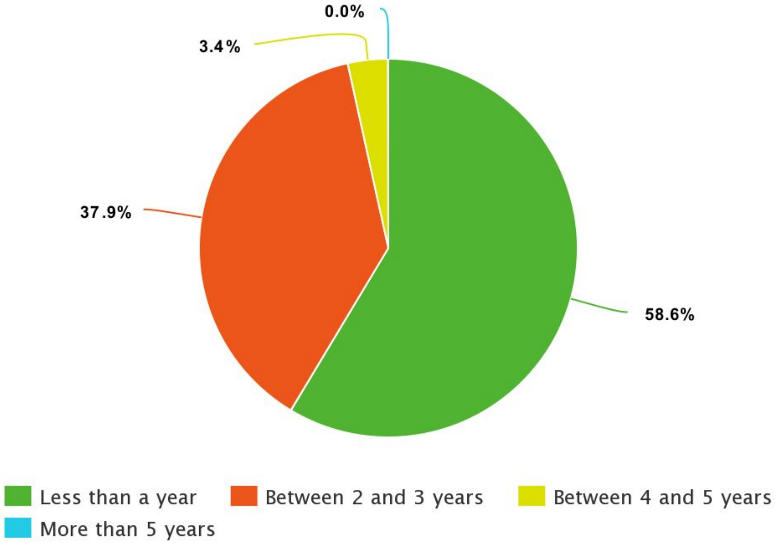
For how many years have you been using React.js?

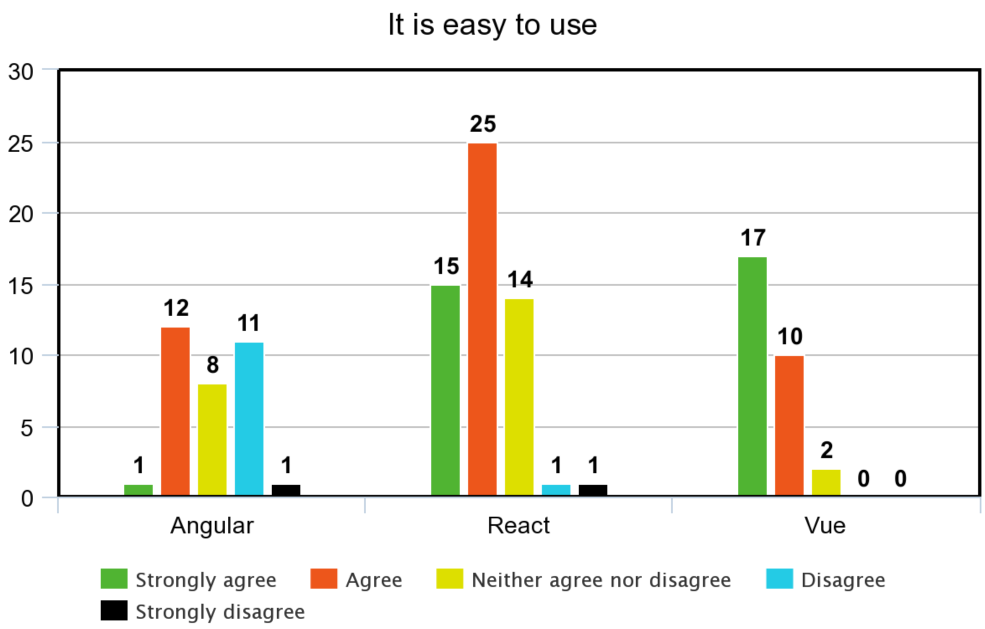


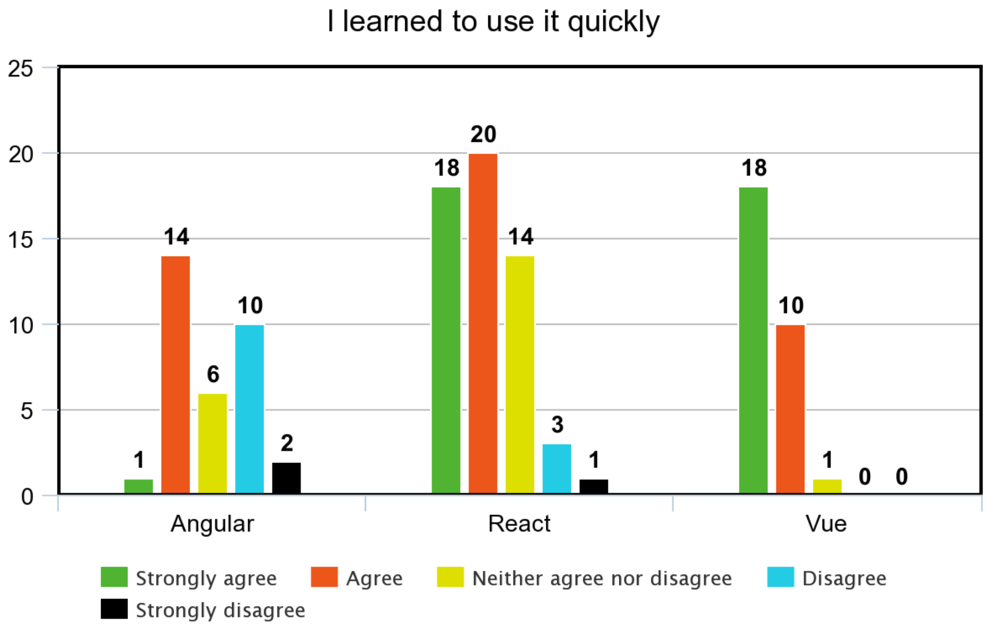
Are you familiar with Vue.js?

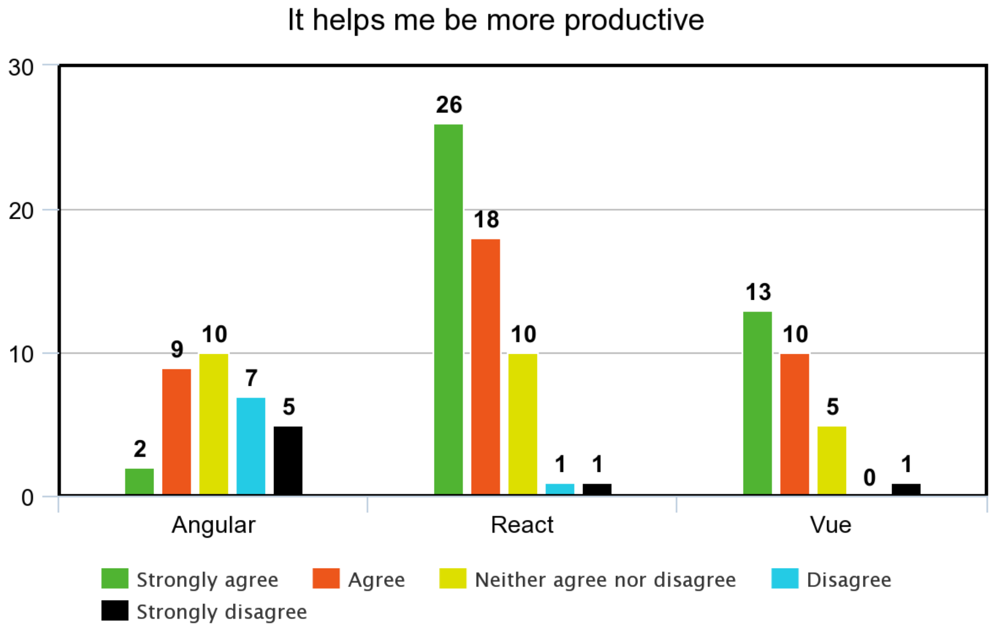


For how many years have you been using Vue.js?

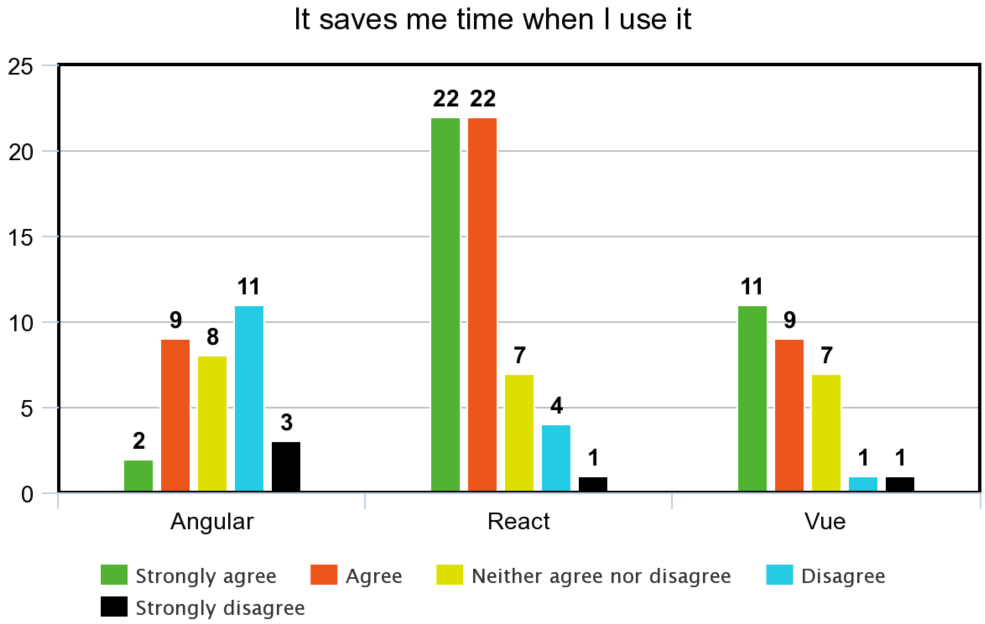



It is easy to use

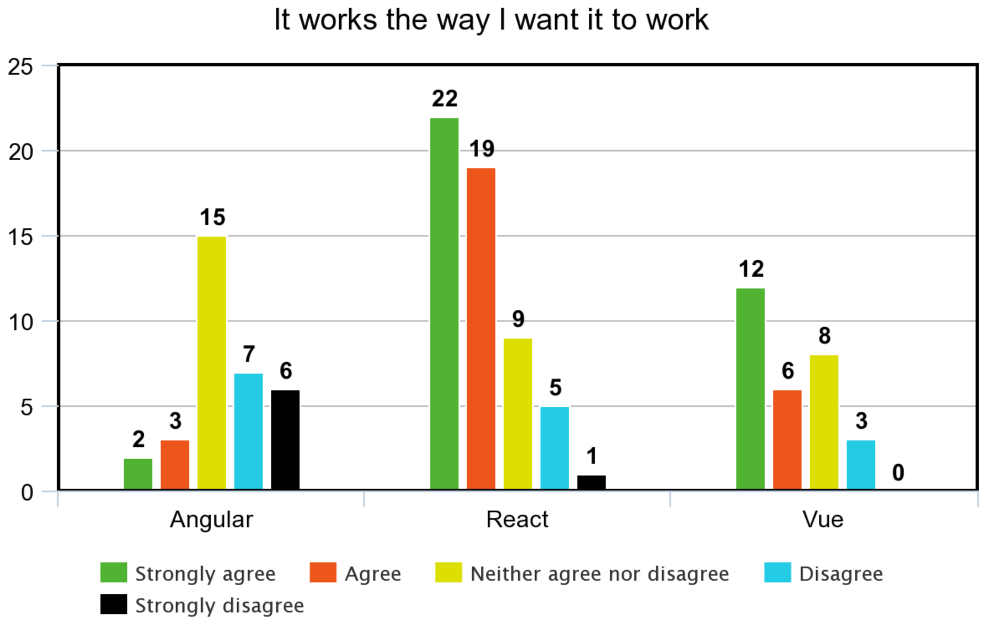

I learned to use it quickly



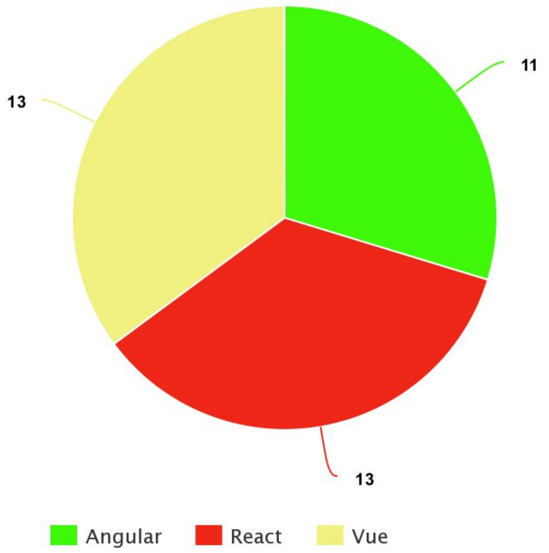
It helps me be more productive



It saves me time when I use it



Which of the following JavaScript framework do you think is the best in terms of performance and speed?


Angular
React
Vue