DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING, FIRST CYCLE, 15 CREDITS
STOCKHOLM, SWEDEN 2022

An Evaluation of Testing Frameworks for Beginners in JavaScript Programming

An evaluation of testing frameworks with beginners in mind

GEORGEK AROUSH

KTH ROYAL INSTITUTE OF TECHNOLOGY
ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

An Evaluation of Testing Frameworks for Beginners in JavaScript Programming

School of Electrical Engineering and Computer Science

GEORGEK AROUSH

Degree Programme in Computer Engineering
Date: September 5, 2022

Supervisor: Anders Sjögren and Anne Håkansson

Examiner: Leif Lindbäck

Swedish title: En utvärdering av testramverk för nybörjare i JavaScript
Swedish subtitle: Skolan för elektroteknik och datavetenskap

Abstract

Software testing is an essential part of any development, ensuring the validity and verification of projects. As the usage and footprint of JavaScript expand, new testing frameworks in its community have made statements about being the best overall solution using minimal intervention from developers. The statements from these frameworks about being the greatest can make it difficult for JavaScript beginners to pick a framework that could affect current and future projects. By comparing different types of frameworks and establishing a guideline for others to do the same, it becomes easier for beginners and others to choose a framework according to their own required needs. The overall method uses Mario Bunge's scientific method via stages, which helps validate the thesis as scientific. Research, empirical data from a qualitative, and objective data from a survey decide the criteria, their priority (to determine their impact and hierarchy), what frameworks to include, and how to compare them. The frameworks Jest, AVA, and Node TAP are compared based on the main criteria of simplicity, documentation, features, and their sub-criteria. Evaluating the frameworks and ranking their performance in each criterion was done through an experiment conducted on a pre-made website without any testing included. The analytic hierarchy process is the primary method used to combine the information gathered and output a result. It makes it possible to create a priority hierarchy for each criterion and subsequently makes it possible to evaluate the choices available on their fulfillment of those criteria. One of these choices will eventually be an overall more suitable fit as the optimal framework for the research question. Combining the survey and experiment data into the analytic hierarchy process revealed that Jest fit the previous criteria better than AVA and Node TAP because of Jest's better learning curve and Stack overflow presence. AVA was just behind in those areas, while Node TAP had a poor fit for all sub-criteria compared to the other two. AVA's almost similar evaluation to Jest shows how the open-source community and small development teams can keep up with solutions from big corporations.

Keywords

Mario Bunge's scientific method, Analytic hierarchy process, Jest, Ava, Node TAP, and JavaScript

Sammanfattning

Programvarutestning är en viktig del av all utveckling, för att säkerställa giltigheten och verifieringen av projekt. Tack vare JavaScripts expandering och användning, så har nya testramverk dykt upp som anser sig vara den bästa lösningen för utvecklare. Dessa påståenden kan göra det svårt för nybörjare inom JavaScript-utveckling att bestämma sig för vilket ramverk de borde använda, vilket kan påverka deras arbete och framtida projekt. Genom att jämföra dessa ramverk och etablera riktlinjer för andra nybörjare, blir det simpelt för olika demografiska grupper att välja rätt testramverk enligt deras egna åsikter. Den övergripande metoden använder Mario Bunges vetenskapliga metod, vilken använder flera steg för att omvandla hypotesen inom arbetet till en vetenskaplig rapport. Forskning och empirisk information från kvalitativa undersökningar, samt objektiva insamlingar från undersökningar, har använts för att bestämma enligt vilka kriterier dessa ramverk ska jämföras, vilken prioritering dessa kriterier har för nybörjare, vilka testramverk som ska användas och hur ramverken ska jämföras. Testramverken Jest, AVA och Node TAP har jämförts baserat på huvudkriterierna enkelhet, dokumentation och funktionalitet, dessa kriterier innehåller även underkriterier. Evalueringen av dessa ramverk och deras grad av prestanda inom dessa kriterier gjordes genom experimentellt utförande och användning inom en förhandsgjord hemsida utan någon form av testning inkluderad. Den analytiska hierarkiska processen var den primära metoden som användes för att kombinera den insamlade informationen till ett slutgiltigt resultat. Detta för att en prioriteringshierarki kan skapas för all kriterier, och gör det även möjligt att evaluera all ramverk inom dessa kriterier. Ett av dessa ramverk kommer eventuellt beräknas som det bästa alternativet, och på så sätt hjälpa besvara huvudfrågan. Kombinationen av resultaten från undersökningen och experimenten gav att Jest passar bäst till nybörjare, baserat på kriterierna och deras prioriteringsrang, detta tack vare att Jest har bättre inlärningskurva och Stack Overflow-närvaro jämfört med AVA och Node TAP. AVA ligger precis efter inom dessa kriterier, medan Node TAP har betydligt sämre prestanda inom alla kriterier. AVA:s närliggande kapacitet till Jest bevisar att mindre grupper av utvecklare kan komma upp med bra lösningar precis som större företag.

Nyckelord

Mario Bunge’s vetenskapliga metod, Analytisk Hierarkisk Process, Jest, Ava, Node TAP och JavaScript

Acknowledgments

I would like to thank Anders Sjögren for his recommendation of Bunge's scientific method and comprehensive guidance throughout this paper.

Stockholm, September 2022
Georgek Aroush

Contents

List of Figures

List of Tables

Listings

List of acronyms and abbreviations

AHP	Analytic Hierarchy Process
JS	JavaScript
MCDM	Multi-Criteria Decision-Making

Chapter 1

Introduction

Testing is a big part of the development of any piece of software. Testing validates that the software meets the specifications required and that other unexpected behavior, known as bugs, does not occur [1]. As project complexity grows with the ever-bigger demand from the end-users, so will the testing during production [2]. Complexity can make it difficult and expensive to maintain and evaluate the testing code for each component [3]. A solution for this is frameworks. Frameworks use automation to ensure developers a robust and enjoyable environment, which helps with faster releases, better quality end products, and minimalistic resource waste [4]. Anyone with enough resources, corporations like Facebook, Google, or even communities of independent programmers on GitHub can and have created these frameworks [5, 6].

1.1 Background

JavaScript has for eight years in a row been the most in-demand language according to stack overflow surveys [7] and has been called the most asked for language by Devskiller [8], yet research in testing frameworks in the most demanded language is lacking. All currently available comparisons for these testing frameworks are shallow and use the framework's creators' own bullet points [9, 10, 11]. When considering that beginners in JavaScript or testing in JavaScript are most likely the demographic for these comparisons and will therefore be the user base concentrates on for this thesis. An organization such as a university could use the conclusion to teach new students what framework to consider when testing JavaScript.

1.2 Problem

There are a lot of testing frameworks out there, and many claims to be the better alternative for testing in JavaScript. Examples of these frameworks are, Jest [12], Mocha [13], Jasmine [14], Chai [15] and AVA[16]. New users will likely pick a popular framework and use that for the remainder of their project without considering that other tools might be better. This raises questions on how one could evaluate testing frameworks for beginners and how an environment might need to be for better results. The problem simplified becomes the research or main question: "Finding an optimal testing framework for beginners in JavaScript programming".

The research question is too broad and has therefore been divided into smaller sub-questions using the divide and conquer method and will answer the question while also limiting the discussion area. The following sub-questions have been identified as questions relevant to give a satisfactory answer for the research question (RQ).

- (SQ1) What frameworks should be compared?
- (SQ2) What criteria do beginners use for evaluating frameworks?

1.3 Purpose

The purpose of the thesis is to present an investigation of different testing frameworks and rank them based on what beginners in JavaScript consider to be crucial for a good experience. The investigation aims at making it easier for any programmer to have an objective way to assess frameworks by listing the user's needed criteria.

1.4 Goals

The goal is to determine if there is a way to differentiate between frameworks and if one is more suitable for beginners than others. Finding testing frameworks and determining what priorities beginners have are all questions needed to be answered to reach the goal. The goal will accomplish this by using qualitative research and implementing small experiments that use hands-on testing with a web application for each framework.

1.5 Methods

There are multiple methods when collecting information. The majority fall under qualitative and quantitative categories [17]. Qualitative methods are about interpreting words or descriptions from texts or having open-ended questionnaires [17]. Quantitative methods are activities in the form of observations, experiments, and questionnaires with close-ended answers [17].

These sub-questions(SQ) will be answered separately but will answer the main research problem in a coherent setting. Qualitative data research was conducted first to answer the two sub-questions identified and was used to establish a conclusion on the findings.

Further research in the form of quantitative was needed to answer the second sub-question and the research question, given that information gathered from the qualitative was limited in that area. A questionnaire or survey was made based on the reasoning that beginners should know what they want in testing frameworks.

1.6 Delimitations

The implementation of the experiment will use a finished project that uses web technologies without any testing done, since the focus of the thesis is to evaluate testing frameworks and not develop a platform to use these testing frameworks on. The finished project is from GitHub and will be used in the implementation of the experiment, meaning that the experiment conducted is to implement these testing frameworks on existing projects. The form of testing will be limited to Unit testing since it is the easiest to grasp, and is widely adopted and documented, making it suitable for beginners.

1.7 Structure of the thesis

The thesis will have this structure:

- Chapter 2 will contain a deeper background about the technologies that will be used and compared.

- Chapter 3 will discuss the development and the methods used, how comparisons are made and why.
- Chapter 4 explains the different implementation made for the comparison.
- Chapter 5 contains the results and analysis.
- Chapter 6 discussion about the result.
- Chapter 7 conclusions and future work.

Chapter 2

Software testing and Testing Frameworks

This chapter presents background information about different topics, testing frameworks, and methods. Section 2.1 will be about software testing, its relevance, and how it relates to beginners in JavaScript. Section 2.1.1 will add to software testing and discusses development strategies to improve workflow and testing scale. 2.2 is about the to-be-evaluated frameworks chosen, evaluated based on their overall match with criteria deemed as requirements for good user experience. The next chapter will discuss the method used to judge and rank the frameworks. The Analytic Hierarchy Process is a multiple-criteria decision analysis method that helps categorize and analyze decisions based on criteria the user considers essential for a definitive choice.

2.1 Software Testing

Software testing is a crucial part of any code development. Testing ensures that everything works and behaves according to the developer's specifications and helps prevent unexpected behavior [18]. Unit testing is the first form of software testing introduced to the development process [19] and is what beginners first learn when taught software testing. Unit testing is a form of validation testing. Validation in software development determines the correctness of the software by ensuring that all components behave as intended [20]. Unit testing is a subset test of validation testing and validates that the smallest pieces or units of software are related to each other functionally [21]. The main benefit of unit testing is to verify that individual code works as intended before integrating separate units of code that make the application.

2.1.1 Behavior and Test Driven Development

Test-Driven Development (TDD) and Behavior-Driven Development (BDD) are agile software development techniques used to improve workflow, test coverage, and validity of the software [22, 23]. Agile software development takes an iterative approach by revisiting its stages rather than a more traditional linear workflow [22]. An example of a linear workflow model is Waterfall. Waterfall uses multiple phases, where each stage acts as the output for the next and is still widely used because of its effectiveness when time constraints apply [24]. This approach makes it difficult to revisit and adjust previous stage inputs [24]; hence the name waterfall, since going up a waterfall is harder than following it. Both techniques are similar since BDD is an extension of Test-driven, where the goal is to test the behavior of units of code once multiple units are combined [25]. Test-driven is more about the individual units of code themselves and ensuring that all tests are available before any code is created [26]. Both techniques have stages where refactoring, meaning changing the code, is the crucial step. In the TDDs case, it starts with making the unit tests, implementing the code, and if the code is not passing the unit tests, change the code until it does [26]. Developing the tests usually starts with creating a failed test and then working on making the tests and other tests from there. BDD adds to this by making sure that any added future features or abilities are also its tests by comparing its predicted behavior to its actual behavior [25].

2.2 Testing Frameworks

This section talks about the testing frameworks used during the thesis. The frameworks that will be compared are, Jest [12], AVA [16] and TAP Node [27].

2.2.1 JEST JS

JEST [12] is a test framework designed and created by a Facebook team led by Christoph Nakazawa for the React [28] library [29].

React JavaScript (JS) or React is a JavaScript library used for building dynamic and declarative projects for the front-end [28]. Front-end is a term for all web software running on the local browser on clients' machines, such as graphical interfaces or javascript [30].

The test framework started as closed sourced and later moved to Facebook's open source project, where the community and Facebook work together [31]. JEST was developed on top of Jasmine JS [14], another testing framework known for its speed and behavior-driven approach. JEST added new functions that Jasmine did not have at that time, making JEST a more batteries included framework, skipping the need for any additional changes or installations to be made for a complete user experience [32].

JEST now works with almost any technology use, including other front-end and back-end frameworks [33]. The back-end is similar to the front-end but is software entirely run by a server. Information and data are stored and calculated on the back-ends business logic and then transmitted to the front-end of client machines [34].

Jest can be installed in any react project and is usually already configured for usage. JEST aims to be fast and easy with broad coverage in the types of tests [33].

2.2.2 AVA JS

AVA [16] is a minimalistic test framework created by Mark Wubben [35] and Sindre Sorhus[36] to give the developer confidence in their testing using highly opinionated syntax and user-friendly output. AVA is influenced by two other open-source testing frameworks called TAPE and TAP. TAP is an acronym for Test Anything Protocol and is a philosophy and testing framework on what information is relevant when outputting results[37]. TAP first appeared in the Perl scripting language [38] and was later adapted to other languages such as JavaScript [37]. Perl is a scripting language introduced in the 1980s and was originally designed for text manipulations, the creator Larry Wall [39] was a linguist that needed a tool for his work [40].

TAPE is an adaption of TAP but designed according to JavaScript guidelines [41]. AVA focuses on unit-testing and runs concurrently using Node JS, to ensure fast execution. NodeJS [42] is a JavaScript runtime framework that includes concurrency on all standard libraries. Because of this, it is possible to design other tools and frameworks on top of NodeJS without worrying about performance issues [42]. AVA also aims to work with any framework that uses Node JS for development, including many of the favorite front-end frameworks and other testing tools or frameworks, such as headless browser testing used in End-to-End. Headless browsers are browsers designed to be automated using predetermined commands and, sometimes, do not have graphical user interfaces, allowing developers to evaluate clients' experience on websites by simulating and benchmarking [43]. AVA aims to be developer-friendly, it has also added functionality that makes it possible to get different outputs using plugins created for TAP [16].

2.2.3 TAP NODE

TAP NODE [27] is a testing framework in JavaScript [44] that is inspired and designed to be like TAP, Test Anything Protocol, in other languages with minimal changes based on the author's interpretation of good testing methodology [27]. The creator and maintainer of TAP Node are Isaac Schlueter [45], and he uses GitHub and the community there to publish new versions at regular intervals. TAP Node was uploaded on Github in 2011 and went through iterations of alpha and beta versions before the first stable release happened in 2015 [46]. TAP Node takes a batteries-included approach, including code coverage, parallel execution, and an already established assertion set, giving developers everything they need to test their code in any library or tool. TAP Node practices against domain-specific languages, something that the creator says is a distraction that increases the lines of code and gives less coverage and tests [27].

2.3 MERN Stack

The MERN Stack [47] is a variant of a full-stack approach for developing web applications using Node JS [42] as the business logic in their application [47]. MERN stands for MongoDB [48], Express [49], React [28], and NodeJS [42] [47]. MongoDB is a database that only stores document data, meaning strings, and primitive data types in JSON format [50]. Express is an HTTP framework built on Node JS, used for communication such as APIs or integrating the front-end and back-end/business logic [49]. JSON, JavaScript Object Notation, is a data format used for exchanging data between computers [51]. MERN takes its inspiration from the MEAN stack [52], which is identical except for the front-end framework, Angular [52]. Angular is a front-end framework developed by Google [53].

The technologies are used in this manner: MongoDB is a data storage solution that uses documents and JSON as their solution instead of the traditional relational database, Express is for the API that connects the database, business logic, and front-end with each other, React as the front-end that the user interacts with and NodeJs for the business logic [47].

2.4 Analytic Hierarchy Process

Analytic Hierarchy Process (Analytic Hierarchy Process (AHP)) is a multi-criteria decision-making (Multi-Criteria Decision-Making (MCDM)) method [54]. Multi-criteria decision-making methods use criteria ranking to evaluate different available options. These options in MCDMs could be anything from humans to inanimate objects that need to be compared using the same criteria [54]. The Analytic Hierarchy Process uses this process and adds tools to ensure that users do not contradict themselves [54].

It was officially introduced in the book "Analytic Hierarchy Process" by Thomas Lorie Saaty [55] from the year 1980 [56]. The way the Analytic Hierarchy Process comes to a decision is similar to other MCDAs, by using divide and conquer. The first step is to consider the criteria/options that the user will compare. The criteria/options are placed in a pairwise matrix the size of $N \times N$.

The pairwise matrix borrows its characteristics from a standard comparison matrix, where the first column and first row are filled with the same criteria and are compared to each other. The diagonal line of the matrix represents how each criterion on the row compares to itself on the column, since they are the same, they are equally important and get the value 1. The number of comparisons made will be $\frac{n^2-n}{2}$, where n is the number of criterion's compared. The pairwise matrix is used for calculating what criteria are critical, as well as what alternatives fit better with said criteria, making the most important aspect of AHP.

The scale used for showing how important one criteria uses a scale of 1 to 9, where 1 says that the criteria are equally important, 3 is somewhat more important, 5 much more important, 7 very important, and, 9 more important, the other numbers are considered to be intermediate. The comparisons are made from row to column, meaning that the first row is compared to the other criteria on the column, if the row criteria are considered more important than the column then the corresponding scale is placed in the cell/entry of the matrix, if the opposite is true then the inverse of that scale is used instead, indicating that the column criterion has more important than the row. The finished calculations show each criterion or option's importance using percentages called weights, that is calculated on the final step of a pairwise matrix.

AHP [56] uses a hierarchy system, making it possible to use smaller sub-criteria for each main criterion established [57]. The sub-criteria would in this case be on the third level of a four-level hierarchy, enabling the user to give AHP more precise information as to what exactly they consider to be important [57]. The smaller sub-criteria can be compared to each other that belong to the same main criteria and will then share their main criteria overall percentage, giving them a final "global percentage". These new global values are then used to calculate the overall score of each alternative or framework, revealing the optimal option.

AHP makes sure that the user does not contradict themselves when determining importance by calculating a consistency ratio [58, 59]. The consistency ratio is determined by multiplying each row of the pairwise matrix by the sum of the weights that were calculated and then divided by the number of rows, this gives the principal eigenvalue that is used to calculate the consistency ratio. Saaty[55] mentions that since no human is ever consistent that the consistency ratio can fluctuate depending on what the user prefers, but later concluded that it is acceptable when the value is under 0.1 or 10% [58]. Meaning that the user can contradict themselves up to 10% on their decisions on what they consider important.

2.4.1 Bunge Scientific Method

Bunge's scientific method is an interpretation of the book 'Epistemology and Methodology I: Exploring the World, Vol 5' [60] by Bunge Mario [61] and is a series of stages in the technical process of evolving a theory or idea into research, followed by experiments, and finally evaluating everything into a conclusion [62]. There are smaller steps within the stages of Bunge's method. There are ten steps in total, with better details on how to perform each stage. The ten steps go as follows:

- 1 Identify a problem in a subject area.
- 2 Describe the problem clearly.
- 3 Map existing knowledge in the area, ie. identify information, methods, or instruments relevant to the problem.
- 4 Explain and solve the problem based on the background knowledge in step 3. If existing knowledge in the field is not enough to solve the problem, go proceed to step 5, otherwise skip to the next step.
- 5 Suggest new ideas, techniques, theories, or hypotheses and generate new empirical data for a solution to the problem.
- 6 Submit solution proposals, either an exact or an approximate solution.
- 7 Derive the consequences of the presented solution.
- 8 Test the solution proposal.
- 9 Correct the solution proposal after the test result (if needed).
- 10 Examine the solution from the perspective of existing knowledge in the subject area (step 3) and identified new issues.

As mentioned earlier, all steps belong to one of the three stages, either research, experimentation/application, or evaluation. The method is also designed to loop from step four to 10 indefinitely. Some of the steps can be changed and modified within reason as long as it still reaches the goal of each stage [62]. The interpretation was made by Niclas Andersson [63] and Anders Ekholm[64] in their report on what a scientific method should be [62]. Their interpretations show the general sequence Bunge thought any research paper should use.

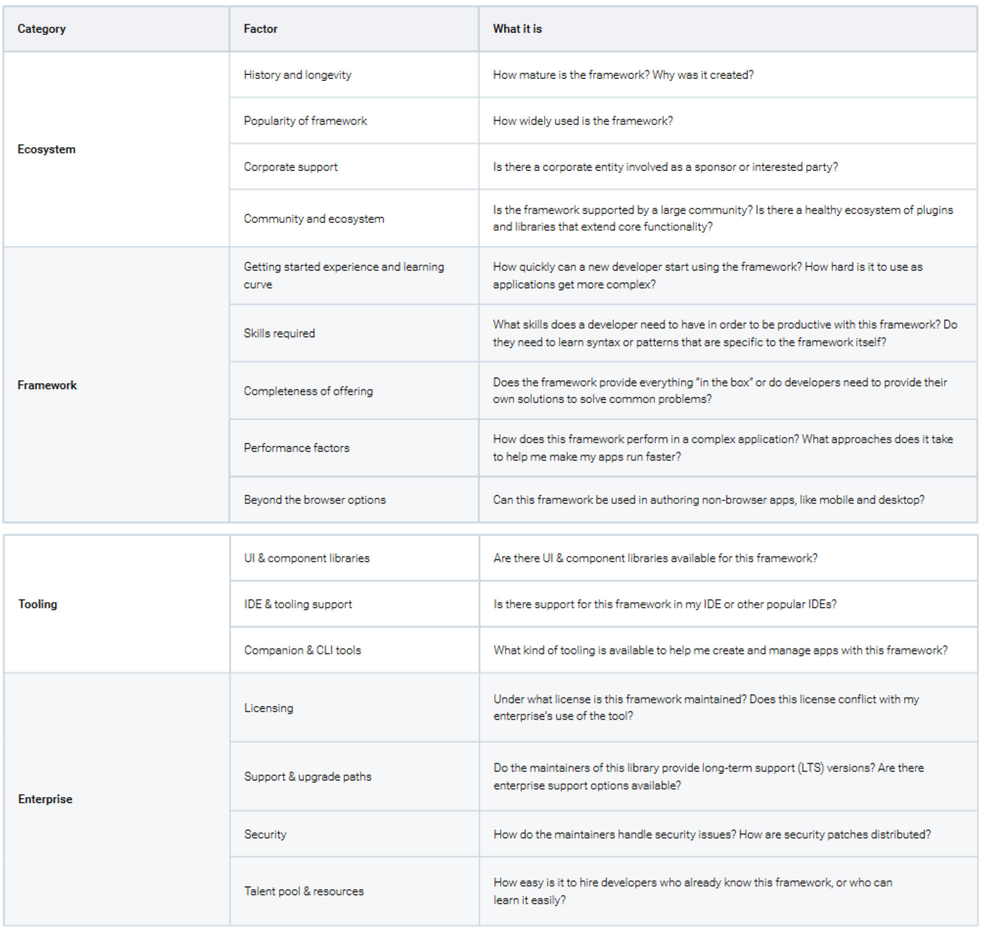
2.5 Related work

In Niclas Olsson's [65] and Nicklas Ockelberg's [66] work *Performance, Modularity and Usability, a Comparison of JavaScript Frameworks* they have three front-end frameworks in JavaScript compared in nine different criteria by making prototypes in each framework. The frameworks for their comparison are React, Angular, and Vue, three popular front-end frameworks. The research paper uses the analytic hierarchy process to conclude its findings [67]. Olsson's and Ockelberg's work are only similar regarding the method analytic hierarchy process. The goal of their work was to compare front-end frameworks in JavaScript. The criteria they presented follow three aspects and can only belong to one of them, Performance, Modularity, and Usability. The criteria are DOM-Manipulation, Memory Allocation, Startup Time, and Build Size for the performance aspect, Rich Package Ecosystem, Flexibility, and Code Reusability for the Modularity aspect and Documentation, and Learning curve for their Usability aspect. Their usage of the multi-criteria decision-making AHP method is the method implemented here but is not related since the thesis aims to evaluate JavaScript testing frameworks for beginners.

The usage of AHP is also different because of the usage of global and local criteria priorities in this work. The analytic hierarchy process has priority weights for each criterion on a pairwise comparison matrix and can be either a local priority or global. The local priority is the weights associated with only the criteria within the same pairwise matrix and is 100 percent. Global priority is the overall priority of a criterion when it shares 100 percent with all other pairwise matrices in the same hierarchical level. The global priority is figured out by multiplying the overlying criteria, which is the category the sub-criteria is under. This method makes it possible to combine the overall priority matrix from the opinion of beginners and mix it with the priorities of the sub-criteria. The formula is simply: Local criterion X parents criteria priority = global priority for local criterion. A data collection stage using surveys was also combined with AHP, shifting the results more towards the opinions of real JavaScript beginners. The criteria Flexibility and Documentation became the main criteria, with smaller sub-criteria, used in this thesis, with the learning curve being under another main criterion.

In the paper *Choosing the Right JavaScript Framework for Your Next Web Application* [68] by Brandon Satrom [69] there is a discussion and analysis about what criteria to use when comparing JavaScript front-end frameworks.

A table of criteria was established, it included a big range of criteria that belong to the categories, the ecosystem of the frameworks, the tooling, and licensing needed for operating. Figure 2.1 show the complete list of criteria and their explanation. Some of the testing framework criteria from the paper are being



Category	Factor	What it is
Ecosystem	History and longevity	How mature is the framework? Why was it created?
	Popularity of framework	How widely used is the framework?
	Corporate support	Is there a corporate entity involved as a sponsor or interested party?
	Community and ecosystem	Is the framework supported by a large community? Is there a healthy ecosystem of plugins and libraries that extend core functionality?
Framework	Getting started experience and learning curve	How quickly can a new developer start using the framework? How hard is it to use as applications get more complex?
	Skills required	What skills does a developer need to have in order to be productive with this framework? Do they need to learn syntax or patterns that are specific to the framework itself?
	Completeness of offering	Does the framework provide everything "in the box" or do developers need to provide their own solutions to solve common problems?
	Performance factors	How does this framework perform in a complex application? What approaches does it take to help me make my apps run faster?
	Beyond the browser options	Can this framework be used in authoring non-browser apps, like mobile and desktop?
Tooling	UI & component libraries	Are there UI & component libraries available for this framework?
	IDE & tooling support	Is there support for this framework in my IDE or other popular IDEs?
	Companion & CLI tools	What kind of tooling is available to help me create and manage apps with this framework?
Enterprise	Licensing	Under what license is this framework maintained? Does this license conflict with my enterprise's use of the tool?
	Support & upgrade paths	Do the maintainers of this library provide long-term support (LTS) versions? Are there enterprise support options available?
	Security	How do the maintainers handle security issues? How are security patches distributed?
	Talent pool & resources	How easy is it to hire developers who already know this framework, or who can learn it easily?

Figure 2.1: The list of categories, their criteria, and explanation of what the criteria represent.

used, mainly the learning curve, the documentation available, and command interface tools included criteria. These three have been included as either a sub-criteria or main criteria for the analytic hierarchy process and have been modified to fit testing frameworks. Satrom's work does compare front-end JavaScript frameworks and is different from this thesis on both method and subject but is still valid for any form of framework whether that is testing or front-end. The criteria "Getting stated experience" was modified to fit beginners and the installation process. "Learning curve" was also included in this thesis and other related work. The two criteria from the tooling categories, "Tooling support" and "CLI tools" was included and part of the flexibility main criteria.

Chapter 3

Methodology

There are three methods essential for proper research projects, namely data collection, data analysis to separate the data and analyze the findings, and verification and validation methods for assuring the quality of the research material [17].

Data collection methods are different forms of collecting information for a research project [17]. There are different collection methods for qualitative and quantitative research, the methods used for qualitative research is ***Language and Text*** [17] and ***Experiments*** [17], and ***Questionnaire*** [17] for quantitative research.

The Language and Text method is one of the most common forms of qualitative research done, where interpretations of words and meanings in research texts and documents occur to gather information about a topic [17]. This method was the first to be carried out to gather information about prior knowledge about the subject or topic area and answers to the sub-questions. The Experiment method is collecting data by applying and observing said experiments [17]. The experiments conducted collected data on how each framework performed in each criterion. The quantitative questionnaire method uses closed alternative answers for questions to collect data from sample populations [17]. The questionnaire method collected data using a survey to answer sub-question two since the qualitative research lacked information about the demographic beginners.

Data analysis methods help analyze and filter the data collected to transform and draw conclusions from your findings [17]. The different methods for data analysis is similar to data collection, where there are different methods for qualitative and quantitative research, the methods used here are ***Statistics***[17] and ***Computational Mathematics*** [17] for the quantitative research, and ***Content analysis*** [70] for the qualitative research.

Statistic analysis is just like it sounds, it's to calculate the data from collection methods such as questionnaires and evaluate the results from the sample population used [17]. Computational Mathematics is inputting the data gathered in numerical methods and algorithms to then conclude the results from the mathematical methods [17]. Content analysis is used for analyzing knowledge and the presence of certain words and concepts to determine if the qualitative data gathered was relevant and to conclude there [70]. These methods were all used. The Statistic analysis method was used for the survey, to see differences in opinions among beginners, and to determine the effect each opinion would have on the AHP calculations. The computational mathematics was implemented during the evaluation of the experiment, where the performance of each framework was converted into input for the analytic hierarchy process. The content analysis was conducted on all qualitative research because of its straightforward approach.

Verification and Validation methods are for quality assurance and are critical for the reliability of the research, ethics, and of course, the verification and validation of the research [17]. The approach for ensuring the verification of the quantitative research used iterations and constant comparisons to check if the testing conducted was relevant to the thesis and criteria performance to compare. A discussion about ethical concerns regarding the survey is also appropriate and a must. The reliability and validation of the methodology are conducted when the results are available and are usually written in the discussion section of the thesis. The same method was later used for sub-question 3 but in a different setting. Two new quantitative research was later made in the form of a questionnaire/survey and experiment, because of the lack of data available for sub-question two. The analysis method used for the questionnaire/survey research was ***Statistics***, given that it's the most common way to calculate results from sample groups and that the standard deviation is what is needed since 5/9 is the part of the whole. The experiment used Computational Mathematics The first approach for answering all sub-questions was an inductive qualitative research approach, that collects current data about the sub-questions. The information gathered is then analyzed, filtered, and reformed within the parameters decided in section 3.2.1. Content analysis [70], a data analysis method, was applied when analyzing the collected data for theories.

3.1 Scientific method and Project Management

An interpretation of Bunge's scientific method from Andersson and Ekholm [62] work was used during the project's method process, to ensure that it was executed using an already established method. The method was slightly modified to fit this project, meaning that the stages are now nine. The new stages go as following:

- 1. Identify a problem in a subject area.
- 2. Describe the problem.
- 3. Map current knowledge about the subject. Information,methods and instruments or tools necessary for the answering the problem.
- 4. Explain and solve the problem using information from stage 3. If the information from stage 3 is not enough, go to stage 5, otherwise go to stage 6 immediately.
- 5. Purpose new ideas,technique's, theories or hypothesize and bring forth data to solve the problem.
- 6. Give solution proposals, using either exact or an approximate solution. Apply the solution.
- 7. Derive consequences of the proposed solution.
- 8. (Optional) Create new sub-questions from information gather on stage 3. To gain broader knowledge about the subject.
- 9 Conclusion. (When done)

The sequence talks about identifying a problem within a subject and what the necessary steps are to find the answer for solving a problem in a subject area [62]. The method's implementation started in the first chapter and goes further toward the conclusion.The original sequence of stages or steps included a separate stage for testing the solution proposal and correcting the solution proposals after creating a test result. The stage specifying to implement the solution was combined into stage 6, and the correction stage was also removed because of the nature of the RQ.

The project management triangle or project triangle was used during the project process to ensure that everything happened within manageable scope. The variant of the triangle used had time, capital, and quality as the three restrains, or sides, to the project scope. The time available for the project restrained the implementation and scope by only allowing 3 different frameworks to be compared. Capital, sometimes called a budget, did not play a big factor since nothing during the project required a budget containing funds, only time was part of the budget. The quality of the results was also limited by time since more precise criteria could be used for a more inclusive result.

3.2 Research Approach

A qualitative research process was conducted to gain more knowledge about the research question, per the third stage of the Bunge method. To ensure that the data collection resulted in relevant data, a few rules and keywords were designed to limit the number of articles. The qualitative data could also be a part of a larger set of methods used to reach an answer for the research question, a figure was therefore created for an overview of how to data collected would be combined. The qualitative research used multiple keywords and sources to find answers for the questions asked in 3.1. Keywords such as

- Testing
- Beginner
- Automated-testing
- Interactive testing
- Teaching
- Taught
- Benefits
- JavaScript
- Framework
- Education

, were used together in multiple ways to find more information about the subject using the sources such as

- ScienceDirect
- scholar.google.com

- learning.oreilly.com / O'Reilly Safari
- DiVA portal
- IEEE Xplore
- Books with related subjects

The research was conducted during the period of September-October of 2021.

3.2.1 Limiting and Filtering the Qualitative Research

To ensure that only trusted sources were used in the making of this thesis, a set of criteria were established and used when searching for information related to the subject of this thesis. The set of criteria used during research:

- The articles used a preciously used method, containing data of the qualitative, quantitative form, if the method is not recognized, then the aforesaid article will later be evaluated, based on more research.
- The articles had to be accessible using a trusted source, with prior knowledge about the subject or regarded as a trusted source by people in the software industry.
- The article is either in English or Swedish. Other languages are passable if other works in English or Swedish explain the content of the original article.
- The article needs to have a topic about any of the keywords used during the research period. Or is somehow related to the sub-question.

The list was constructed using previously known criteria that the author decided were important for quality during research, reducing the number of articles to the essential one with about the subject.

3.2.2 Further research for SQ1

Smaller research, referred to as 'SR' henceforth, to answer sub-question one on what frameworks should be compared, by using its list of criteria for finding suitable frameworks to compare. Websites GitHub and NPM JS were used to find these frameworks using search tags "testing" and "framework" to ensure that each framework that appeared was test-related and was still actively developed. Their synopsis and description had to be different enough in methodology and descriptions such as "easy to use" and "flexible" was sought out. Easy and flexible to ensure that each framework could be compared within the allocated time, and flexible enough to be used in the same code project without collisions between test files.

3.3 Collecting data for RQ and SQ2

The information gathered from the qualitative research was determined to be incomplete for concrete answers to sub-question two and in turn the research question, and how to evaluate the frameworks for the research question. Making it clear that already established knowledge about the sub-subjects is not enough for an adequate answer, in these scenarios Bunge's method says that stage 5 is appropriate, as it says that own research might be required. An experiment and survey were therefore conducted to get closer to an answer, and it used the already discovered information to do so. A flow diagram was created to illustrate how and what was used to reach and establish an AHP result.

3.3.1 Motivation for the need of the Experiment

The experiment was implemented to gain more information about the sub-criterias (SC) priority for AHP and to make it possible for the author to compare them for the final weights. The research approach gave the overall main criteria and some of their SC but said nothing about what criteria could have higher priority. The experiment was conducted using a codebase of a full-stack application to create tests using the different testing frameworks. This made it clear what other criteria a beginner could consider valuable, giving a clearer picture of the criteria that SQ2 needed as answers. The experiment also gave the author the necessary information to give each sub-criteria their importance for AHP by using the frameworks. While this gave the final sub-criteria and sub-criteria scale, it did not answer what beginners thought about those criteria, this is what the survey has done instead.

3.3.2 Limitation and cause of the survey

The survey aimed to gather further information for SQ2, by asking beginners in a survey what main criteria they thought were a priority over the others. The reason for only asking them what main criteria they preferred instead of all sub-criteria, that could be categorized under the main criteria, was to ensure that people answered the questions in the survey instead of simply not answering anything. Asking survey participants to evaluate and compare multiple sub-criteria(SC) for AHP, which at the time were eight, would cause the survey to receive no answers, instead, the main criteria were presented with comments on what SC they contained and what each meant. This made it possible to combine the summarized thoughts of beginners with the findings from the experiment, making sure that actual beginners could include their views. The two least voted on frameworks were picked, as they are less popular and use different domain-specific language for their test, and Jest was also picked because of its popularity and the fact that it is included in ReactJS, the front-end framework with the currently highest downloads.

3.3.3 Ethical issues concerning Survey

The survey used the website known as SurveyMonkey [71]. The service offered an anonymous experience to the respondents by allowing anyone to answer without login in or using external services such as Google accounts. A survey link was generated to share with groups and individuals deemed appropriate for the study. Google Forms are free to use for anyone with a Google Gmail account. SurveyMonkey was used with a connected Google Account and was free to use with some advanced features blocked by a paywall. The only private information gathered during the survey was the participant's age, ensuring that only people at 18 and above answered.

3.4 Using the Analytic Hierarchy Process as a Computational Mathematical method

The Analytic Hierarchy Process(AHP) is used as the computation mathematical method for the data analysis solution because of its use of using subjective data and presenting it in an understandable quantitative form, using simple math.

AHP is considered to give an approximate result to finding the answer for the research question, and therefore fits into the 6th stage of Bunge's scientific method. AHP could also be seen as an appropriate method considering that the RQ is also subjective. The method used to combine the main criterion's weights calculated using the survey answers and sub-criteria weights was the method example shown by Anders Hermansson in his research papers example of advanced AHP usage [72].

Chapter 4

Requirements and design

Requirements and design are two sides of the same coin when creating a system. When developing a system, used to solve a specific problem, it is usually necessary to delve into the requirements and design. Requirements are focused on what the system is solving and what it needs to solve said issues. The design is about how to implement the system using the requirements.

4.1 System Requirements

The overall goal when designing a system is to produce one according to the requirements asked or needed, making it higher quality [73]. Requirements fall under two categories, functional and non-functional [73]. The solution proposed in chapter three is the system mentioned here.

Functional requirements are the functions the system has to meet to perform its intended job [73]. An example of a functional requirement is user input and the system doing something about it [73].

Non-functional requirements are the parts of the system that are not part of the service that the system provides but are still part of the system[73]. An example of non-functional is things such as flexibility and availability [73].

- Functional requirements for the system are:
    1. **Using multiple sources for criteria and priority**: The system should be able to integrate criteria and priorities from research

papers and questionnaires from specific demographics. These sources should also affect the results in their favor.

2. **Conducting experiments**: The system should allow objective observations from experiments to affect the priority and evaluation of each criterion and framework's performance.

- Non-functional requirements for the system are:

  1. **Scalability**: The system should be scalable enough to add or remove the number of criteria and frameworks to be evaluated, making it possible to increase or decrease the scale of a research area.

  2. **Flexibility**: Being able to change and modify the system after development makes it more resilient and changeable against industry, community, and new users' needs or implementations.

4.2 System Design

The system is designed according to the waterfall method and uses the previous stage's output as input for the current stage, because of its simplicity and other benefits the waterfall method provides. The diagram 4.1 follows a top to bottom flow, shows the flow, and has comments about the type of information flowing. The implementation is designed to follow a more structured approach of qualitative research, quantitative research, and then using a Computational Mathematics method to combine everything into a final result. The figure starts in Stage one, where qualitative research is conducted and some criteria were established. The second Stage inputs the results of the qualitative research, which has given some main criteria and sub-criteria, and inputs do to the Experiments for analysis and basis for other criteria and Survey to ask beginners about their opinion. Stage thee shows that the results from the Experiment and Survey are used n AHP for the final answer. There are comments connected to the same arrow, that explain more about the input data used.

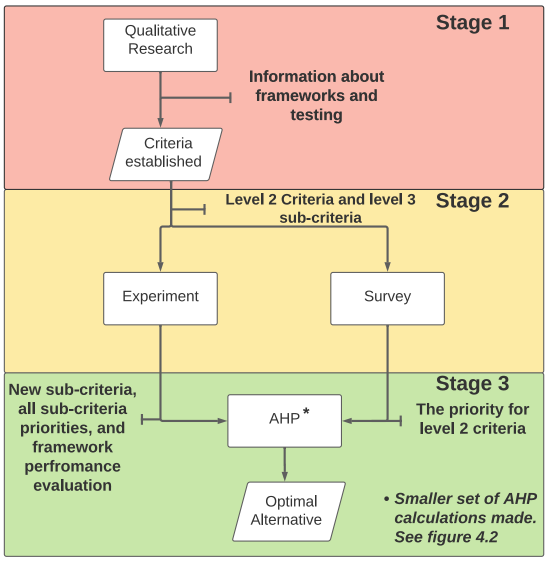

Stage 1
Qualitative Research
Information about frameworks and testing
Criteria established
Level 2 Criteria and level 3 sub-criteria
Stage 2
Experiment
Survey
Stage 3
New sub-criteria, all sub-criteria priorities, and framework perfromance evaluation
AHP *
The priority for level 2 criteria
Optimal Alternative
• Smaller set of AHP calculations made. See figure 4.2

Figure 4.1: Diagram showing the stage flow conducted for the system proposed.

Diagram 4.2 shows further details about how the data from the second stage is used, for the computational method AHP. Diagram 4.2 shows where the input comes from and how it is combined to give the final result.

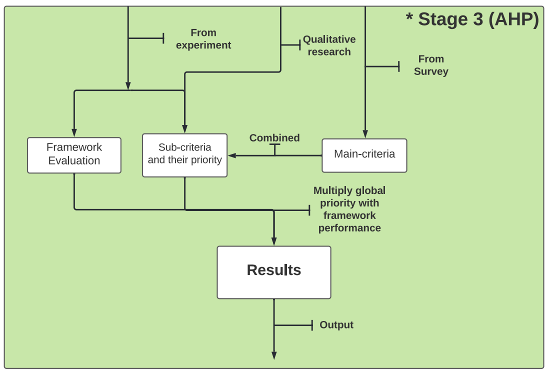


Figure 4.2: Showing how the internal AHP calculations will be made for the final results, using multiple AHP tables and combining results.

4.2.1 Designing Survey questions

The survey contained 3 stages, each of which have questions to gain information about the participant's knowledge of programming and testing. The first stage had questions if the participants had any experience using JavaScript and testing code. The second stage had similar questions as the first stage but wanted information about the participant's experience using testing frameworks in JavaScript, and what they used. The last stage had questions about where they considered themselves to be, in terms of the experience of testing code, and what they prioritize when looking for a testing framework. These stages were used to filter out the beginners from everyone else who answered the survey since they were the demographic the survey wanted to collect data from, beginners. The information gathered from the survey was used to calculate the priority weights for the second-level criteria of the AHP hierarchy. Each criterion got its priority for the pairwise matrix, by taking their score and using the difference between them as the values for their priority over the other options.

4.3 Software Used

The operating system used was Windows 10 21H1 with the combination of Windows subsystem for Linux or WSL, using the Ubuntu distribution with release 20.04. WSL had tools included in Linux that analyzed the result from the experiment. NPM was the package manager used for all module needs during the study, Node was also included as it was required for the modules to work. They used versions 6.14.15 and 14.18.0 respectively. Visual studio code was the IDE used during the experiment, it has multiple plugins for snippets and better syntax highlighting for a better experience, using version 1.61.0. All frameworks used the latest stable versions as of October 2021.

Chapter 5

Implementation

This chapter shows how the steps of how Bunge's scientific method was implemented with the research, experiment, survey, and AHP calculations in the method chapter were implemented to gain the result. The events of the implementation are presented in chronological order.

5.1 Implementing Bunge's scientific method

Bunge's method is essentially a list, see 3.1, of the steps needed to ensure that conducted research is scientific and is using an approach that is well established in the academic world. The first two steps refer to identifying the problem in the subject area and describing the problem(s) pinpointed. These two steps have already been carried out through the introduction chapter, using the background 1.1 subsection for step one and problem 1.2 subsection for step two.

The third step is to map the current knowledge about the subject and find methods and tools to help find answers if needed. The research approach 3.2 was used in this steps of the method since it showcases what keywords and sources helped find and gather information about the topic. Subsection 3.2.1 is the set of rules to filter and limit the research.

The fourth step says to explain and solve the problem in the subject area with the information gathered, but the qualitative research only answered SQ1 and partially answered SQ2. The fourth step of Bunge's method says that if the information is not enough go to step five. The fifth step of the sequence is where other ideas and techniques must be used to bring new data.

The fifth step of Bunge's method resulted in a proposed solution to use a survey or questionnaire to answer SQ2, "What criteria do beginners use for evaluating frameworks?" and another smaller form of qualitative research to find testing frameworks for SQ2. Section 3.2.2 has details about how the new qualitative research for SQ2 was conducted. The survey asked simple questions about their experience in programming and testing, what type of testing they had been taught first, and what criteria they considered most important when choosing testing frameworks. More details on how the survey and the ethics surrounding it are available in section 3.3.2. This is also where the method to answer how to evaluate the frameworks based on their criteria came up, multi-criteria decision-making (MCDM) methods were a new solution, and the specific type of MCDM selected was the analytic hierarchy process.

The sixth step of Bunge's method is the implementation of the experiments done with the finished website, the performance evaluation for each framework in their sub-criterion, and the overall calculations done for the analytic hierarchy process.

The seventh step discusses the solution and its consequences, positive or negative. The discussion could be criticism against or justification for the method or methods used, about the reliability and validity of the research, and how the results turned out.

The eighth step is optional and only included if a new sub-question appears during the entire process and if that question is required to answer the original research question.

The ninth step concludes the paper. It brings up the results and their significance, if any limitations have occurred, if there are any improvements future work could perform, and if the author has reflections on the research and implementation.

5.2 Implementing the experiment

This section shows how the experiment was conducted, showing the setup process,tests made and what was compared. The frameworks chosen are Jest, AVA, and TAP Node, three different testing frameworks for JavaScript.

5.2.1 Setting up the experiment

Modification to both Jest and AVA configuration files enabled them to ignore a specific file extension or folder. TAP does not have this problem since it uses regular JavaScript files for testing, which the others disregarded. The file extensions used are ".spec.js" and ".test.js" both were used but only by one framework each, so Jest used ".test.js" while AVA used ".spec.js." Listing 5.1 and 5.2 are the modification made to Jest's configuration file, with Listing 5.3 showing how AVA's configuration file (package.json) was modified.

```
testMatch: [
    "**/__tests__/**/*.[jt]s?(x)",
    "**/?(*.)+(test).[tj]s?(x)"
  ],
```

Listing 5.1: Here are the changes made to Jests pattern matching when it looks up what files to run through. The object testMatch contains the two string patterns that Jest will use when searching for test files to run. The first string tells Jest to go through the entire software project and find the folder labeled tests, and run files that end with a test.js. The other line tells Jest to run its test no matter the folder name.

```
testMatch: [
    "**/__tests__/**/*.[jt]s?(x)",
    "**/?(*.)+(test|spec).[tj]s?(x)"
  ],
```

Listing 5.2: The second string in the testMatch object originally contained a clause stating that any file ending in spec.js or test.js would also run through Jest, meaning that Jest would run code from other testing frameworks and thus result in a failed test.

```
"ava": {
    "files": [
      "Tests/AVA_Test/*"
    ]
  },
```

Listing 5.3: Ava's configuration was simpler to modify than Jests. The newly added change tells Ava that the only files it needs to run are in the Tests folder and its sub-folder "AVA_tests", ignoring everything else.

5.2.2 Evaluating frameworks through the experiment

The experiment used Visual studio code as the IDE of choice because of its support native JavaScript support. One principle of Test and behavior-driven development (subsection 2.1.1) is to modify a failed test to a passed test using incremental changes. The comparisons will start by showing how each framework handles these use cases.

Both Jest and AVA require a test function to pass through the name and code to work, while TAP only needs to call the function "fail" with a comment as the parameter. Tap's way does make it easier to create a failed test with two short lines, making it easier for users to perform the first step of TDD and BDD. However, implementing further testing code requires the user to replace the line, like in listing 5.7, in contrast to Jest and AVA, where the test function stays but the internal changes as TDD and BDD usually intents. The listings 5.4 to 5.6 show the code.

```
// Jest

 test("Simplest way to get a failed test",() => {
   throw new Error("This is an error")
});
```

Listing 5.4: This listing shows how Jest prefers to fail its tests on purpose. It starts with creating a test module and then passing another function that throws an error with an optional command.

```
// AVA
 const test = require('ava');
```

```
test("Simplest way to get a failed test", t => {
    t.fail();
});
```

Listing 5.5: This listing shows how AVA purposely fails a test, AVA is imported, and a test is made with an inner function unit that calls the fail function that AVA includes for us.

```
// TAP
 const tap = require('tap')
 tap.fail('fail');
```

Listing 5.6: This listing shows how TAP Node fails using a simple call to the global tap module, instead of creating a test and then failing the testing using the inner function.

```
// TAP, example of simple match assertion
 const tap = require('tap')
 tap.test("Comment", t => {
    const value = returnValue();
    t.match(value, "value")
 })
```

Listing 5.7: How TAP test for match/equal value, dramatically different than simply passing or failing a test.

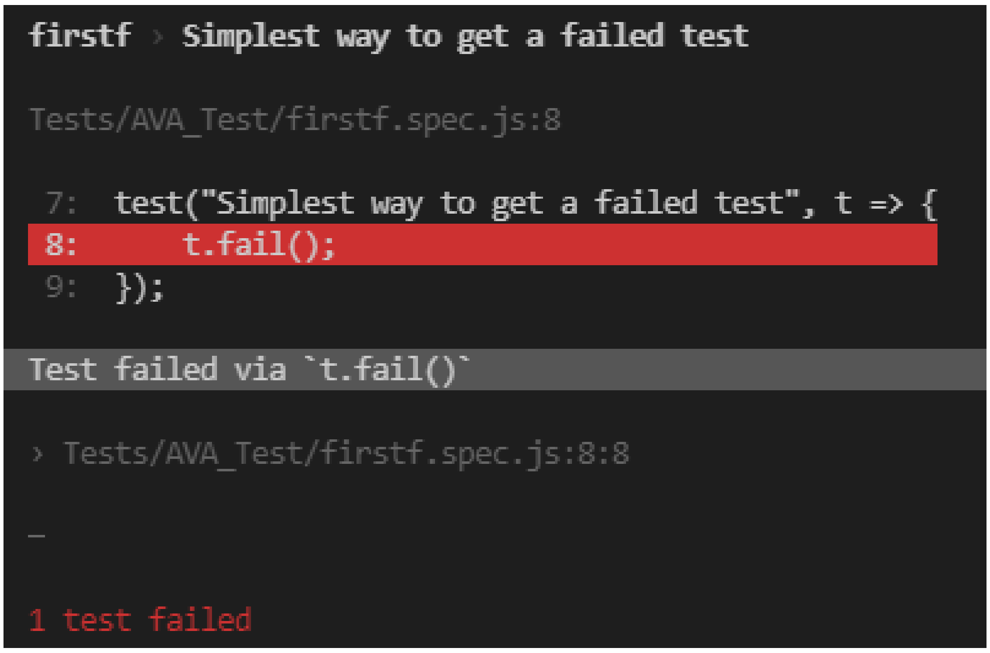
5.2.3 Framework assertions

All three frameworks have their functions for comparing values, using names such as expect in Jest, is in AVA, and match in TAP. They all have more advanced versions of it, where the type of the values also matter when comparing them. The frameworks differ in the number of built-in assertions that help the developer along with development. Jest has over 50 assertions [74], where they are handy in different areas, such as comparing the closeness of floating-point numbers (expect.closeTo) or finding some matching data in objects (expect.toMatchObject) come to mind.

Their documentation also shows how most are one or a few lines without setup. AVA and TAP do expect the user to implement most certain case tests by themselves using either macros or custom functions. This contrast does make AVA and TAP have a shallow learning curve since it essentially becomes JavaScript's learning curve. All frameworks have self-explaining names that are easy to remember, making them readable.

5.2.4 Framework tooling and reporters

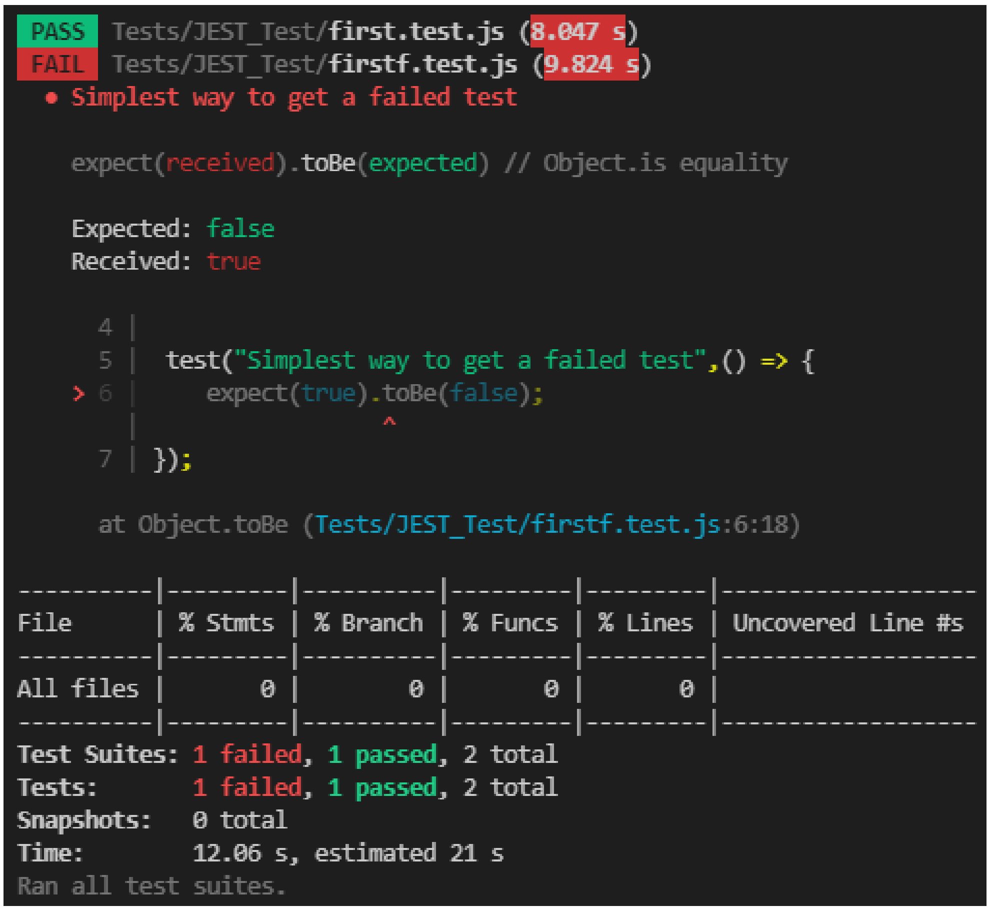
The default reporters on all three explain what test failed, on what line, and why. Both Jest and TAP show the time spent on each test and continue to perform all tests until everything has been executed, whereas AVA stops all testing if a single test fails. AVA is also missing a code coverage reporter by default, but the documentation does recommend using the c8 coverage tool. Both Jest and TAP have code coverage tools that use Istanbul, an open-source project, preinstalled and enabled by default. Listings 5.1, 5.2, and 5.3 also show how verbose each output becomes respectively, which makes it harder to read and analyse when a larger number of test fail.



```
firstf › Simplest way to get a failed test

Tests/AVA_Test/firstf.spec.js:8

 7:  test("Simplest way to get a failed test", t => {
 8:      t.fail();
 9:  });

Test failed via `t.fail()`

› Tests/AVA_Test/firstf.spec.js:8:8

─

1 test failed
```

Figure 5.1: AVA's outputs a failed test, and showcases that in the same fashion as the image.

AVA's output, as shown in 5.1, tells the user what test fails, in what file that test is in, and on what line this occurs. It is a friendly output that tells only the useful information needed to fix those failed tests. But it only shows the first test that fails, making it hard to know why other fails tests. AVA does not include a code coverage tool, giving it a worse position compared to the other frameworks.

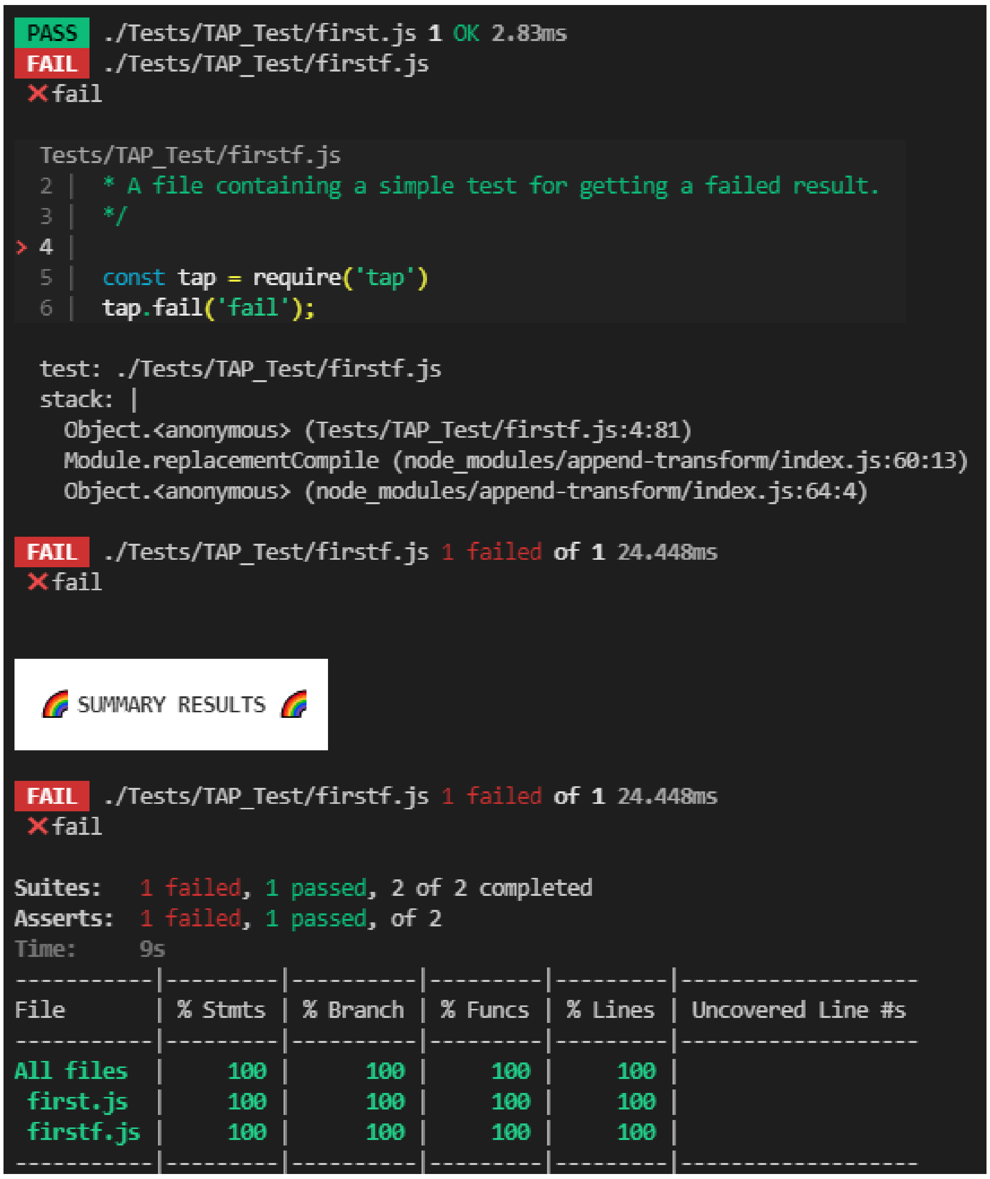


```
PASS  Tests/JEST_Test/first.test.js (8.047 s)
FAIL  Tests/JEST_Test/firstf.test.js (9.824 s)
  ● Simplest way to get a failed test

    expect(received).toBe(expected) // Object.is equality

    Expected: false
    Received: true

      4 |
      5 |   test("Simplest way to get a failed test",() => {
    > 6 |      expect(true).toBe(false);
        |                   ^
      7 | });

      at Object.toBe (Tests/JEST_Test/firstf.test.js:6:18)

----------|---------|----------|---------|---------|-------------------
File      | % Stmts | % Branch | % Funcs | % Lines | Uncovered Line #s
----------|---------|----------|---------|---------|-------------------
All files |       0 |        0 |       0 |       0 |
----------|---------|----------|---------|---------|-------------------
Test Suites: 1 failed, 1 passed, 2 total
Tests:       1 failed, 1 passed, 2 total
Snapshots:   0 total
Time:        12.06 s, estimated 21 s
Ran all test suites.
```

Figure 5.2: Jest's outputs a failed test, and shows related information.

Jest's output, presented in figure 5.2, report contains everything related to the tests conducted. It includes what tests failed and passed, why they failed, and on what line and character number resulted in a failed test. Code coverage is also included but not configured in this configured, and is therefore showing 0 coverage. It also shows the time needed to run through all tests.

Figure 5.3 shows how TAP Node's output is similar to Jest but is significantly more bloated, with unfriendly stack messages for each failed test and a summary result that takes up too much space. The coverage reporter is included, just as Jest, and is working instantly.



```
 PASS  ./Tests/TAP_Test/first.js 1 OK 2.83ms
 FAIL  ./Tests/TAP_Test/firstf.js
 ✕ fail

   Tests/TAP_Test/firstf.js
   2 |  * A file containing a simple test for getting a failed result.
   3 |  */
 > 4 |
   5 |  const tap = require('tap')
   6 |  tap.fail('fail');

   test: ./Tests/TAP_Test/firstf.js
   stack: |
     Object.<anonymous> (Tests/TAP_Test/firstf.js:4:81)
     Module.replacementCompile (node_modules/append-transform/index.js:60:13)
     Object.<anonymous> (node_modules/append-transform/index.js:64:4)

 FAIL  ./Tests/TAP_Test/firstf.js 1 failed of 1 24.448ms
 ✕ fail


   🌈 SUMMARY RESULTS 🌈

 FAIL  ./Tests/TAP_Test/firstf.js 1 failed of 1 24.448ms
 ✕ fail

Suites:   1 failed, 1 passed, 2 of 2 completed
Asserts:  1 failed, 1 passed, of 2
Time:     9s
-----------|---------|----------|---------|---------|-------------------
File       | % Stmts | % Branch | % Funcs | % Lines | Uncovered Line #s
-----------|---------|----------|---------|---------|-------------------
All files  |     100 |      100 |     100 |     100 |
 first.js  |     100 |      100 |     100 |     100 |
 firstf.js |     100 |      100 |     100 |     100 |
-----------|---------|----------|---------|---------|-------------------
```

Figure 5.3: TAP's outputs a failed test, and reports a lot of information.

5.3 Establishing the main criteria for AHP from the qualitative research

The qualitative research conducted resulted in information to evaluate front-end frameworks and languages based on multiple criteria. An adjustment to these criteria established a set of custom criteria to testing frameworks that fit the subject of this paper. The custom set concluded to a list of three, simplicity, documentation, and features. All three of these criteria contain smaller sub-criteria which explain what the main criteria indicate. The main criteria names were chosen as the group name for other smaller criteria that make a testing framework friendlier to beginners.

5.3.1 Simplicity Criteria for Tested Frameworks

Simplicity has the sub-criteria setup, behavior-driven or Test-driven development practicable, and learning curve, both setup and a good learning curve criteria were established by teams in companies when deciding tools for their project [75]. A good and friendly setup experience and the learning curve can make or break a testing framework for beginners. A good start investment in the framework makes it more worthwhile, especially for beginners in a new language. The learning curve will also represent the ease of use and readability of the framework, since they are related, and having them as a separate criterion would require more time, which is limited according to the project triangle in 3.1.

The BDD or TDD practical criteria were added by the author as an extra criterion so that the testing framework should be friendly, considering that BDD and TDD have become a big part of testing development. It has been shown to decrease defective software and time when using TDD as a method for development compared to conventional methods [76], and might be used by the user in the future.

5.3.2 Documentation Criteria for Tested Frameworks

The purpose of documentation is to show the user how to use the framework, which uses examples and explanations about how the framework works. The documentation criteria, which was also added by the author from the experiment and survey as the documentation was used constantly on how to implement the framework, have two sub-criteria that ask if information about the framework is available on the official framework website, GitHub website, and/or Stack Overflow since documentation can be unclear, outdated or has nothing to contribute for a specific problem the users has encountered.

5.3.3 Feature Criteria for Tested Frameworks

Features is a custom group name for criteria that include tooling and other functionality. These sub-criterions are considered in frameworks because of their effect on the experience of using the framework [68]. The sub-criteria selected are command line interface tooling and support, Advanced features, and flexibility.CLI is always considered when using frameworks, the CLI is what the user interacts with after creating tests [68], it is primarily used for the test reports, how readable and understandable it is under different circumstances such as passed and failed tests. Advanced features are tools such as code coverage, to ensure that all code is under some kind of confirmed passable tests, and other functionality that could be used in the future, such as mocking. Flexibility was added as a sub-criteria for the framework's ability to configure itself to the user's need, such as what tests to ignore if the tests should use concurrency, and what result in reporters the framework should use. The CLI sub-criteria was modified to fit with what testing frameworks use. Both Advanced features and Flexibility were added by the author from their experience of using these frameworks during the experiment period. Figure 5.4 show the hierarchy structure made so far, with the priority of each main criteria. The hierarchy diagram that will be used during the implementation of the analytic hierarchy structure, shows the hierarchical levels and includes the main criteria and their priorities from the survey conducted.

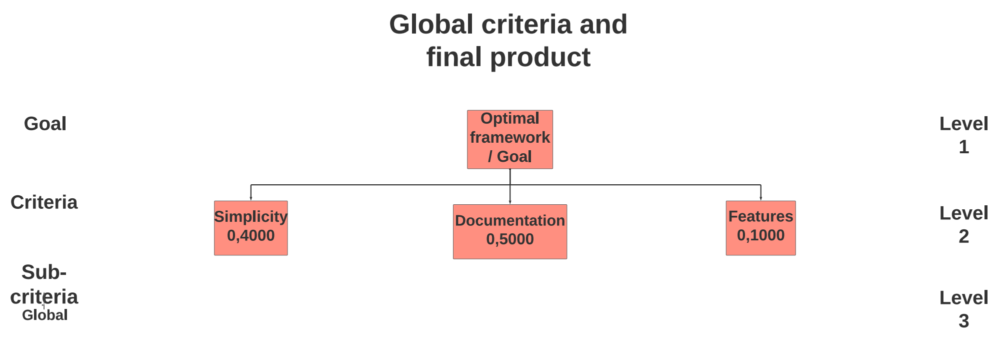


Figure 5.4: Showcases all levels of the hierarchy, the main criteria and their priorities.

5.4 AHP calculations

The software used for implementing AHP was Microsoft Excel. Each step of the process was divided into colored sections that show the chronological steps throughout the calculations. All of the calculations are based on the relation each criterion has, and because of this was the first thing to be added. An example of this is the first matrix, which uses the main criteria, you could also call them categories, Table 5.1 shows how the layout and sum of each column. The standardized version of Table 5.1 is shown in Table 5.2, with their final weight on an extra column to the right. The pairwise matrix 5.1 shows how each criterion compares to the others in the matrix. The matrix shows that the Documentation criteria are 20 percent more crucial than the Simplicity criteria, while Simplicity is four times more important. These are converted from the survey results and are included as pairwise matrices to simplify future processes

Pairwise	Simplicity	Documentation	Features
Simplicity	1	4/5	4
Documentation	5/4	1	5
Features	1/4	1/5	1
SUM	2 1/2	2	10

Table 5.1: The matrix shows priority differences between each criterion.

Standardized	Simplicity	Documentation	Features	Weight
Simplicity	4/10	4/10	4/10	40.00%
Documentation	5/10	5/10	5/10	50.00%
Features	1/10	1/10	1/10	10.00%

Table 5.2: Standardized matrix of Table 5.1 showing each criterion's final local priority.

5.4.1 How to read a pairwise AHP matrix

The priority for each cell/element in a pairwise matrix is determined based on how important the row criteria are compared to the column criteria, if they have the same one then the cell has the priority value of 1, since a criterion is only as important as itself, not less or more. These natural 1's create a horizontal line that separates the matrix into a right and left side, where the left side is simply the inverse of the right, and the right side is where the user gives each priority their value based on the AHP scale. If the opposite is true, that the column criteria have a higher priority than the row, then the cell has the inverse of the priority value on the right side, and an integer on the left instead.

Table 5.1 is used as an example of a pairwise matrix, that was also used for the main criteria, belonging to the second level of the hierarchy, it shows that the Simplicity row criteria is four times more important the column criteria of Features, meaning that the cell was the criteria switch places has a value of 1/4. A similar thing occurs with Documentation and Simplicity, where Documentation is considered to be about 20 percent as important as Simplicity, meaning that the right side of the horizontal line is a fraction and the left side is an integer, the opposite of the previous example. An extra row called SUM was added by the author to the bottom of the matrix, it is there for feature calculations when the matrix is standardized for their respective priority weights.

5.4.2 How to standardize an pairwise AHP matrix

A standardized AHP gives the final priority weight of criteria, this is achieved by dividing all the cells by their column sum, to make it easier for readers, the sum of each column will appear in the non-standardized version of the AHP matrices, as shown in Table 5.1. This division makes it easier to compare and evaluate each criterion's priority in each column. Once everything has been standardized, it is time for calculating the priority weights. This is done by adding up each criterion in the row axis, creating a new column to the right of the matrix. the sums of each row are then divided by N, the number of criteria in one matrix, which will finally show the priority based on percentage. An example of the final output is Table 5.2, which is also the result for the first matrix.

5.4.3 How the main criteria is used with the sub criteria

The same steps, as shown above, were used for each pairwise matrix, that contains the sub-criteria of Simplicity, Documentation, and Features. The sub-criteria weights were collected in a list and then multiplied with the priority weight of the main priority that they belong to, so Setup with Simplicity, Website|GitHub with Documentation, and CLI with Features. These new weights are called global priorities because of how they are affected by the main criterion's weights. This method makes it possible to combine different levels of the AHP hierarchy while ensuring that everything falls within 100 percent, also known as the percentage of a percentage.

5.4.4 Determining the sub-criteria priority and presenting global weights

The matrices shown in this subsection belong to the third level of the hierarchy, as shown in figure 5.5, aka the sub-criteria. Only one comparison between the criterion's priorities will be made for each matrix, but a list on GitHub is available for the rest, keeping this paper short while still showing how each criterion got its priority. The knowledge gather from the qualitative research, mentioned in chapter 3, and the subjective understanding of beginners by the author was used to determine the priority of the criterion in each pairwise matrix. The simplicity matrix shows that the Learning curve is considered to be 5 times more important than the setup criterion, considering how a beginner would in all likelihood drop the testing framework because of the difficult learning curve, in favor of an easier tool. The weight calculated from Table 5.3 would result in 0,129 for Setup, 0,277 for BDD | TDD, and 0,595 for the Learning curve, showing that the Learning Curve will later be a bigger factor than the other two. These priorities were then multiplied with the priority weight for Simplicity shown in Table 5.2, resulting in table 5.5. This was also done with the Documentation and Features matrices, as shown in Tables 5.6 (standardized in 5.7) and 5.8 (standardized in 5.9).

The hierarchy from figure 5.4 has evolved, is 5.5 and now includes the sub-criteria from the related work and experiments conducted and is now in the third hierarchical level. Each sub-criteria belongs to a criterion in the second level.

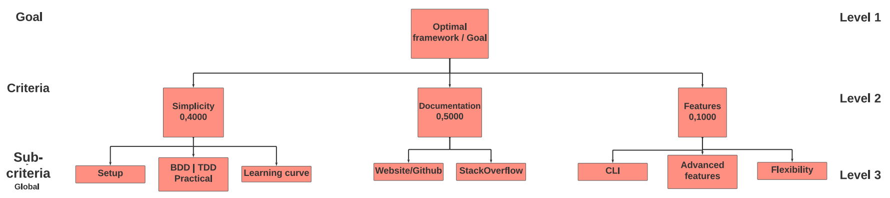


Figure 5.5: The evolved version of 5.4.

Simplicity pairwise matrix, figure 5.3 used for the third layer of the hierarchy, used in the results and as an example of how to calculate the global priority of the sub-criteria. It shows that BDD | TDD is twice as important as Setup, and that Learning curve is five times more than Setup

Simplicity	Setup	BDD | TDD	LC
Setup	1	1/2	1/5
BDD | TDD	2	1	1/2
LC	5	2	1
Sum	8	3 1/2	1 7/10

Table 5.3: Table showing how the simplicity pairwise matrix looks like.

Standardized	Setup	BDD | TDD	LC	Weight
Setup	1/8	1/7	2/17	12.85%
BDD | TDD	1/4	2/7	5/17	27.66%
LC	5/8	4/7	10/17	59.49%

Table 5.4: Standardized matrix of Table 4.3 with priority weights to the right. Using 2 decimals or numbers when needed.

Priority	Local priority	Global priority
Setup	0,1285	0,0514
BDD | TDD	0,2766	0,1106
Learning curve	0,5949	0,2380

Table 5.5: A table showing how the global priorities for simplicity. The right column shows the global priority, that will be used with the scores of each framework.

The Website|GitHub criteria in Table 5.6 has a higher priority than Stack Overflow, since the official documentation has to be clear about features and examples for a beginner to find what they need easier. Stack Overflow is instead only used for when solving configuration problems and questions that the documentation does not explain clearly and thus was decided to have a lower priority.

Documentation	Website | GitHub	Stack Overflow
Website | GitHub	1	3
Stack Overflow	1/3	1
Sum	1 1/3	4

Table 5.6: A table showing how the documentation pairwise matrix looks like.

Standardized	Website | GitHub	Stack Overflow	Weight
Website | GitHub	3/4	3/4	75%
Stack Overflow	1/4	1/4	25%

Table 5.7: Standardized matrix of Table 4.5 with priority weights to the right. Using 2 decimals or numbers when needed.

The table above shows that the CLI criteria have a higher priority than the others because their outputs and commands can affect productive workflow for beginners. Yet other criteria are still considered to be important functions, such as flexibility when using these frameworks because the user might not like the coverage tool or error reporter. That is why this matrix has more of a one, two, and three approaches than the other sub-criteria matrices, because of how everything is connected and how they can affect each other.

Features	CLI	Advanced	Flexible
CLI	1	3	2
Advanced features	1/3	1	1/2
Flexible	1/2	2	1
Sum	1 5/6	6	3 1/2

Table 5.8: A table showing how the features pairwise matrix looks like.

Standardized	CLI	Advanced	Flexible	Weight
CLI	6/11	1/2	4/7	53.90%
Advanced	2/11	1/6	1/7	16.38%
Flexible	3/11	1/3	2/7	29.73%

Table 5.9: Standardized matrix of Table 5.8 with priority weights to the right. Using 2 decimals or numbers when needed.

Priority weights	Local	Global
Setup	12,85 %	5,14 %
BDD | TDD	27,66 %	11,06 %
Learning curve	59,49 %	23,80 %
Website | GitHub	75 %	37,50 %
Stack Overflow	25 %	12,50 %
CLI	53,90 %	5,39 %
Advanced features	16,38 %	1,64 %
Flexible	29,73 %	2,97 %

Table 5.10: A matrix showing the sub-criterion's local priority weights for inside the same matrix, and global priority weights from the product of the categories and criterion's priority. Using percentage instead of decimals.

Table 5.10 is the resulting output from everything so far. It shows each criterion's local priority weight that is only affected by other group members with the same color. It also shows how each criterion has a new priority weight on the right column, caused by the overlaying criterion from level 2 of the hierarchy. These new global values were used in the same way as the main level 2 matrix was used, by multiplying these new values with the local framework priority weights, which will be shown later in this chapter.

5.4.5 Evaluating each framework

A similar approach was used for calculating the framework's overall priority weight, the main difference is that the priority acts more like points for each framework in this case. Each sub-criteria gets a matrix, where the three frameworks are compared, the better a framework performs on the criteria the higher the score, and the scores are still relative to each other like before. Eight new matrices were created, and their priority weights at the end were then multiplied with the sub-criteria global priority, giving a final percentage for each framework, the framework with the highest percentage was then the most suited choice according to AHP. Setup was the first sub-criteria to apply this, the other matrices were also implemented similarly. Each framework receives its overall performance in each sub-criterion through a point system. Each framework starts with one point and gets one more for a better implementation. The difference in points decides by what factor each framework is better at each sub-criterion. Setup is presented in Table 5.11 and shows that the Jest framework is considered to perform better than the others. The caption used for Table 5.11 is how the author determined the points each framework received, however, future captions will only include minor detail, to keep the section short. More information about the reasoning behind each framework's scores will be included in a text file on GitHub [77].

Setup	Jest	AVA	TAP
Jest	1	2	3
AVA	1/2	1	2
TAP	1/3	1/2	1
Sum	1 5/6	3 1/2	6

Table 5.11: The figure shows the information from the section prior, where Jest performs better by a factor of two compared to AVA and three times better than TAP.

Figure 5.11 shows that Jest scores the highest because of its pre-installed nature in react projects, ease of installation on any JavaScript project, and ease of configuration. AVA has a similar setup to Jest but loses a point for not being pre-installed on any tool or environment. TAP only received one point because it can be installed using npm, but is not pre-installed on any tool, and has a somewhat manual configuration setup, where it requires the user to dump the files from TAP itself.

BDD|TDD	Jest	AVA	TAP
Jest	1	1	2
AVA	1	1	2
TAP	1/2	1/2	1
Sum	2 1/2	2 1/2	5

Table 5.12: Jest and AVA score the same because of their TDD nature, Tap does not use either method without using outside tools.

LC	Jest	AVA	TAP
Jest	1	2	3
AVA	1/2	1	2
TAP	1/3	1/2	1
Sum	1 5/6	3 1/2	6

Table 5.13: The pairwise matrix that Jest has a twice and three-time easier learning curve than AVA and TAP respectively.

TAP and AVA do not include assertions for specific situations that require high-order operations, making it necessary to implement own JavaScript functions to solve. TAP requires more code in most cases to implement the same test like the others. This information is reflected in figure 5.13.

Website | GitHub	Jest	AVA	TAP
Jest	1	1	3
AVA	1/2	1	3
TAP	1/3	1/3	1
Sum	2 1/3	2 1/3	7

Table 5.14: It shows that both Jest and AVA is three times better in terms of documentation than TAP.

All three frameworks have GitHub repositories, but both Jest and TAP have their main documentation on their own website. AVA has everything on GitHub for simplicity and makes it easier to contribute to the documentation from the community. Guide lists for combining with other tools is available for AVA and Jest, but is missing from TAP as shown in 5.14.

Stack Overflow	Jest	AVA	TAP
Jest	1	2	3
AVA	1/2	1	2
TAP	1/3	1/2	1
Sum	1 5/6	3 1/2	6

Table 5.15: Jest comes out on top in this criterion because of its wide adoption. Showing that Jest is in a similar position to before when comparing Learning curve performance.

In the sub-criterion "Stack overflow" it appeared that Jest and AVA have Stack overflow tags, making it easier to search for answers, but TAP for Node does not. Jest has a sizable number of votes, questions, and answers compared to the other two.

CLI	Jest	AVA	TAP
Jest	1	2	1
AVA	1/2	1	1/2
TAP	1	2	1
Sum	2 1/2	5	2 1/2

Table 5.16: The matrix tells that both Jest and TAP perform better than AVA by a factor of two.

When testing each framework in the sub-criterion "CLI" it appeared that all frameworks allow the user to use them simultaneously with minimal configuration. Jest and TAP include code coverage out of the box, AVA requires additional installation and changes to the config. Possible to change reporter and code coverage reporter on all frameworks, for customization and preferred look. AVA requires configuration to continue to run if a single test fails. AVA reporter is easier to read by far.

Advanced	Jest	AVA	TAP
Jest	1	2	1/2
AVA	1/2	1	1/3
TAP	2	3	1
Sum	3 1/2	6	1 5/6

Table 5.17: Jest and TAP include mocking for future usage. TAP includes a plugin that can run mocha tests.

Flexible	Jest	AVA	TAP
Jest	1	2	2
AVA	1/2	1	1
TAP	1/2	1	1
Sum	2	4	4

Table 5.18: Customization the framework inside package.json is possible, but Jest includes its own file for this, making it less cluttered.

The final step for implementation

Taking the results from Table 5.19 and multiplying them with the global priority weights creates a new framework weights table, presented in chapter 6, changing the values and adjusting them to the global priority of the actual sub-criteria and their importance for the optimal framework.

Framework performance	Jest	AVA	TAP
Setup	53,9%	29,7%	16,4%
BDD|TDD	40%	40%	20%
Learning curve	53,9%	29,7%	16,4%
Website|GitHub	42,86%	42,86%	14,28%
Stack Overflow	53,9%	29,7%	16,4%
CLI	40%	20%	40%
Advanced features	29,7%	16,4%	53,9%
Flexibility	50%	25%	25%

Table 5.19: This is the final table that will be needed to calculate the final results. This table contains the priority weights of each framework when it comes to their performance with specific sub-criteria. The percentage with the same color belong to the same framework. Each row shows that the priority weights belongs to the same sub-criteria named on the far left column.

Chapter 6

Results and Analysis

6.1 Results

Table 6.1 and figure 6.1 is the result of using the Analytic Hierarchy Processes to determine the criterion's priority weights and the optimal testing framework asked by the research question (RQ), "Finding an optimal testing framework for beginners in JavaScript programming". The process uses multiple sources such as qualitative research, experiment, survey, and the author's perceived criteria and framework importance.

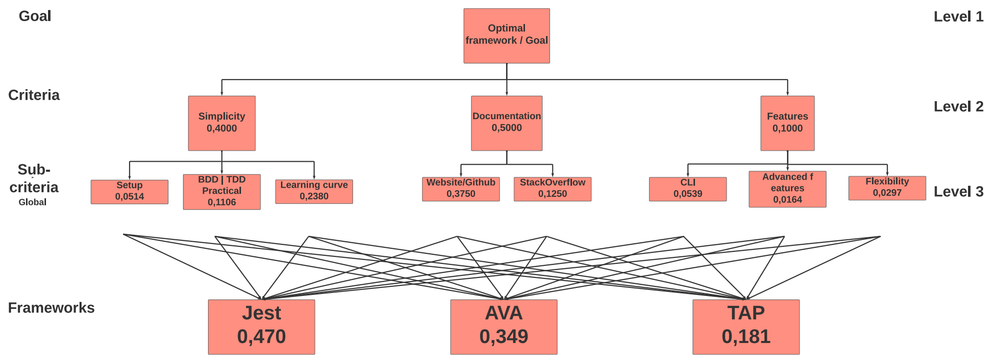


Figure 6.1: AHP diagram with priorities and final result

	Framework		
Criteria priority	**Jest**	**AVA**	**TAP**
Setup	2,77%	1,53%	0,84%
BDD | TDD	4,43%	4,43%	2,21%
Learning curve	12,82%	7,07%	3,90%
Website | GitHub	16,07%	16,07%	5,36%
Stack Overflow	6,74%	4,72%	2,05%
CLI	2,16%	1,08%	2,16%
Advanced features	0,49%	0,27%	0,88%
Flexibility	1,49%	0,74%	0,74%
Sum:	47,0%	34,9%	18,1%

Table 6.1: This table shows that multiplying the framework weights from table 5.19 with the global priority calculated during section 5.4.4.

n	3	4	5	6	7	8	9
RI	0,5245	0,8815	1,1086	1,2479	1,3417	1,4056	1,4499

Table 6.2: The generated RI set by Alonso and Lamata using 100 000 random generated tables compared to Saaty's original 500. [78]

Main criteria Lv2	λmax	**CI**	**CR**
	3	0	0.0%

Table 6.3: A small table containing the consistency ratio and related numbers, from the main-criteria, level 2 of the hierarchy, shown in table 5.1.

	λmax	**CI**	**CR**
Simplicity	3.005538701	0.0027695	0.528%
Documentation	2	0	0
Features	3.009208667	0.0046045	0.878%

Table 6.4: The calculated consistency ration, Alpha max, and consistency index. Using the three matrices, Simplicity, Documentation, and Features from the second level of the analytic hierarchy process. The values shown are from tables 5.4 to 5.9

Framework	λ**max**	**CI**	**CR**
Setup	3.009208667	0.0046045	0.878%
BDD TDD	3	0	0%
LC	3.009208667	0.004604333	0.878%
W|GitHub	3	0	0%
Stack Overflow	3.009208667	0.004604333	0.878%
CLI	3	0	0%
Advanced	3.009208667	0.004604333	0.878%
Flexible	3	0	0%

Table 6.5: The collection of consistency ratios from the framework matrices. Calculated from table 5.19.

6.2 Survey results

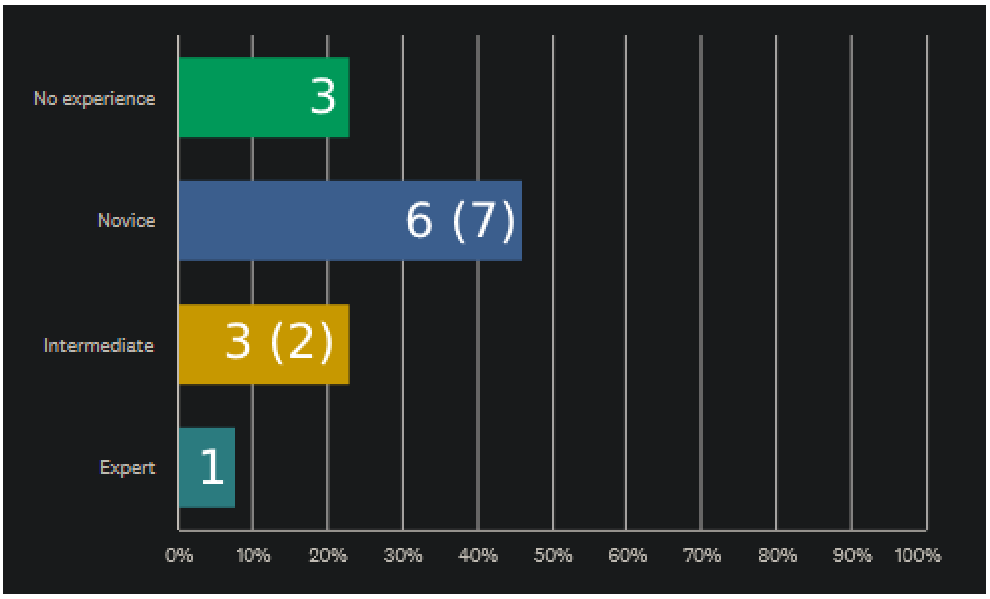


Figure 6.2: This survey question asked "What is your level of expertise when it comes to testing software?", one of the answers was placed in intermediate instead of a novice. The parentheses show their final value, after correcting.

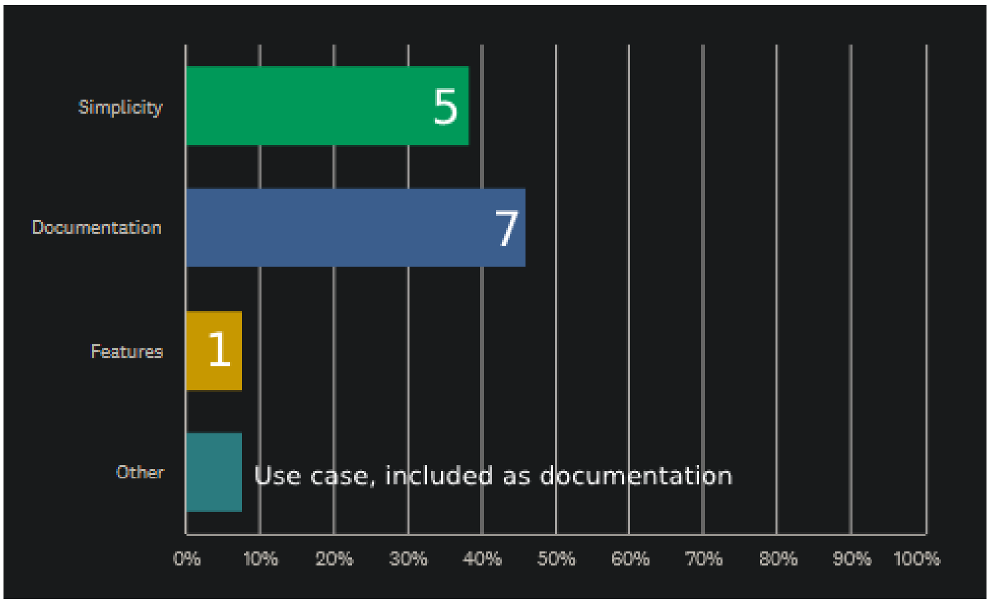


Figure 6.3: This question asked, "What would you consider to be the most important aspect when learning a new framework?". The "other" was a good use case example and was determined to belong with documentation.

6.2.1 Understanding the tables and figures

Figure 6.1 is the final overall result, showing a hierarchy according to the analytic hierarchy process method. It goes from the top to the bottom, level 1 to level 3. Level three is the framework alternatives, which will get a decimal value based on their performance. Level one represents the goal of the process, which is to find the optimal testing framework for JavaScript beginners and has the value one, meaning 100% that is its importance or priority interchangeably. The second level is the main criteria, which divides the goal into three main criteria, it shows what alternatives are important based on the higher decimal number, where the decimal values are from the filtered survey. Level 3 is the sub-criteria, and it acts in the same way that level 2 does with the goal, which is to divide their parent criterion's priority between themselves depending on how important they are. This is done by multiplying their calculated importance with their parent criterion's priority.

Table 6.1 shows how the different frameworks perform in each sub-criteria. The table 5.19 is multiplied with the global priority in table 5.10, resulting in a new table with performance of each framework based on the criteria priority. A higher percentage means that the framework fits the criteria better than the others. Adding up the column values gives an overall performance for each framework in row sum. It shows that Jest performed better than the others. The survey tables 6.2 and 6.3 are the two main questions used to determine what main criteria beginners consider crucial over the others. Individual responses were combed through so that only the opinions of novices or those with no experience affected the result in 5.1, the main criteria table. Any section about the survey in discussions will be excluded since it's explained throughout the paper.

Consistency ratio

Tables 6.3 to 6.5 is the collection of the consistency ratio of the matrices used during the implementation and the other values needed to calculate it. The consistency ration or CR is needed to figure out if the user of AHP is contradicting themselves when determining each criterion's importance in pairwise matrices. It uses the largest eigenvalue of the matrix and the size of the matrix, n, to determine the consistency index. The formula used:

$$Consistency \quad index = \frac{\lambda max - n}{n-1}$$

, where λmax is the largest eigenvalue.

The CR is calculated by dividing the consistency index (CI) with the random index (RI).

$$CI = \frac{CI}{RI}$$

The random index is the average value of consistency ratios from randomized pairwise matrices. For every size of matrices, represented by N, there is a specific random index belonging to the same size. The RI values belong to the same set of RI values, where it goes from the size of 3 to 9 for N. All matrices lower than 3, meaning 1 and 2, get a consistency index of zero making the average of these values zero. The largest size for random indexes is because AHP only supports those numbers, and any size of N larger is undefined. The set of RI values used for these tables come from a sample size of 100 000 tables used in Alonso and Lamata's research, which is available on table 6.2. Alonso and Lamata's research "CONSISTENCY IN THE ANALYTIC HIERARCHY PROCESS: A NEW APPROACH" further develops the calculations made by Saaty by using a larger set of random matrices to calculate the consistency index, giving the users of AHP a better consistency ratio [78]. The conclusion for the CR calculations is in chapter 7.

6.3 Sub-question results

The divide and conquer method was used to reach the major result using smaller sub-questions. These sub-questions helped with answering the main research question and was organized into two questions:

1. What frameworks should be compared?

   Jest, Ava, and TAP node were picked as the frameworks to compare for this report. Jest is the default framework for React projects and is what the experiment codebase uses. Jest is also one of the most popular testing frameworks for any JavaScript project. AVA is opposite to Jest, AVA has a significantly smaller community and focuses mainly on support for any JavaScript projects. TAP Node uses an adapted version of the 'Test Anything Protocol' used in other languages, the syntax, and way of thinking is different from how JavaScript does things.

2. What criteria do beginners use for evaluating frameworks?

   The criteria simplicity, documentation, and features were the main ones determined to be important when choosing frameworks. These main criteria contained smaller sub-criteria that made the main criteria. The sub-criteria helped with calculating their priority in AHP.

Chapter 7

Discussion

This chapter will talk about the choices made during the method and implementation of this research paper. The main criteria and sub-criteria will be discussed, including criticism of the analytic hierarchy process afterward. A subchapter about the reliability and validity of this paper will also be made at the end of the chapter.

7.1 The results and criterion

This section will contain three subsections representing the main criteria. Each criteria section will discuss what framework performed better and why.

7.1.1 Simplicity

Simplicity contains three sub-criteria setup, BDD | TDD, and learning curve. Jest was determined to perform the best in this area. It is pre-installed in almost ant react project, allows configurations in multiple ways, has an uncomplicated structure and naming for functions, making the learning curve better. Jest has more advanced assertions for objects and promises, assertions as toMatchObjects that compare the internal state between objects, and "resolves" that resolve promises.

7.1.2 Documentation

Jest and AVA have simple, understandable, and friendlier documentation for beginners by using examples for their API. TAP has the same approach only for the basics, making it harder to recommend. Portions of the documentation for all frameworks describing different assertions had only brief text-based explanations. Jest and AVA's documentation also have "recipes" that explain how to combine their frameworks with other tools for testing, editing, and JavaScript frameworks. Jest has substantially more votes and answers on stack overflow, making it more likely than the others to have solutions for beginners for different problems.

7.1.3 Features

Jest was determined to achieve better efficiency in the main criteria Features. In the first sub-criteria CLI, both Jest and AVA had built-in code coverage reports using IstanbulJS, a popular coverage tool, while AVA only recommends using code coverage separately. AVA will also stop all subsequent tests if anything fails. AVA also says that tests pass without saying what succeeded, something the author considers less desirable. TAP has a slight advantage in advanced features with its inclusion of changeable domain-specific language by including Mocha JS as an alternative to TAP syntax. Finally, Jest makes it easy to configure its settings with either a separate file or changes to package.json, Jest does also have more support for other testing tools, as mentioned earlier as recipes.

7.2 Consistency Ratio discussion

As mentioned in chapter 2.4 the consistency ratio is a crucial step of AHP to ensure that the user does not contradict themselves or have placed values randomly. Saaty has stated that any pairwise matrix with a consistency ratio below or at 0.1 (10%) is an acceptable consistency ratio and is otherwise contradicting itself [58]. There are also recommendations for different matrix sizes based on Saaty's observations, where a matrix of three should not have a consistency ratio higher than 0.05 (5%) or a matrix with four having anything above 0.08 (8%) [79]. Tables 6.3 to 6.5 from results show the consistency ratio on the far right column and values affiliated with it to the left. All consistency ratios are below 0.1, with the recommendations of below 0.05 applied, ensuring that all matrices are valid according to AHP.

7.3 Criticism against the Analytic Hierarchy Process

Ensuring that the process to evaluate the frameworks using the Analytic Hierarchy Process is scientific through validity ensures that this paper is scientific and suitable for the questions asked in the method chapter. Bringing up criticism against AHP and explaining how those issues were solved or avoided will help it in the right direction. Two research papers about AHP will be presented.

The research paper *"Analytic Hierarchy Process – en kritisk genomgång"*[80] by *Anders Hermansson* from 2014 brings up two arguments about critical issues with AHP. The first issue is the software known as Expert Choice™, the primary way AHP is used in the industry. Hermansson argues that the way Expert Choice™ hides matrix operations and can recommend adjustments to the pairwise matrices to lower inconsistency [81] can result in wrong results without the user understanding why. The author implemented calculations in Microsoft Excel and verified them by hand, ensuring that the matrix operations and criteria priorities were according to the author's original intent.

The second issue that Hermansson argues is rank reversal. A rank reversal occurs when a previously established pairwise matrix rearranges its priority order upon changes to the number of criteria in the matrix; a reorder that should not occur [82]. An example of this is how the priority order $C > B > A$ changes to $B = C$ with A absent, rather than simply resulting in $B > A$. The previous example presented results in equal value for criteria B and C because they are equally important in both matrices, but the influence of criteria A changes their relationship. The calculations done follows the AHP method according to its specification, with no changes to any matrix after establishing the criteria, meaning that rank reversal does not affect the results of this research paper. Hermansson does mention a solution involving the method *ideal mode* [83], where all local priorities are divided by the largest priority from the same local criteria list, and in turn, also changes how the global priority weights works.

The second research paper *Criticisms of the Analytic Hierarchy Process: Why they often make no sense*[84] by Rozzan Whitaker brings up another essential criticism against AHP that the majority argue when discussing the method. The issue brought up is that any mathematical operations (addition, multiplication, division, normalizing, and others) done to any pairwise matrix comparison values change the overall priority weights, when they should not since they all change in the same way. Whitaker explains why they are often baseless when applying AHP according to its requirements, such as the hierarchical structure and criteria weights. Whitaker brings up an example where a user normalizes the values of each row before adding them up, resulting in the wrong priority weight. But by using the hierarchical structure containing the criteria weights established before, we can determine the correct priority weights by multiplying the values. Whitaker argues that the lack of hierarchical structure and criteria weights mentioned will always conclude in incorrect outcomes and is flawed. Whitaker then shows the same example but uses priority weights, which results in the correct priority weights from the same matrix before modification. Meaning that AHP will have an accurate outcome if the user follows the requirements for the method.

The implementation of AHP used by the author is valid according to the original paper on AHP [56] and Hermansson's examples [80] making the argument against AHP invalid, the hierarchical structure is applied, while the main criteria priorities as based according to data collected during the survey. They are also no mathematical modifications done to any pairwise matrix during the implementation process.

7.4 Reliability and validity Analysis

Reliability and validity are two similar and related concepts used to assess the quality of the research paper's methods. Reliability ensures that the results can reproduce given the same methods under the same conditions, making them consistent. Validity ensures that the results measured during the research are what they are supposed to measure.

7.4.1 Reliability

The reproducing of the same results given the same research done ensures that the work is credible, making reliability essential. There are four stages done in the implementation section, qualitative research, survey, experiment, and AHP calculations. The qualitative research done followed the research approach shown in section 3.3, if the same rules and sources apply, then the reliability of the results for the qualitative research is high, based on the information available at the current time of this study. The survey is quantitative and is only reliable considering that the majority who answered have a background in computer science or similar, and should be kept in mind when applying similar or additional surveys. The experiment was done solely by the authors and could be considered less objective, but the same conditions apply to the survey. The authors have a background in programming and created the tests according to the documentation recommended suggestions, making them reliable only when someone of similar knowledge performs the experiments. The results helped determine the sub-criteria priority and the framework's performance, while the author decided the number in AHP calculations using the results. The calculations of these are reliable when considering performing them according to section 5.2. The method for using global and local priority in conjunction was from Saaty's AHP work and other research papers and is reliable by proxy. The reliability of the computations for AHP was done in the same manner and checked multiple times to ensure correctness.

7.4.2 Validity

The validity of this research paper is determined by how well the qualitative research, survey, experiment, and analytic hierarchy process is to evaluate their intended objective; answer the research question(RQ) and sub-questions(SQs). In the case of the survey, its purpose was to figure out what criterion out of the main criteria beginners thought of as a priority when choosing a testing framework. The information was sifted to only include the results from beginners to help adjust the AHP results closer to their opinion. The survey answered one sub-question, SQ2, which subsequently answered the main research question (RQ). The calculations used for the results implemented the analytic hierarchy process, answering the research question(RQ) while following Saaty's instructions on AHP. While the qualitative research and experiment answered the other sub-questions, one through three, indicating that all measurements used during this paper were related to and answered either the research question or sub-questions.

Chapter 8

Conclusions and Future work

This chapter concludes the research presented and discusses the last three steps of Bunge's method from section 3.1. Other discussions such as limitations, future work, and reflections will appear near the end of the chapter. The reflections section will discuss the social and ethical aspects of the paper while excluding the environmental and economical due to their nonexistence.

8.1 Conclusions

The research question (RQ) "Finding an optimal test framework for beginners in JavaScript programming" is about finding one framework suitable for beginners in JavaScript from a list of alternatives that all claim perfection. Jest claims to give a simple user experience that is also fun, while AVA claims simplicity and speed in a similar sense. The headlines make it difficult for beginners to choose based on overall performance.

Results from AHP answer the RQ by giving a clear upper hand to Jest JS, with AVA and TAP Node after it, respectively. From Table 6.1, it becomes obvious why Jest becomes the overall performer. Its Stack overflow presence and better learning curve give it a higher percentage. While AVA is just behind in exactly does areas, Node TAP has far worse performance in all criteria. Jests and AVA's similar results show that even smaller projects that rely on code contributions can almost keep up with big technology corporation and their open-source organizations. The two sub-question answered crucial pieces of information regarding what frameworks to use, and what criteria beginners have or use when comparing frameworks. The frameworks chosen were decided based on their differences in popularity and methods used. The criteria used for the comparison were chosen based on prior work and the author's previous knowledge and were decided based on the areas appropriate for beginners discovered during experimentation. Both sub-questions could be answered differently based on other requirements and factors such as different demographics or even author(s). The goal has also been answered, where differentiation between frameworks has been done using criteria and using beginner-friendly criteria to determine it is suitable for the same group.

8.2 Limitations

The limitation of only using three frameworks to compare did help with time constraints but could have affected the conclusion from earlier. Additionally, frameworks would give a better picture of the current situation for users. Including other forms of testing such as end-to-end and mocking could have given a better and more comprehensive view of how those frameworks perform in all testing areas considering how the experiment had limited number of functions suitable for unit testing, but this could count as an out-of-scope addition. Sub-criteria could also be further divided, such as the learning curve where readability and ease of use are included instead of separated as their own criteria. The survey had a limited range of questions due to time constraints and the risk of staying unanswered for too long.

8.3 Reflections

The social and ethical characteristics of the paper are related to the survey conducted. As mentioned in 3.3.3, the only private information gathered was the participants' age, which helped to filter out answers from minors. The participants were aware that their submission would be included and affects the results. The service used for collecting the data did not require an online account, but did block contributors from submitting multiple answers, so a level of tracking was probably implemented in the background using either cookies, tokens, or other methods.

This thesis follows the principles of research ethics, such as anonymity respect.

The data collected had integrity since nothing was modified, fabricated, or selected explicitly by inappropriate manipulation.

8.4 Future work

Further research into framework criteria and priorities would also benefit the reliability. A way of collecting more information is by giving surveys or/and examinations to different groups where programming beginners are. Places such as coding boot camps or universities would be suitable for finding beginners. This data compiled using the geometric or arithmetic mean would grant better empirical evidence and could establish a better objective point of view. Making more definite learning curve graphs would also be possible if done over time. Using another multi-criteria decision-making (MCDM) method, instead of AHP, could also be possible and comparisons between the results could be discussed in length.

References

[1] G. J. Myers, C. Sandler, and T. Badgett, “The art of software testing,” vol. 3rd Edition, p. 2, 1980. [Online]. Available: https://books.google.se/books?hl=sv&lr=&id=GjyEFPkMCwcC&oi=fnd&pg=PP7&dq=software+testing&ots=AivPE-qY2k&sig=Y1TWd8Ir7e9kK8qyyzrzGKnPsDA&redir_esc=y#v=onepage&q=software%20testing%20has%20become&f=false

[2] ——, “The art of software testing,” vol. 3rd Edition, p. 1, 1980. [Online]. Available: https://books.google.se/books?hl=sv&lr=&id=GjyEFPkMCwcC&oi=fnd&pg=PP7&dq=software+testing&ots=AivPE-qY2k&sig=Y1TWd8Ir7e9kK8qyyzrzGKnPsDA&redir_esc=y#v=onepage&q=software%20testing%20has%20become&f=false

[3] M. A. Umar and C. Zhanfang, “A study of automated software testing: Automation tools and frameworks,” vol. Vol. 8, p. 217, 2019. [Online]. Available: http://www.ijcse.net/docs/IJCSE19-08-06-011.pdf

[4] L. Williams, G. Kudrjavets, and N. Nagappan, “On the effectiveness of unit test automation at microsoft,” vol. IEEE, p. 85, 2009. [Online]. Available: https://ieeexplore.ieee.org/document/5362086

[5] A. R. Chowdhury, “Best 9 javascript testing frameworks,” vol. January 24, 2020, 2020. [Online]. Available: https://www.lambdatest.com/blog/top-javascript-automation-testing-framework/

[6] Z. Pandovski, “Github, a curated list of awesome testing tools,” vol. 2022 12 MAR, 2022. [Online]. Available: https://github.com/ZoranPandovski/awesome-testing-tools#automated-testing-tools

[7] S. Overflow, “Programming, scripting, and markup languages,” 2021. [Online]. Available: https://insights.stackoverflow.com/survey/2021#most-popular-technologies-language-prof

[8] DevSkiller, “72% of companies are looking for javascript developers,” p. 1.3, 2020. [Online]. Available: https://devskiller.com/it-skills-report-2020/#One

[9] Testim, “What is the best unit testing framework for javascript?” 2021. [Online]. Available: https://www.testim.io/blog/best-unit-testing-framework-for-javascript/

[10] A. Kothari, “11 best javascript unit testing framework and tools,” 2021. [Online]. Available: https://geekflare.com/javascript-unit-testing/

[11] J. Hartman, “Best javascript unit testing frameworks,” 2022. [Online]. Available: https://www.guru99.com/javascript-unit-testing-frameworks.html

[12] I. Codesido, “Jest: Delightful javascript testing.” 2009. [Online]. Available: https://github.com/facebook/jest

[13] O. Foundation, “Mocha simple, flexible, fun.” [Online]. Available: https://mochajs.org/

[14] D. W. Frank, “Jasmine a simple javascript testing framework for browsers and nodejs.” 2010. [Online]. Available: https://web.archive.org/web/20140222155946/http://pivotallabs.com/jasmine-1-0-released/

[15] ChaiJS, “Chai js.” [Online]. Available: https://www.chaijs.com/

[16] M. Wubben and S. Sorhus, “Node.js test runner that lets you develop with confidence,” vol. 2021, version 3.15.0. [Online]. Available: https://github.com/avajs/ava

[17] A. Håkansson, “Portal of research methods and methodologies for research projects and degree projects,” in *Proceedings of the International Conference on Frontiers in Education : Computer Science and Computer Engineering FECS’13*. CSREA Press U.S.A, 2013. ISBN 1-60132-243-7 pp. 67–73, qC 20131210. [Online]. Available: http://www.world-academy-of-science.org/worldcomp13/ws

[18] IBM, “What is software testing?” 2022. [Online]. Available: https://www.ibm.com/topics/software-testing

[19] G. D. Everett and R. McLeod, “Software testing: Testng across the entire software development life cycle,” vol. Wiley-IEEE Press, pp. 51–52, 2007. [Online]. Available: http://worldcolleges.info/sites/default/files/

software-testing-testing-across-the-entire-software-development-life-cycle.9780471793717.28214.pdf

[20] W. R. Adrion, M. A. Branstad, and J. C. Cherniavsky, "Validation, verification, and testing of computer software," vol. NBS Special Publication 500-75, p. 5, 1982. [Online]. Available: https://www.govinfo.gov/content/pkg/GOVPUB-C13-d8cbebacc4749b5c4284cce1ffb796d1/pdf/GOVPUB-C13-d8cbebacc4749b5c4284cce1ffb796d1.pdf

[21] IEEE, "610.12-1990 - ieee standard glossary of software engineering terminology," vol. 12-9-2002, p. 79, 1990. doi: 10.1109/IEEESTD.1990.101064. [Online]. Available: https://ieeexplore.ieee.org/document/159342

[22] A. Alliance, "What is agile software development?" 2022. [Online]. Available: https://www.agilealliance.org/agile101

[23] Cucumber and S. Software, "What is bdd?" 2019. [Online]. Available: https://cucumber.io/docs/bdd/

[24] K. Petersen, C. Wohlin, and D. Baca, "The waterfall model in large-scale development," pp. 386 – 388, 2009. [Online]. Available: https://www.diva-portal.org/smash/record.jsf?pid=diva2:835760

[25] A. Alliance, "Behavior driven development (bdd)," 2022. [Online]. Available: https://www.agilealliance.org/glossary/bdd/#q=~(infinite~false~filters~(postType~(~'page~'post~'aa_book~'aa_event_session~'aa_experience_report~'aa_glossary~'aa_research_paper~'aa_video)~tags~(~'bdd))~searchTerm~'~sort~false~sortDirection~'asc~page~1)

[26] ——, "Test-driven development (tdd)," 2022. [Online]. Available: https://www.agilealliance.org/glossary/tdd/#q=~(infinite~false~filters~(postType~(~'page~'post~'aa_book~'aa_event_session~'aa_experience_report~'aa_glossary~'aa_research_paper~'aa_video)~tags~(~'tdd))~searchTerm~'~sort~false~sortDirection~'asc~page~1)

[27] I. Z. Schlueter and T. Brassie, "A test-anything-protocol library for javascript," vol. Why Tap? [Online]. Available: https://node-tap.org/

[28] F. . M. Platforms, "Reactjs," vol. 2022. [Online]. Available: https://reactjs.org/

[29] C. Nakazawa, “Jest 11.0,” vol. April 2016, Apr. 2016. [Online]. Available: https://jestjs.io/blog/2016/04/12/jest-11

[30] “What is front-end development?” 2022. [Online]. Available: https://www.theguardian.com/help/insideguardian/2009/sep/28/blogpost

[31] R. H. II, “Supporting jest open source,” vol. June 27, 2018. [Online]. Available: https://jestjs.io/blog/2018/06/27/supporting-jest-open-source

[32] T. Seckinger, “New defaults for jest, 2021 edition,” vol. May 25, 2021. [Online]. Available: https://jestjs.io/blog/2021/05/25/jest-27#flipping-defaults

[33] “Jest, delightful testing framework,” vol. As of version 27.2. [Online]. Available: https://jestjs.io/

[34] Y. Gong, F. Gu, K. Chen, and F. Wang, “The architecture of micro-services and the separation of frond-end and back-end applied in a campus information system,” in *2020 IEEE International Conference on Advances in Electrical Engineering and Computer Applications(AEECA)*, 2020. doi: 10.1109/AEECA49918.2020.9213662 pp. 321–324.

[35] M. Wubben, “Mark wubben, novemberborn on github,” 2022. [Online]. Available: https://github.com/novemberborn

[36] S. Sorhus, “Sindre sorhus, sindresorhus on github,” 2022. [Online]. Available: https://github.com/sindresorhus

[37] K. B. P. Kinoshita, J. Kingston, and M. Layman, “Test anything protocol,” Apr. 2022. [Online]. Available: https://testanything.org/

[38] L. Wall, “Perl is a highly capable, feature-rich programming language with over 30 years of development.” 2022. [Online]. Available: https://www.perl.org/

[39] ——, “Larry wall’s very own home page,” 2022. [Online]. Available: http://www.wall.org/~larry/

[40] ——, “Perl is a highly capable, feature-rich programming language with over 30 years of development.” 2022. [Online]. Available: https://www.perl.org/about.html

[41] J. Halliday, “Tap-producing test harness for node and browsers,” 2022. [Online]. Available: https://github.com/substack/tape

[42] O. Foundation, “About node.js,” 2022. [Online]. Available: https://nodejs.org/en/about/

[43] E. Bidelman, “Getting started with headless chrome,” 2017. [Online]. Available: https://developers.google.com/web/updates/2017/04/headless-chrome?hl=en

[44] T. M. Foundation, “About javascript.” [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript

[45] I. Z. Schlueter, “About isaac z. schlueter.” [Online]. Available: https://izs.me/

[46] ——, “Github releases,” vol. 2021, Mar. [Online]. Available: https://github.com/tapjs/node-tap/releases

[47] MongoDB, “Mern stack explained,” vol. 2021. [Online]. Available: https://www.mongodb.com/mern-stack

[48] ——, “About us - our story | mongodb,” vol. 2022. [Online]. Available: https://www.mongodb.com/company

[49] O. Foundation, “Express | node js web application,” vol. 2022. [Online]. Available: https://expressjs.com/

[50] MongoDB, “What is mongodb?” vol. 2021. [Online]. Available: https://www.ibm.com/cloud/learn/mean-stack-explained

[51] E. International, “Introducing json,” 2017. [Online]. Available: https://www.ecma-international.org/publications-and-standards/standards/ecma-404

[52] I. C. E. . IBM, “Mean stack explained,” may 2019. [Online]. Available: https://www.mongodb.com/mern-stack

[53] Google, “Angular,” 2022. [Online]. Available: https://angular.io/

[54] A. Hermansson, “Analytic hierarchy process – en kritisk genomgång,” vol. 2014, Magisterprogrammet i Besluts-, risk- och policyanalys, p. 1, 2014. [Online]. Available: https://www.diva-portal.org/smash/get/diva2:726909/FULLTEXT01.pdf

[55] C. D. Foundation, “Thomas l. saaty.” [Online]. Available: http://www.creativedecisions.org/about/ThomasLSaaty.php

[56] T. L. Saaty, “The analytic hierarchy process: planning, priority setting, resource allocation,” vol. 1980 McGraw-Hill, 1980.

[57] A. Hermansson, “Analytic hierarchy process – en kritisk genomgång,” vol. 2014, Magisterprogrammet i Besluts-, risk- och policyanalys, pp. 4–5, 2014. [Online]. Available: https://www.diva-portal.org/smash/get/diva2:726909/FULLTEXT01.pdf

[58] T. L. Saaty, “The analytic hierarchy process: planning, priority setting, resource allocation,” vol. 1980 McGraw-Hill, p. 21, 1980.

[59] A. Hermansson, “Analytic hierarchy process – en kritisk genomgång,” vol. 2014, Magisterprogrammet i Besluts-, risk- och policyanalys, pp. 8–11, 2014. [Online]. Available: https://www.diva-portal.org/smash/get/diva2:726909/FULLTEXT01.pdf

[60] N. Andersson and A. Ekholm, “Vetenskaplighet – utvärdering av tre implementeringsprojekt inom it bygg och fastighet,” vol. IT Bygg och Fastighet 2002, p. 49, 2002. [Online]. Available: https://www.lth.se/fileadmin/projekteringsmetodik/research/Other_projects/utvarderingver1.pdf

[61] D. of Philosophy | McGill University, “Mario bunge | department of philosophy, mcgill university,” 2022. [Online]. Available: https://www.mcgill.ca/philosophy/people/emeritus-faculty/bunge

[62] N. Andersson and A. Ekholm, “Vetenskaplighet – utvärdering av tre implementeringsprojekt inom it bygg och fastighet,” vol. IT Bygg och Fastighet 2002, p. 17, 2002. [Online]. Available: https://www.lth.se/fileadmin/projekteringsmetodik/research/Other_projects/utvarderingver1.pdf

[63] ——, “Vetenskaplighet – utvärdering av tre implementeringsprojekt inom it bygg och fastighet,” vol. IT Bygg och Fastighet 2002, pp. –3, 2002. [Online]. Available: https://www.lth.se/fileadmin/projekteringsmetodik/research/Other_projects/utvarderingver1.pdf

[64] L. University, “Anders ekholm,” 2022. [Online]. Available: https://portal.research.lu.se/sv/persons/anders-ekholm

[65] N. Olsson. [Online]. Available: https://www.kth.se/profile/nicolss

[66] N. Ockelberg. [Online]. Available: https://www.kth.se/profile/ockel

[67] N. Olsson and N. Ockelberg, “Performance, modularity and usability, a comparison of javascript frameworks,” pp. 1–83, 2020.

[68] B. Satrom, “Choosing the right javascript framework for your next web applicationby brando,” vol. Progress 2018, pp. 6, 24–25, 2018. [Online]. Available: https://www.telerik.com/docs/default-source/whitepapers/telerik-com/choose-the-right-javascript-framework-for-your-next-web-application_whitepaper.pdf

[69] ——, “About brandon satrom,” 2022. [Online]. Available: https://www.telerik.com/blogs/author/brandon-satrom

[70] C. university, “Content analysis,” 2022. [Online]. Available: https://www.publichealth.columbia.edu/research/population-health-methods/content-analysis

[71] Momentive, “Surveymonkey, an online survey service,” 2022. [Online]. Available: https://www.surveymonkey.com/

[72] A. Hermansson, “Analytic hierarchy process – en kritisk genomgång,” vol. 2014, Magisterprogrammet i Besluts-, risk- och policyanalys, pp. 14–20, 2014. [Online]. Available: https://www.diva-portal.org/smash/get/diva2:726909/FULLTEXT01.pdf

[73] V. Kristen, “Funktionella och icke funktionella krav: Identifierar de behov som finns för en specifik produkt,” 2021. [Online]. Available: https://projektledning.se/funktionella-och-icke-funktionella-krav/

[74] “Jest expect assertions,” Apr. 2022. [Online]. Available: https://jestjs.io/docs/expect

[75] B. Satrom, “Choosing the right javascript framework for your next web applicationby brando,” vol. Progress 2018, p. 5, 2018. [Online]. Available: https://www.telerik.com/docs/default-source/whitepapers/telerik-com/choose-the-right-javascript-framework-for-your-next-web-application_whitepaper.pdf

[76] N. Nagappan, E. M. Maximilien, T. Bhat, and L. Williams, “Realizing quality improvement through test driven development: results and experiences of four industrial teams,” vol. Empir Software Eng

(2008) 13:289–302, 2008. doi: 10.1007. [Online]. Available: https://www.microsoft.com/en-us/research/wp-content/uploads/2009/10/Realizing-Quality-Improvement-Through-Test-Driven-Development-Results-and-Experienc tdd.pdf

[77] G. Aroush, "Txt file with how the scores for frameworks were decided," 2022. [Online]. Available: https://gits-15.sys.kth.se/aroush/Exjobb/blob/main/comparing_Based_On_Citeria.txt

[78] J. A. ALONSO and M. M. T. LAMATA, "Consistency in the analytic hierarchy process: A new approach," p. 450, 2005. [Online]. Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.108.4785&rep=rep1&type=pdf

[79] T. L. Saaty, "Fundamentals of decision making and priority theory with the analytic hierarchy process," vol. RWS publications Pittsburgh, p. 85, 1994.

[80] A. Hermansson, "Analytic hierarchy process – en kritisk genomgång," vol. 2014, Magisterprogrammet i Besluts-, risk- och policyanalys, 2014. [Online]. Available: https://www.diva-portal.org/smash/get/diva2:726909/FULLTEXT01.pdf

[81] ——, "Analytic hierarchy process – en kritisk genomgång," vol. 2014, Magisterprogrammet i Besluts-, risk- och policyanalys, p. 14, 2014. [Online]. Available: https://www.diva-portal.org/smash/get/diva2:726909/FULLTEXT01.pdf

[82] ——, "Analytic hierarchy process – en kritisk genomgång," vol. 2014, Magisterprogrammet i Besluts-, risk- och policyanalys, pp. 26–29, 2014. [Online]. Available: https://www.diva-portal.org/smash/get/diva2:726909/FULLTEXT01.pdf

[83] ——, "Analytic hierarchy process – en kritisk genomgång," vol. 2014, Magisterprogrammet i Besluts-, risk- och policyanalys, pp. 28–29, 2014. [Online]. Available: https://www.diva-portal.org/smash/get/diva2:726909/FULLTEXT01.pdf

[84] R. Whitaker, "Criticisms of the analytic hierarchy process: Why they often make no sense," vol. 2007, Mathematical and Computer Modelling, pp. 948–949, 2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0895717707001033

TRITA-EECS-EX-2022:677