Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 02, NCRTCS - 2026

A Decision Support Model for Selecting Front-End Frameworks Based on Project and Team Constraints

Shubham Sunil Chandgude
Dept. of Computer Science, Nowrosjee Wadia College (Autonomous), Pune, Maharashtra, India

Sanket Arun Bhagat
Dept. of Computer Science, Nowrosjee Wadia College (Autonomous), Pune, Maharashtra, India

Sahil Gogawale
Dept. of Computer Science, Nowrosjee Wadia College (Autonomous), Pune, Maharashtra, India

Abstract **-** The selection of a suitable front-end framework will have a substantial effect on web application performance, scalability, maintainability, and development success. Currently, developer familiarity with a certain framework or general industry trends often affects the selection of that framework; however, there is little formal systematic process for evaluating said framework before/while designing or developing an application. To assist project managers and/or developers in making informed decisions related to front-end framework selection, this study introduces a structured multi-criteria decision-making (MCDM) model that integrates quantitative performance benchmarks with qualitative developer experience and organizational constraints to evaluate four popular front-end frameworks (React, Angular, Vue, and Svelte) based upon established CRUD application performance benchmarks and developer surveys. These results indicate that Svelte has superior runtime performance and bundle size compared to the other frameworks, Vue offers the best balance of usability and performance compared to the other frameworks, React has stronger support from external libraries/services, and Angular received the highest total score for enterprise-level development. The decision support model proposed in this paper provides a mechanism for developers and organizations to make more informed front-end framework selection decisions based upon individual project considerations**.**

Keywords—Front-End Framework, Angular, React, Vue, Svelte, Decision Support System, Performance Benchmark, Developer Experience, MCDM

I. INTRODUCTION

Modern web applications rely heavily on front-end frameworks for many of the application's performance, scalability, usability and long-term maintainability aspects [2]. As web applications continue to grow in complexity, selecting an appropriate front-end framework will be one of the most important architectural decisions developers will make [10].

Frameworks like React, Angular, Vue, and Svelte differ depending on their architecture, tools, learning curves, and level of maturity within the ecosystem. Therefore, when developers select a framework for production use on a real-world project very often the selection is based on either preference or a desire to use a popular framework rather than through objective analysis of the specific characteristics of the framework.

Past research has focused primarily on technical, isolated metrics such as rendering times and bundle sizes but does not take into account important organizational and/or human-centric constraints such as developer experience, maintenance effort required, scalability of the architecture, and development timelines. As a result, there is a lack of a structured, holistic framework selection approach that can be applied by developers when selecting a framework for use with production systems.

This research paper proposes an organized decision support model that incorporates both quantitative performance benchmarks and qualitative developer-experience metrics with respect to relevant organizational considerations to facilitate informed decision-making regarding framework selection.

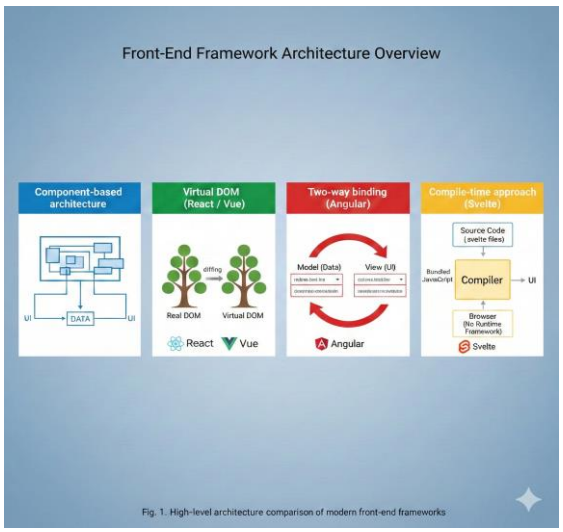
II. BACKGROUND & RELATED WORK

The widespread use of component-based front-end frameworks has increased the ability to create maintainable, scalable, and efficient application development for many developers. Many researchers have performed comparative studies regarding framework comparisons based on the overall performance of the frameworks, the popularity of the frameworks, and the developer's preference for the frameworks [4][5].

Angular has been preferred in most enterprise environments as a result of its highly structured architecture, integrated tooling, and significant support for TypeScript [6]. React, on the other hand, is selected for its flexibility, its ability to implement reusable components, and the broad array of additional utility tools available through the framework's ecosystem. The Vue framework has gained a reputation as easy to learn with well-balanced performance characteristics. The Svelte framework provides the highest level of runtime efficiency by leveraging a compilation process but lacks a mature developer ecosystem.

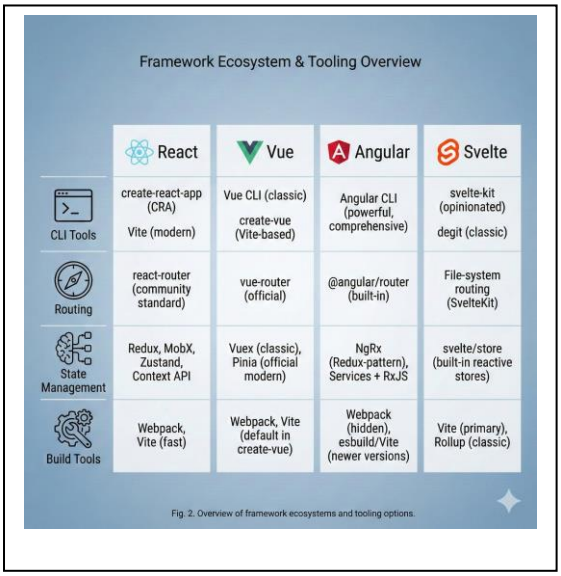
International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 02, NCRTCS - 2026

Figure 1 shows the main architectural differences between the frameworks which were evaluated. While React/Vue both use a component-based Virtual DOM, Angular uses two-way data binding; Svelte has a compiler-based architecture.



The architecture of each of the evaluated frameworks has some distinct characteristics, which have been shown in the chart below. Angular utilizes a two-way data binding technique, while React and Vue both employ a component-based virtual DOM approach for their architectures. Svelte uses a compiling architecture.

Figure 2 compares tool maturity and ecosystem support between CLI Tools, Routing, State Management, and Build Systems for React, Vue, Angular & Svelte.



Despite numerous studies evaluating the various front-end frameworks, they do not evaluate multiple frameworks against each other or take into account the organization's internal constraints when doing so. The research being conducted will improve upon these prior studies by utilizing a

Many others provided us with all the resources and technical assistance we needed to complete this project. In addition, we would like to thank everyone in our department and school for their help during the data analysis and writing of our manuscript.

III. METHODOLOGY

The research consists of four phases to assist decision making:

1. Developing a framework
2. Measuring performance against other frameworks
3. Evaluating developer experience
4. Conducting multi criteria decision analysis

Identical CRUD (Create, Read, Update, and Delete) applications were created using the following four JavaScript frameworks – Angular, ReactJS, VueJS, and Svelte, taking into consideration the official recommendations or best practice for each framework.

Figure 3 shows the workflow diagram shows how the frameworks compare based on Implementation of Framework, Measurement of Performance, Developers Survey and Multi-Criteria Decision-Support Analysis.

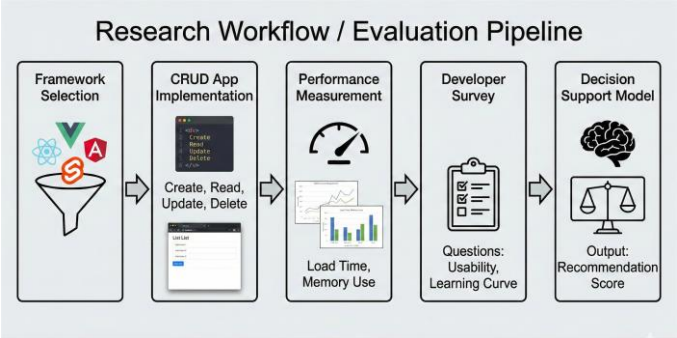


Fig. 3. Research workflow for front-end framework evaluation

Performance Metrics

To evaluate performance, the following metrics were used and evaluated using an industry standard tool (Lighthouse):

1. First Contentful Paint (FCP)
2. Time to Interactive (TTI)
3. Bundle size
4. Total Performance Score

Developer Experience Metrics

To evaluate developer experience, both surveys and direct observation of implementation were used, measuring:

1. Learning curve to get started with a framework
2. Speed of development with the framework
3. Amount of effort spent on debugging
4. Tooling and support from the framework ecosystem

All of the above data was normalized so they could be entered into the multi-criteria decision model [11].

IV. DATA COLLECTION AND PREPROCESSING

Multiple benchmarks run resulted in obtaining performance metrics and developer responses, which was the primary data of this study. Established sources of performance metric data included several categories of academic publications, industry reports, and other types of official documentation for each framework.

To minimize variability affected by environmental conditions, averaged performance results across runs were aggregated into

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 02, NCRTCS - 2026

the following calculated performance metrics: usability (measured by a defined set of descriptive characteristics), productivity (measured using historical project data), and maintenance effort (using the estimated hours that would take to maintain respective frameworks). Scoring systems for user survey responses were also established to ensure fairness in comparison consistency across respective frameworks.

V. RESULTS AND ANALYSIS

This is a summary of performance benchmarks done across the different frameworks. The performance benchmarks showed that there were differentiations between the frameworks for performance.The best overall performing framework was Vue. Svelte outperformed all frameworks on runtime efficiency and has the fastest First Contentful Paint (FCP). React has consistent and stable performance. Angular has the slowest initial render primarily due to the size of the bundles generated and the number of runtime dependencies.In the area of First Contentful Paint (FCP), the fastest performing framework was Svelte primarily due to compiler optimizations and not using a virtual DOM. Following Svelte was Vue, which rendered efficiently/reactively. Then came React with moderate FCP values. Finally, Angular was the slowest performing framework with FCP primarily due to the size of the framework as indicated by the bundle size.The framework with the fastest Time to Interactive (TTI) was Vue. Svelte and React demonstrated reasonable TTI values. Angular was the slowest performing framework for TTI due to the amount of JavaScript that must be executed during initialization.When analyzing bundle sizes as they relate to performance, it can be seen that Vue generates the greatest performance, and thus the smallest bundle size, while Angular generates the largest bundle size and thus has the slowest FCP and TTI. Because of this, the size of bundles created from built code has a significant impact on performance, especially for those on low-bandwidth or unreliable Internet connectivity.

Figure 4 shows Angular, React, Svelte, Vue – Benchmarks for First Contentful Paint (FCP), Time to Interactive (TTI), accessibility and Bundle Size metrics summarized in a table.
In terms of accessibility and SEO scores, all frameworks received passing scores. In terms of SEO, Angular scored the best out of the four. All frameworks received similar passing scores for Accessibility and SEO.

Performance	FCP (s)	TTI (s)	Accessibility	Bundle Size (KiB)
Angular	57.0	7.8	100.0	2263.0
React	63.0	5.0	100.0	1358.0
Svelte	71.0	2.6	89.0	1463.0
Vue	78.0	4.6	100.0	558.0

Fig. 4: Performance and Bundle Size Comparison of Front-End Frameworks

Figure 5 depicts the distributions of performance metrics for each of the four frameworks. The FCP, TTI and bundle size metrics were all taken from Lighthouse performance measurements.

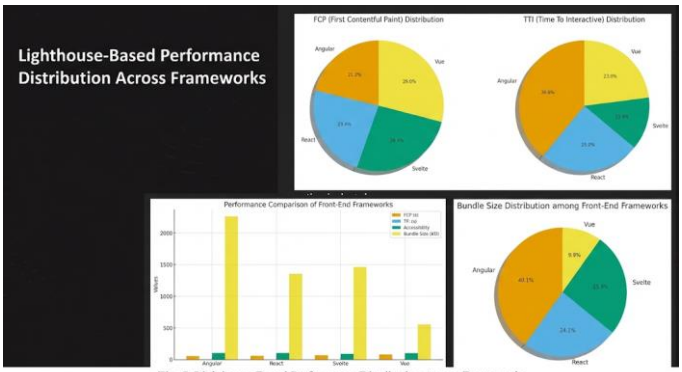


Fig. 5. Lighthouse-Based Performance Distribution Across Frameworks

Comparative Decision Matrix

Developers' Framework Compa rational Decision Matrix has been created using the compare and normalized framework - developer experience and ecosystem criteria. Each Framework was rated against three major sections/areas:

1. Technical Performance (FCP, TTI, run-time efficiency, and bundle size)
2. Developer Experience (learning curve, speed of development, and debugging effort)
3. Ecological and Organizational Support (tool maturity, community support, scalability, and maintainability)

Table: Framework Performance Scores

Framework	Technical	Ecosystem	Organizational	Weighted Score	Rank
Angular	8.5	7.5	9.0	**8.23**	1st
React	7.0	9.0	8.0	**8.05**	2nd
Vue	8.0	8.0	8.0	**8.00**	3rd
Svelte	9.0	6.0	6.0	**7.05**	4th

Table 1. Framework Performance Scores

Above Table 1 shows the final ranking of the various frameworks based on their evaluation scores across all criteria (Technical/Ecosystem/Organizational).
Weights were placed on each of the three sections according to the importance of each one when building an industrial web application. The Angular, React, Vue, and Svelte aggregated weighted scores were calculated.

Figure 6 shows Angular provides the best overall score due to their high degree of support from their organization as well as consistent tech performance across all areas of functionality, while VueJS & React show consistent performance and competition across multiple areas of functionality.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 02, NCRTCS - 2026

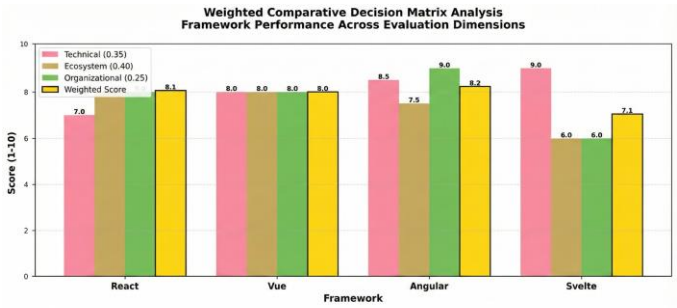


Fig. 6. Weighted Comparative Decision Matrix Analysis - Framework Performance Across Evaluation Dimensions

According to the Results of the Developer's Framework Comparational Decision Matrix, Angular had the highest aggregated score, followed by React and Vue, as shown in figure 6. Svelte had a very high technical score but because of the lower levels of ecosystem and organizational maturity and support, it had a lower overall score [14][15].

In conclusion, these Results show that there is no single framework that is the best in all areas of comparison. However, each framework has its unique advantages and disadvantages based upon the area of comparison [12]. Angular is the best framework to use when building large and/or enterprise-level applications where a strong architectural structure and tooling support is required. React is the better framework for projects where flexibility and the ability to adapt to a constantly-changing environment will be the primary concern, due to the size of its available tooling ecosystem. Vue is the best and most balanced framework of the four frameworks when considering the user experience of the developer versus their technical performance. Svelte is the highest-performing framework of the four for the lightweight and highly-performance-critical applications.

VI. DEVELOPER EXPERIENCE EVALUATION

Based on developer experience analysis:

- Svelte is the easiest framework to learn with little boilerplate required.
- Vue has solid documentation and provides a good balance of usability and functionality.
- React has an extensive ecosystem, requiring more tooling decisions.
- Angular has more tools to work with but has the steepest learning curve and most complex configurations of all frameworks.

VII. DECISION SUPPORT MODEL

A decision matrix considering performance factors, ecosystem factor and org. factors was developed and scored to rank Angular, React, Vue and Svelte in order from highest overall weighted score to lowest [11][9][8][7].

- Angular received the highest overall score making it well suited for both large scale and enterprise use.
- React received a high ranking due to its strong ecosystem.
- Vue exhibited an even level across all criterion.
- Svelte received low marks due to its small ecosystem but received good marks for its technical performance.

Gini Index Dispersion Analysis

Gini Index dispersion analysis measured performance balance across all criteria. Vue exhibited the lowest level of dispersion which was indicative of consistent performance. Svelte exhibited high levels of specialization in technical performance while Angular exhibited high levels of suitability for Organizations.

Figure 7 demonstrates the relative balance of performance between the evaluation criteria; lower values indicate a framework has consistently performed well across all criteria.

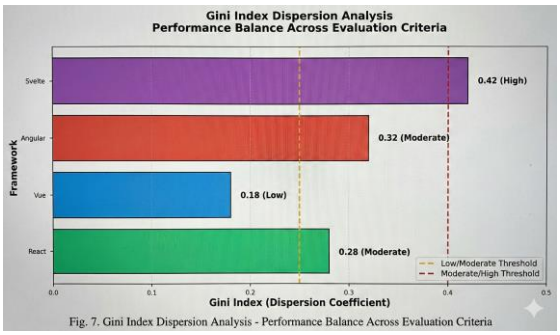


Fig. 7. Gini Index Dispersion Analysis - Performance Balance Across Evaluation Criteria

Decision Tree Matrix

Figure 8 shows the decision tree's recommendations are the frameworks to use based on three criteria: size of the project, learning curve, and how well a framework meets project performance requirements.

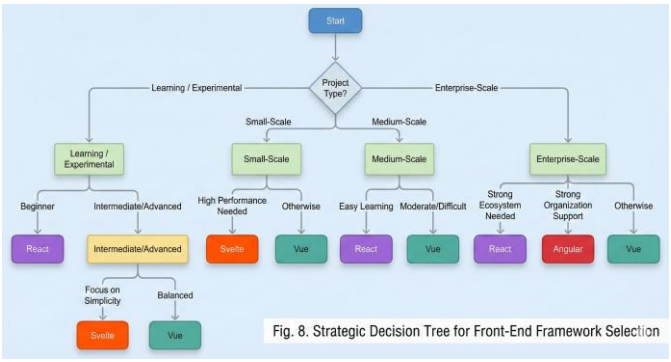


Fig. 8. Strategic Decision Tree for Front-End Framework Selection

VIII. CONCLUSION

This research finds no one front-end framework is optimal for all use cases; front-end framework selection should be based on both the needs of the project, team skills and company restrictions.

For efficient, lightweight applications Svelte is a good choice.

Vue has the best combination of usability and performance.

React has strong ecosystem-agnosticism, which results in a framework that can be adapted for a longer period of time.

Angular is most appropriate for enterprise-level scale applications that require structured architecture.

The multi-criteria decision-support model presented here provides a practical and systematic method for selecting front-end frameworks based on actual constraints.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 02, NCRTCS - 2026

IX. ACKNOWLEDGEMENT

The authors would like to express their heartfelt appreciation to **Dr. Reena Bharathi** for her assistance, motivation and constructive criticism during this project. The authors would like to also acknowledge the faculty and other members of the department for their aid during the entire process of research: from the initial implementation, to data collection and analysis, and even through the writing and publishing phases

X. REFERENCES

[1] A. Strode and S. L. Huff, "Agile vs. Waterfall - A Decision Model for Choosing the Best Project Management Methodology," 2021 Link: https://www.researchgate.net/publication/349525663

[2] S. Usmani, Decision Support Systems in Software Development - A Comparative Study, 2022 Master's Thesis, Tampere University, Finland Link: https://www.diva-portal.org/smash/get/diva2:1758858/FULLTEXT01.pdf

[3] R. Bharathi et al., "Decision Support Models for Technology Selection in Software Engineering," UC eScholarship, 2020 Link: https://escholarship.org/content/qt76j9q5pk/qt76j9q5pk_noSplash_8c1c25b1677c4cc9708f5ce7454d52da.pdf

[4] S. Choudhary et al., "Framework Selection in Modern Front-End Development," International Journal of Engineering Trends and Technology, vol. 9, no. 2, 2021 Link: https://iaeme.com/MasterAdmin/Journal_uploads/IJETR/VOLUME_9_ISSUE_2/IJETR_09_02_027.pdf

[5] S. Chhetri, Performance Evaluation of Modern JavaScript Frameworks, 2023 Bachelor's Thesis, Metropolia University of Applied Sciences Link: https://www.theseus.fi/bitstream/10024/865488/2/Khati%20Chhetri_Sanjay.pdf

[6] A. Patel et al., "Analysis of Front-End Frameworks and Libraries Used in Web Development," International Journal for Research in Applied Science and Engineering Technology (IJRASET), 2022 Link: https://www.ijraset.com/research-paper/analysis-of-the-font-end-frameworks-and-libraries-in-web-development

[7] R. K. Verma et al., "Decision-Making Support System for Team Projects Using the MERN Stack," International Journal of Professional Research in Engineering and Management Studies (IJPREMS), 2023 Link: https://www.ijprems.com/ijprems-paper/a-decision-making-support-system-for-team-projects-using-mern-stack

[8] J. Li et al., "Multi-Criteria Decision-Making Models in Software Engineering - A Systematic Review," Journal of Systems and Software, 2024 Link: https://pmc.ncbi.nlm.nih.gov/articles/PMC10850423/

[9] A. Ali, A Benefits Realisation Decision Support System for Front-End Development, 2021 Master's Thesis, University of Huddersfield Link: https://pure.hud.ac.uk/en/studentTheses/a-decision-support-system-for-benefits-realisation-in-front-end-d/

[10] A. Usmani, Comparative Evaluation Models in Software Architecture Selection, 2023 Master's Thesis, Tampere University Link: https://trepo.tuni.fi/bitstream/handle/10024/152993/UsmaniAsbah.pdf Citation:

[11] L.Wang.et.al., Decision Support Systems Using Multi-Criteria Approaches, 10(2), (2020), Buildings. Link: https://www.mdpi.com/2075-5309/10/2/34

[12] P. Sharma, et al. Comparative Study of Front-End Frameworks; 2022; IJARSCT. Link: https://ijarsct.co.in/Paper4892.pdf

[13] Netlify. 5 Considerations When Choosing a Front-End Framework; 2023. Link: https://www.netlify.com/blog/5-considerations-when-choosing-a-frontend-framework/

[14] Sencha. Choosing the Best Front-End Framework; 2022. Link: https://www.sencha.com/blog/front-end-frameworks-choosing-best-option/

[15] MindInventory; Best Front-End Frameworks; 2023. Link: https://www.mindinventory.com/blog/best-frontend-frameworks/