Microfront-End: Systematic Mapping

Luiz Felipe Cirqueira dos Santos[1][a], Marcus Vinicius Santana Silva[1][b],
Shexmo Richarlison Ribeiro dos Santos[1][c], Fábio Gomes Rocha[1][d]
and Elisrenan Barbosa da Silva[2][e]
[1]*Postgraduate Program in Computer Science - PROCC/UFS, Federal University of Sergipe, Av. Mal. Cândido Rondon, Rosa Elze, 1861, São Cristóvão, Brazil*
[2]*Computer, Estácio University Center of Sergipe, R. Teixeira de Freitas, Salgado Filho, 10, Aracaju, Brazil*
luizfelipecirqueira@outlook.com, {marcus.vinicius.3007, Shexmor, gomesrocha, elisrenan}@gmail.com

Keywords: Microfrontend, Micro Front-End, Micro-Frontend.

Abstract: In the context of backend development, adopting microservices has brought new challenges to frontend integration, leading to the development of microfrontends. Monolithic frontends go against the principles of microservices specialization, requiring a more modular approach. This study aims to characterize the adoption of micro frontends within microservices. To achieve this, a systematic mapping was conducted across six databases, selecting 32 articles that identified 11 tools for managing applications with micro frontends. The study discusses the increase in complexity and the contexts and conditions for applying micro frontend implementation models. As a contribution, this study outlines the main paths for adopting micro frontends within a microservices architecture, providing valuable insights for software engineering and promoting better integration between the backend and front end. Additionally, it describes the characteristics of micro frontends and the advantages and disadvantages of their adoption, highlighting security concerns, their impact on team collaboration, and how they may affect the organizational structure and team development processes.

1 INTRODUCTION

Developing the presentation layer of a modern web application has become a crucial task for industrial companies. Development teams constantly seek new ways to build, deploy, and maintain applications effectively so that companies can deliver value quickly and efficiently to their customers (Peltonen et al., 2021). Modern applications are built with feature-rich and powerful frontends. The more functionalities there are, the harder it becomes to maintain the front end, significantly when different teams develop it (Taibi and Mezzalira, 2022).

New frontend frameworks are continuously introduced to the market, providing developers with many valid options to build robust and feature-rich web applications, such as Single Page Applications (SPA), Server-Side Rendering (SSR) applications, or static HyperText Markup Language (HTML) files combined with a web page. However, most of them end up being monolithic. Therefore, the application's client-side grows, making its development challenging, especially when teams need to edit the same frontend application (Peltonen et al., 2021).

Micro frontends were introduced in 2016 to solve this problem. Micro frontends allow the front end to be broken down into individual and semi-independent micro-applications, separating the front end's business logic and creating independent services that interact with each other (Taibi and Mezzalira, 2022).

Micro frontends share microservices' main principles, benefits, and challenges (Geers, 2020): both are modeled around business domains, hiding implementation details from each other. Each team should own its microservice (backend) and the related frontend, enabling decentralized decision-making and independent deployment. However, micro frontends also introduce some disadvantages, such as the risk of communication overhead if the system is not well-designed and reviewed as the business grows, potential performance issues when project vendors are not carefully considered (e.g., for SPA), and the compromise of the user experience when the governance be-

[a] https://orcid.org/0000-0003-4538-5410
[b] https://orcid.org/0009-0000-9211-5259
[c] https://orcid.org/0000-0003-0287-8055
[d] https://orcid.org/0000-0002-0512-5406
[e] https://orcid.org/0000-0001-8890-9718

Santos, L., Silva, M., Santos, S., Rocha, F. and Barbosa da Silva, E.
Microfront-End: Systematic Mapping.
DOI: 10.5220/0013015400003825
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 20th International Conference on Web Information Systems and Technologies (WEBIST 2024)*, pages 119-130
ISBN: 978-989-758-718-4; ISSN: 2184-3252
Proceedings Copyright © 2024 by SCITEPRESS – Science and Technology Publications, Lda.

hind a design system is not well thought out (Peltonen et al., 2021).

This study focuses on conducting a systematic literature mapping to provide answers on the characterization of micro frontends, their positives and negatives, and the paths followed in current research by analyzing the entire population of articles in the chosen databases that address micro frontends. The rest of this paper is structured as follows. Section 2 provides the context for this work, discussing microservices, frontends, and micro frontends and offering a comparison. Section 3 discusses related work on micro frontends. Section 4 addresses the methodology employed in this paper. Section 5 describes and provides the results, and Section 6 discusses the proposed questions. Section 7 contains the final considerations.

2 THEORETICAL FRAMEWORK

In recent years, microservices have gradually become a trend in software architecture design. Microservices are generally considered a collection of independent services, each being an independent process that focuses on a single responsibility and functionality, decoupling dependencies between services. This idea of division was transferred to the front, but it requires a different understanding and application (Wang et al., 2020).

Having a well-defined architecture to maintain scalability and organization in the application is undeniable, aiming to achieve development efficiency and technological heterogeneity. One of the most commonly used forms of architecture is monolithic architecture. However, the difficulty of scalability and code reuse is evident due to the high coupling between its parts (Perlin et al., 2023).

With the awareness of these scalability and reuse impediments, the need arises to adopt service-oriented applications. Microservices have been widely adopted to address the complexity of monolithic systems in the backend due to their modularity and independence (Perlin et al., 2023).

Micro frontends extend the concepts of Microservices to the frontend side of the application. This transforms monolithic web applications from a single codebase architecture into an application that combines multiple small frontend applications into one. These independent applications can be executed, developed, and deployed independently. The ability for independent development and deployment allows development teams to build isolated, loosely coupled services. The idea behind micro frontends is to treat a web application as a combination of functionalities or business subdomains. Each team should handle only one domain (Peltonen et al., 2021).

Frontend monoliths introduce horizontal layers on the frontend side of the application, but micro frontends aim to divide the application vertically. Each vertical slice is built entirely from scratch and serves a specific business domain or functionality. Each development team can be technologically agnostic with micro frontends and decide what technology stack to use. Teams can update or change the stack without coordinating with other teams (Peltonen et al., 2021).

3 RELATED WORK

As mentioned, this study aims to characterize micro frontends, their strengths and weaknesses, and the current research paths. In the work of Taibi and Mezzalira (Taibi and Mezzalira, 2022), it is reported that micro frontends emerged from the evolution of software architectures and shared the fundamental principles, benefits, and problems of microservices. They are modeled around business domains, hiding implementation details between them, and each team has its backend and related frontend service, bringing decentralization of decisions and independent deployment.

In their work, Bian, Ma, Zou, and Yue (Bian et al., 2022) report that systems are becoming increasingly complex and that extensive web frontend projects must be completed with more than one development team. They introduce a concept similar to microservices in web applications, which can help traditional monolithic web applications overcome the existing dilemmas associated with monolithic applications.

Mannisto, Tuovinen, and Raatikainen (Männistö et al., 2023) present the experiences of a small team from the company Visma in transforming its monolithic User Interface (UI) into a micro-frontend solution. They discuss the motivations and their essential design and implementation decisions, including an architectural assessment. The main conclusions are that (1) the motivations for micro frontends are related to improving customer-specific reconfigurability and reducing associated costs rather than enabling team independence, (2) the APIs (Application Programming Interfaces) provided by web standards—and the Web Component API based on them—offered a competitive alternative to JavaScript frameworks, avoiding many issues induced by the frameworks, and (3) small organizations without a large number of resource teams can benefit from micro-frontend architectures.

In summary, these works help outline the objectives of the study, such as the characterization of micro frontends, their strengths and weaknesses, and the current research paths, providing theoretical and empirical foundations that contextualize the evolution, benefits, and challenges of micro frontends, essential for the systematic analysis. They help identify common motivations for adopting micro frontends, such as decentralization, team independence, and improvements in configurability and cost. They also point out specific challenges and practical solutions, enriching the analysis and conclusions of the systematic mapping study on micro frontends.

4 METHODOLOGY

This article aimed to characterize, in a structured manner, from a broad perspective, micro frontends. To this end, a systematic mapping was chosen as the research instrument. A systematic mapping is a comprehensive and rigorous review of a field or research topic, allowing for a broad view of the paths taken and their respective gaps (Velásquez-Durán and Ramírez Montoya, 2018) (Keele et al., 2007). The systematic mapping presented in this article follows the guidelines of Kitchenham and Charters [34] and is divided into three phases: planning, execution, and, finally, the communication of results. Thus, we first defined the topic to be researched. Next, we detailed the methodology applied and the selection of databases where the articles were searched. We prepared the questions to be answered by this study and the search string related to the topic, all using a tool called Parsifal.

To ensure the transparency and reproducibility of the study, we followed a detailed protocol for classifying the articles. This protocol included the following steps:

- Definition of Inclusion and Exclusion Criteria
- Article selection
- Classification and Analysis of Articles
- Definition of research questions
- Number of publications per year

Articles that deal, even partially, with micro frontends were included in defining the inclusion criteria. As for the exclusion criteria, we removed duplicate articles, those published as abstracts or expanded summaries, articles not written in English, articles not discussing micro frontends, and secondary studies, as shown in Table 1.

We performed a search using the terms "microfrontend" OR "micro front-end" OR "micro-frontend", without time restrictions, in the ACM, IEEE, ISI Web of Science, Science Direct, Scopus, and Springer Link databases on April 3, 2024. The selected databases are hybrid, being both search engines and bibliometric databases, widely adopted in studies, as they ensure good coverage for systematic reviews, according to (Kitchenham and Brereton, 2013). We identified 107 relevant articles that addressed, even partially, the concept of micro frontends.

Table 1: Criteria.

Inclusion	Articles that deal, even partially, with micro frontends
Exclusion	Duplicate article
	Article published as an abstract or expanded summary
	Articles that are not written in English
	Articles that do not discuss micro frontends
	Secondary study

After removing duplicates and applying inclusion and exclusion criteria, 25 duplicate articles were eliminated, resulting in 82 articles for verification. Of these, 32 were considered relevant, as they dealt with micro frontends in their title and abstract for the study. At the same time, the remaining 50 were excluded for not meeting the established criteria, as shown in Figure 1. Within the stages of this study, we read the articles that met the inclusion and exclusion criteria to find answers to the questions, using structured questions that can guide future research sequences and evaluate the research process (Kitchenham and Brereton, 2013), described in Table 2.

Figure 2 illustrates the annual increase in the number of articles on micro frontends, reflecting this approach's growing interest and adoption. These publication trends provide insights into the evolution and maturity of micro frontends practices, highlighting the development of academic and practical interest in this topic. This information is essential as it allows for identifying research trends, measuring the growth of the adoption of this technology, and understanding periods of significant development and innovation. Furthermore, it helps recognize the increasing relevance of micro frontends in the developer and researcher communities, providing a foundation for future investigations and improvements in the field.

5 RESULTS

This section presents the results of our systematic mapping on micro frontends. Each research question was thoroughly examined to extract significant insights into this architecture's benefits, challenges, and trends. We divided our findings into subthemes to facilitate understanding and visualization of the information. In each subtheme, we will present the answers to the specific questions.

Table 2: Research Questions and Justifications.

Question Number	Question Text	Justification
Q1	Where are micro frontends being applied? What are the conditioning factors for the adoption of micro frontends?	Identify patterns and factors that drive or hinder adoption.
Q2	Which frameworks are most adopted using micro frontends?	Standardize development and find active support.
Q3	How are micro frontends being tested, and what are the best practices to ensure software quality?	Ensure software quality and avoid problems.
Q4	How are micro frontends built, integrated, and delivered to end-users?	Optimize processes and improve efficiency.
Q5	What are the positive aspects of adopting micro frontends?	Demonstrate added value to stakeholders.
Q6	What are the negative aspects of adopting micro frontends?	Anticipate challenges and mitigate risks.
Q7	How do micro frontends impact collaboration between teams within an organization?	Understand the influence on team dynamics and efficiency.
Q8	What specific security challenges are associated with micro frontend architecture?	Ensure data protection and system integrity.
Q9	What tools exist to manage applications with micro frontends when there is an increase in their complexity?	Manage complexity and maintain efficiency.
Q10	How does introducing micro frontends affect a team's organizational structure and development processes?	Adjust organization and processes to the new approach.
Q11	What kinds of results were obtained?	Measure success and identify improvements.
Q12	What types of validation were performed?	Ensure effective implementations and compliance with requirements.

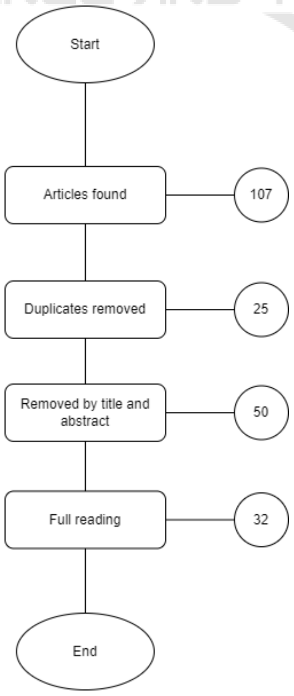

Figure 1: Selection of The Articles.

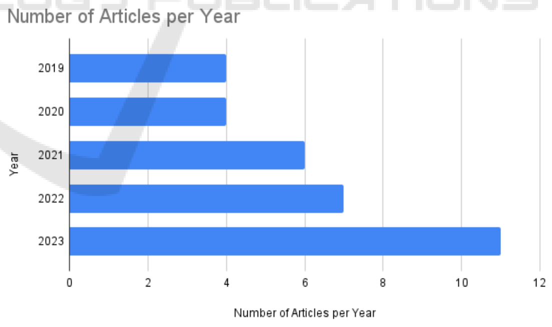

Figure 2: Articles of Year.

5.1 RQ1. Where Are Micro Frontends Being Applied? What Are the Conditioning Factors for Adopting Micro Frontends?

Micro frontends are applied in a variety of contexts, such as the development of complex web applications (Peltonen et al., 2021) (Taibi and Mezzalira, 2022), system architecture engineering (Schäffer et al., 2019), integration with advanced technologies

(Simões et al., 2023), and companies with distributed teams (Männistö et al., 2023). In large-scale projects, such as site portals and enterprise systems (Bian et al., 2022) (Zhang et al., 2023), micro frontends facilitate the integration of various subsystems. In contrast, modular systems, such as industrial installations (Lorenz et al., 2021), offer flexibility and adaptability. The ability to handle distributed development teams (Perlin et al., 2023) and the need to integrate emerging technologies, such as Web 3D (Simões et al., 2023), also drive the adoption of micro frontends.

Factors influencing the adoption of micro frontends include frontend complexity (Taibi and Mezzalira, 2022), the need for agile and independent development (Männistö et al., 2023), demand for flexibility and scalability (Taibi and Mezzalira, 2022), decoupling and reuse of components (Schäffer et al., 2019) (Tosic et al., 2023), independence in updates (Knecht et al., 2020), reduction of maintenance costs, and availability for organizational changes (Taibi and Mezzalira, 2022). The modularity provided by micro frontends allows for handling the increasing complexity of user interfaces (Taibi and Mezzalira, 2022), while independent development capabilities simplify the development process (Schäffer et al., 2019). Flexibility in technology choice and component reuse contribute to a more scalable and efficient architecture (Männistö et al., 2023) (Zhang et al., 2023).

5.2 RQ2. What Are the Most Adopted Frameworks for the Adoption of Micro Frontends?

Based on the mapped articles, as shown in Table 3, we identified that micro frontends are implemented using a variety of frameworks and approaches, with Angular, four articles (Peltonen et al., 2021) (Perlin et al., 2023) (Pavlenko et al., 2020) (Yang et al., 2019), React, four articles (Peltonen et al., 2021) (Perlin et al., 2023) (Schäffer et al., 2019) (Pavlenko et al., 2020), and Vue.js, five articles (Peltonen et al., 2021) (Wang et al., 2020) (Perlin et al., 2023) (Bian et al., 2022) (Pavlenko et al., 2020) standing out for their versatility and robustness in building dynamic interfaces. Tools such as Webpack 5 with Module Federation (Männistö et al., 2023) (Pavlenko et al., 2020) (Petcu et al., 2023) facilitate dynamic module integration, while Single-SPA (Männistö et al., 2023) (Pavlenko et al., 2020) (Petcu et al., 2023) allows combining micro frontends from different frameworks into a single application. Other alternatives include using Luigi, Application Shell [2], and methods such as iFrames, two articles (Taibi and Mezzalira, 2022) (Pölöskei and Bub, 2021), and ESI (Edge Side Includes) (Taibi and Mezzalira, 2022) for micro frontend integration, each adapted to the specific needs and preferences of the project.

Table 3: Frameworks and Approaches for Implementing Micro Frontends.

Framework / Approach	**Number of Articles**
Angular	4
React	4
Vue.js	5
Webpack 5 with Module Federation	3
Single-SPA	3
Luigi	2
Application Shell	1
iFrames	2
ESI (Edge Side Includes)	1

5.3 RQ3. How Are Micro Frontends Being Tested, and What Are the Best Practices to Ensure Software Quality?

The analyzed articles show that micro frontends are tested independently and collectively using unit, integration, and end-to-end (E2E) tests (Mena et al., 2019) (Schäffer et al., 2019) (Zhang et al., 2023). Using tools such as Jest and Mocha, unit tests verify isolated components, while integration tests ensure the joint functionality of micro frontends (Mena et al., 2019) (de Oliveira et al., 2022). E2E tests, using Cypress and Selenium, ensure that the entire application works correctly (Schäffer et al., 2019) (Zhang et al., 2023) (Bühler et al., 2022). To maintain software quality, practices such as test automation and CI/CD (Continuous Integration/Continuous Deployment) pipelines are essential, allowing for the automatic and continuous execution of tests (Yang et al., 2019) (Tilak et al., 2020). Technologies like Docker, NGINX, and Node.js simulate production environments, ensuring the correct functionality of micro frontends (Simões et al., 2023) (Noppadol and Limpiyakorn, 2021).

5.4 RQ4. How Are Micro Frontends Built, Integrated, and Delivered to End Users?

After analyzing the articles, we found that the construction, integration, and deployment of micro frontends to end users are carried out independently by each team, using tools such as Webpack and Module Federation (Bühler et al., 2022), as well as technologies like Docker and NGINX for container hosting and routing (Yang et al., 2019). Component integration is achieved through an application shell architecture or iframes (Simões et al., 2023), allowing for dynamic and modular composition of the user interface. CI/CD pipelines facilitate deployment (Schäffer et al., 2019) and may involve hosting on proprietary servers, cloud services such as Amazon Web Services (AWS), and Content Delivery Networks (CDNs), ensuring continuous and efficient updates (Simões et al., 2023).

5.5 RQ5. What Are the Positive Aspects of Adopting Micro Frontends?

Several positive aspects have been identified in the adoption of micro frontends. These benefits include modularity, component reuse, increased software development, and maintenance efficiency. Technological independence allows teams to use different technology stacks without compromising integration, facilitating both scalability and customization of the application. Additionally, code isolation prevents errors in one component from affecting the rest of the application, promoting greater resilience (Peltonen et al., 2021) (Taibi and Mezzalira, 2022) (Geers, 2020) (Wang et al., 2020) (Perlin et al., 2023) (Bian et al., 2022) (Männistö et al., 2023) (Velásquez-Durán and Ramírez Montoya, 2018) (Mohammed et al., 2022) (Sorgalla et al., 2021) (Knecht et al., 2020) (Mena et al., 2019) (de Oliveira et al., 2022) (Schäffer et al., 2019) (Simões et al., 2023) (Pavlenko et al., 2020) (Petcu et al., 2023) (Yang et al., 2019) (Tosic et al., 2023) (Wanjala, 2022) (Capdepon et al., 2023) (Pölöskei and Bub, 2021) (Zhang et al., 2023) (Lorenz et al., 2021) (Bühler et al., 2022) (Tilak et al., 2020) (Noppadol and Limpiyakorn, 2021) (Zhao et al., 2023) (Nishizu and Kamina, 2022) (Shakil and Zoitl, 2020) (Stefanovska and Trajkovik, 2022).

The modular approach facilitates team collaboration, system scalability, and incremental improvement implementation (de Oliveira et al., 2022). Adopting micro frontends also improves performance and user experience, especially in high-demand scenarios (de Oliveira et al., 2022) (Bühler et al., 2022).

5.6 RQ6. What Are the Negative Aspects of Adopting Micro Frontends?

The articles indicate that adopting micro frontends increases the complexity of applications, requiring careful coordination in integration, routing, and communication between micro frontends, leading to development overhead and performance issues. Managing shared state and UI consistency is also challenging (Peltonen et al., 2021) (Taibi and Mezzalira, 2022) (Geers, 2020) (Wang et al., 2020) (Perlin et al., 2023) (Bian et al., 2022) (Männistö et al., 2023) (Velásquez-Durán and Ramírez Montoya, 2018) (Mohammed et al., 2022) (Sorgalla et al., 2021) (Knecht et al., 2020) (Mena et al., 2019) (de Oliveira et al., 2022) (Schäffer et al., 2019) (Simões et al., 2023) (Pavlenko et al., 2020) (Petcu et al., 2023) (Yang et al., 2019) (Tosic et al., 2023) (Wanjala, 2022) (Capdepon et al., 2023) (Pölöskei and Bub, 2021) (Zhang et al., 2023) (Lorenz et al., 2021) (Bühler et al., 2022) (Tilak et al., 2020) (Noppadol and Limpiyakorn, 2021) (Zhao et al., 2023) (Nishizu and Kamina, 2022) (Shakil and Zoitl, 2020) (Stefanovska and Trajkovik, 2022).

Other challenges include the initial investments in setting up architecture and CI/CD pipelines, ensuring security in a distributed architecture, and managing dependencies and style conflicts. The additional complexity may not offset the benefits for smaller projects, making the approach less advantageous (Taibi and Mezzalira, 2022) (Mohammed et al., 2022).

5.7 RQ7. How Do Micro Frontends Impact Collaboration Between Teams Within an Organization?

Micro frontends significantly impact team collaboration, promoting independence and autonomy by allowing each team to develop and maintain its part of the frontend with their technology choices (Knecht et al., 2020) (Schäffer et al., 2019) (Pavlenko et al., 2020) (Bühler et al., 2022) (Zhao et al., 2023). This increases efficiency, development speed, and team accountability, enhancing engagement (Knecht et al., 2020) (Schäffer et al., 2019) (Pavlenko et al., 2020) (Bühler et al., 2022). However, this approach also presents challenges, such as effective coordination and communication between teams to ensure smooth integration of micro frontends (Knecht et al., 2020) (Schäffer et al., 2019) (Zhao et al., 2023). Fragmenting the monolith into microprojects requires clear team subdivision, which can be complex (Knecht et al., 2020) (Schäffer et al., 2019). Additionally, the initial effort needed to set up architecture and CI/CD pipelines may affect development efficiency and clarity of responsibilities (Knecht et al., 2020) (Schäffer et al., 2019).

5.8 RQ8. What Are the Specific Security Challenges Associated with Micro Frontend Architecture?

Specific security challenges associated with microfrontend architecture, as found in the research, include managing permissions and access, protecting against code injection attacks (such as XSS, Cross-Site Scripting), and ensuring security in communications between micro frontends (Pavlenko et al., 2020). Each micro frontend needs to ensure proper authentication and authorization, which can be complex due to the distributed nature of the architecture (Perlin et al., 2023). Additionally, protection against CSRF (Cross-Site Request Forgery) attacks and the integrity of communication interfaces are essential to prevent data leaks (Pavlenko et al., 2020).

Another significant challenge is managing dependencies between projects, as different micro frontends may use different versions of the same libraries, increasing the attack surface and the potential for security flaws. The additional complexity of managing multiple distributed vulnerability points requires ongoing monitoring, updating, and protecting each component individually while maintaining overall application security (Männistö et al., 2023) (Pavlenko et al., 2020).

5.9 RQ9. What Tools Exist to Manage Applications with Micro Frontends as Their Complexity Increases?

As seen in Table 4, to manage the complexity of applications with micro frontends, several tools are used, such as Webpack 5 with Module Federation for module sharing (Männistö et al., 2023) (Pavlenko et al., 2020) (Petcu et al., 2023), and frameworks like Single-SPA and Luigi (Männistö et al., 2023) (Pavlenko et al., 2020) (Petcu et al., 2023) that facilitate micro frontend integration. Automation and CI/CD tools such as Jenkins, GitLab CI/CD, and Docker and Kubernetes for containers and orchestration are also essential (Schäffer et al., 2019). NGINX is used for

Routing and load balancing, while Node.js with Express and Socket.IO (Schäffer et al., 2019) supports web servers and real-time communication. Monitoring and tracking tools like New Relic, Datadog, and Sentry help with performance analysis and error management.

Table 4: Tools for Managing the Complexity of Applications with Micro Frontends.

Tool	Number of Articles
Webpack 5 with Module Federation	3
Single-SPA	2
Luigi	2
Automation and CI/CD Tools (Jenkins, GitLab CI/CD)	1
Docker	1
Kubernetes	1
NGINX	1
Node.js with Express	1
Socket.IO	1
Monitoring and Tracking Tools (New Relic, Datadog, Sentry)	1

5.10 RQ10. How Does the Introduction of Micro Frontends Affect the Organizational Structure and Development Processes of a Team?

Introducing micro frontends affects the organizational structure and development processes by promoting team autonomy and independence. Teams can be responsible for specific system parts, facilitating a more agile and iterative approach (Peltonen et al., 2021) (Taibi and Mezzalira, 2022) (Geers, 2020). This also allows for greater flexibility and a more granular division of labor, with each team fully owning its business domain (Wang et al., 2020) (Perlin et al., 2023) (Bian et al., 2022). As a result, there is a reduction in code conflicts and an increase in responsiveness to customer needs (Männistö et al., 2023) (Velásquez-Durán and Ramírez Montoya, 2018). However, this shift can also bring challenges, such as learning new technologies, greater coordination between teams, and managing dependencies and complex integrations (Mohammed et al., 2022) (Sorgalla et al., 2021) (Knecht et al., 2020). Additionally, a significant initial effort may be required to set up architecture, CI/CD pipelines, and continuous integration processes (Mena et al., 2019) (de Oliveira et al., 2022). Overall, this approach encourages a component-oriented mindset and promotes collaboration among different functional areas of the organization (Schäffer et al., 2019) (Simões et al., 2023).

5.11 RQ11. What Type of Results Were Obtained?

Based on the articles reviewed during the research, the results obtained were diverse, as shown in Table 5. Most results (14) consisted of specific solutions, prototypes, responses, or judgments (Taibi and Mezzalira, 2022) (Perlin et al., 2023) (Bian et al., 2022) (Mohammed et al., 2022) (Sorgalla et al., 2021)

(Simões et al., 2023) (Pavlenko et al., 2020) (Yang et al., 2019) (Tosic et al., 2023) (Wanjala, 2022) (Noppadol and Limpiyakorn, 2021) (Zhao et al., 2023) (Nishizu and Kamina, 2022) (Shakil and Zoitl, 2020). Analytical models (8) (Peltonen et al., 2021) (Wang et al., 2020) (Sorgalla et al., 2021) (Simões et al., 2023) (Pavlenko et al., 2020) (Yang et al., 2019) (Tosic et al., 2023) (Pölöskei and Bub, 2021) and reports (8) (Peltonen et al., 2021) (Wang et al., 2020) (Sorgalla et al., 2021) (Schäffer et al., 2019) (Petcu et al., 2023) (Yang et al., 2019) (Tosic et al., 2023) (Pölöskei and Bub, 2021) followed as the most common types of results. Additionally, qualitative or descriptive models (5) (Männistö et al., 2023) (Sorgalla et al., 2021) (Mena et al., 2019) (de Oliveira et al., 2022) (Lorenz et al., 2021), procedures or techniques (4) (Perlin et al., 2023) (Mena et al., 2019) (Bühler et al., 2022) (Tilak et al., 2020), empirical models (2) (Sorgalla et al., 2021) (Schäffer et al., 2019), and tools or notations (2) (Peltonen et al., 2021) (Yang et al., 2019) were identified. These data indicate a predominance of practical and applicable solutions, accompanied by various analytical and descriptive models, reflecting the diversity of approaches and results in the literature on micro frontends.

Table 5: Types of Results Obtained in Research on Micro Frontends.

Type of Result	Number of Articles
Specific solutions, prototypes, responses or judgments	14
Analytical models	8
Reports	8
Qualitative or descriptive models	5
Procedures or techniques	4
Empirical models	2
Tools or notations	2

5.12 RQ12. What Validation Methods Are Most Commonly Used in Micro Frontend Studies?

As seen in Table 6, the data on the types of validation used in the articles indicate that evaluation was the most common method, appearing in 8 occurrences (Taibi and Mezzalira, 2022) (Perlin et al., 2023) (Bian et al., 2022) (Männistö et al., 2023) (Mohammed et al., 2022) (Sorgalla et al., 2021) (Knecht et al., 2020) (Mena et al., 2019). Next is the combination of analysis, evaluation, and Experience, used in 5 occurrences (Mohammed et al., 2022) (Sorgalla et al., 2021) (Knecht et al., 2020) (Mena et al., 2019) (de Oliveira et al., 2022). Validation through examples combined with experiences was used in 7 occurrences (Peltonen et al., 2021)(Perlin et al., 2023) (Schäffer et al., 2019) (Simões et al., 2023) (Pavlenko et al., 2020) (Petcu et al., 2023) (Yang et al., 2019). Additionally, analysis and evaluation were used in 4 occurrences (Männistö et al., 2023) (Sorgalla et al., 2021) (Knecht et al., 2020) (Simões et al., 2023) and isolated analysis, which also appeared in 4 occurrences (Schäffer et al., 2019) (Tosic et al., 2023) (Wanjala, 2022) (Capdepon et al., 2023). Evaluation combined with Experience was used in 2 occurrences (Perlin et al., 2023) (Bian et al., 2022) and isolated Experience, which also appeared in 2 occurrences (de Oliveira et al., 2022) (Capdepon et al., 2023). There was also one occurrence of validation that included an extensive combination of analysis, evident assertion, evaluation, Experience, and persuasion (Mena et al., 2019). Finally, combining analysis with examples was used in 3 occurrences [(Schäffer et al., 2019) (Simões et al., 2023) (Wanjala, 2022). These data show that validation in micro front-end studies generally involves a mix of practical evaluations and empirical examples, reflecting a robust and varied approach to verifying proposed solutions.

Table 6: Types of Validation Used in Articles on Micro Frontends.

Type of Validation	Number of Occurrences
Evaluation	8
Combination of Analysis, Evaluation and Experience	5
Examples combined with Experiences	7
Analysis and Evaluation	4
Isolated Analysis	4
Evaluation combined with Experience	2
Isolated Experience	2
Extensive Combination (Analysis, Assertion, Evaluation, Experience and Persuasion)	1
Analysis combined with Examples	3

6 DISCUSSION

The results obtained from the systematic mapping on micro frontends have several significant implications for both academia and Software Engineering professionals. For academia, the insights extracted from the analyzed articles reveal emerging trends and current challenges, guiding future research and contributing to the theoretical evolution of micro frontend architecture. Identifying benefits, such as modularity and technological independence, and challenges, such as integration complexity and security, provides a solid foundation for in-depth studies that can result in new solutions and advancements in the field. The findings offer practical guidance for practitioners on applying and adopting frameworks and best practices for testing and delivering micro frontends. Knowledge about management tools, the impacts on organizational structure, and strategies to overcome specific security challenges are valuable, enabling a more informed and efficient adoption of this architecture.

Through our systematic mapping, we discovered that micro frontends offer a promising approach to web development despite challenges such as integration complexity. Table 7 presents the implications of the systematic mapping results in a clear and concise table containing a synthesized discussion summarizing the essential findings about micro frontends.

Thus, the findings, organised into seven categories, can be highlighted as follows: Tools and frameworks emerge as central themes, highlighting their importance in implementing micro frontends. However, aspects related to quality and security still require further investigation and the development of more robust solutions. Significant gaps are evident, especially in integrating quality assurance tools specific to modular architectures. The inherent complexity of distributed testing, which involves multiple modules developed autonomously, demands a more efficient and automated approach capable of dealing with the peculiarities of this architecture.

Furthermore, there is a pressing need for more research on strategies for migrating legacy systems to micro frontends, a common challenge in companies with large monoliths that are considering this modular approach. Clear guidelines for this transition could pave the way for the widespread adoption of the architecture. Future research should concentrate on developing methodologies that facilitate this migration and minimise the associated risks and costs.

Moreover, there is yet to be a deep understanding of the long-term impact of adopting micro frontends on companies' organisational structures. The modularisation of interfaces and the autonomy of development teams can profoundly transform how teams organise and collaborate. Studying this dynamic and the possible benefits of productivity and organisational efficiency constitutes a significant research gap.Thus, the findings, organised into seven categories, can be highlighted as follows Tools and frameworks emerge as central themes, highlighting their importance in implementing micro frontends. However, aspects related to quality and security still require further investigation and the development of more robust solutions. Significant gaps are evident, especially in integrating quality assurance tools specific to modular architectures. The inherent complexity of distributed testing, which involves multiple modules developed autonomously, demands a more efficient and automated approach capable of dealing with the peculiarities of this architecture.

Additionally, there needs to be more studies on strategies for migrating legacy systems to micro frontends, a common challenge in companies with giant monoliths that wish to adopt the modularity of this approach. Clear guidelines for this transition may allow the widespread adoption of the architecture. Future research should focus on developing methodologies that facilitate this migration and minimize the risks and costs associated with the process.

Moreover, there is yet to be a deep understanding of the long-term impact that adopting micro frontends can have on the organizational structure of companies. The modularization of interfaces and the autonomy of development teams can profoundly transform how teams organize and collaborate. Studying this dynamic and the possible productivity and organizational efficiency benefits constitutes a significant research gap.

7 FINAL CONSIDERATIONS

The results of this mapping not only enrich the academic body of knowledge but also provide practical guidelines that can be directly applied to real-world software development projects, promoting a more harmonious and effective integration of micro frontends into Software Engineering practices. However, addressing security and management concerns and robust validation methods for successful implementation are crucial. Micro frontends represent a substantial evolution in web development, but their adoption requires a strategic approach for optimal results.

As software architectures evolve, micro frontends are likely to become an essential practice in web application development. The increasing adoption of cloud-based architectures and the integration with microservices amplify the relevance of this approach, enabling greater scalability and modularity. Moreover, the advancement of automation tools and DevOps practices can facilitate the implementation of micro frontends, accelerating the development cycle and ensuring more agile and continuous delivery. The emergence of frameworks and libraries focused on micro frontends also points to the maturation of the technique, making it more accessible and efficient for

Table 7: Implications and Key Findings about Micro Frontends.

Category	**Aspect**	**Description**
Implications for Academia	Emerging Trends	Insights from the analyzed articles reveal emerging trends and current challenges, guiding future research and contributing to the theoretical evolution of micro frontend architecture.
	Benefits and Challenges	Identification of benefits, such as modularity and technological independence, and challenges, such as integration complexity and security, providing a solid foundation for in-depth studies.
Implications for Practitioners	Practical Application	Findings provide practical guidance on application, framework adoption, and best practices for testing and delivering micro frontends.
	Tools and Strategies	Knowledge about management tools' impacts on organizational structure and strategies for overcoming security challenges are precious for effectively adopting this architecture.
Synthesis of Findings	Emerging Trends	Reveal the industry's evolution and adoption of micro frontends.
	Benefits	Include modularity, technological independence, and greater organizational flexibility.
	Challenges	Include integration complexity and security issues that must be addressed.
	Practical Application	Offers guidance on frameworks, testing practices, delivery, and management tools.
	Organizational Impacts	Include changes in organizational structure and team dynamics.

large corporations.

Despite the benefits, the future of micro frontends presents significant challenges that must be addressed. The growing complexity of managing multiple parts of an application can become an obstacle, requiring robust coordination and monitoring practices. Additionally, the lack of standardization between tools and frameworks may hinder interoperability and increase adoption costs. To overcome these challenges and broaden the scope of research, gray literature sources could be included in future investigations, aiming to gather market-driven insights. Although these sources were not used in this initial mapping due to the absence of peer-reviewed rigor, their inclusion could provide practical and valuable insights into adopting micro frontends. Furthermore, as new solutions emerge, such as the use of artificial intelligence to automate the integration of micro frontends, there is great potential for innovations that could transform the development and maintenance of web applications. Therefore, the strategic adoption of this approach presents a promising path for the future.

In addition to practical guidelines and implementation challenges, future research must focus on the quality of micro frontends. The diversity of frameworks and tools can directly impact the maintainability and performance of applications. Studies investigating automated testing practices, such as integration and end-to-end testing, are essential to ensure the robustness of applications composed of micro frontends. Additionally, performance analyses should be conducted to identify bottlenecks and optimize user experience. Code quality, dependency management, and interoperability between different components deserve attention. A greater focus on quality assurance methodologies adapted to this architecture will enable development teams to implement more reliable and scalable solutions, maximizing the benefits of micro frontends in software projects.

REFERENCES

Bian, Y., Ma, D., Zou, Q., and Yue, W. (2022). A multi-way access portal website construction scheme. In *2022 5th International Conference on Artificial Intelligence and Big Data (ICAIBD)*, pages 589–592. IEEE.

Bühler, F., Barzen, J., Harzenetter, L., Leymann, F., and Wundrack, P. (2022). Combining the best of two worlds: Microservices and micro frontends as basis for a new plugin architecture. In *Symposium and Summer School on Service-Oriented Computing*, pages 3–23. Springer.

Capdepon, Q., Hlad, N., Seriai, A.-D., and Derras, M. (2023). Migration process from monolithic to micro frontend architecture in mobile applications. In *IWST*.

de Oliveira, D. S. M., Oliveira, F. C., Pernencar, C. A., de Morais, B. S., Silva, J. W., Costa, A. R., Pereira, J. B., and Saboia, I. F. (2022). Licor: Beyond the design system. a proposal to empower teams to develop software in compliance with the principles of accessibility, usability, and privacy by design in the extreme contexts and challenging domains post-covid-19. In *International Conference on Human-Computer Interaction*, pages 139–147. Springer.

Geers, M. (2020). *Micro frontends in action*. Simon and Schuster.

Keele, S. et al. (2007). Guidelines for performing systematic literature reviews in software engineering. Technical report, Technical report, ver. 2.3 ebse technical report. ebse.

Kitchenham, B. and Brereton, P. (2013). A systematic review of systematic review process research in software engineering. *Information and software technology*, 55(12):2049–2075.

Knecht, C., Schuller, A., and Miclaus, A. (2020). Manageable and scalable manufacturing it through an app based approach. In *Advances in Manufacturing, Production Management and Process Control: Proceedings of the AHFE 2019 International Conference on Human Aspects of Advanced Manufacturing, and the AHFE International Conference on Advanced Production Management and Process Control, July 24-28, 2019, Washington DC, USA 10*, pages 14–26. Springer.

Lorenz, J., Lohse, C., and Urbas, L. (2021). Microfrontends as opportunity for process orchestration layer architecture in modular process plants. In *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 01–04. IEEE.

Männistö, J., Tuovinen, A.-P., and Raatikainen, M. (2023). Experiences on a frameworkless micro-frontend architecture in a small organization. In *2023 IEEE 20th International Conference on Software Architecture Companion (ICSA-C)*, pages 61–67. IEEE.

Mena, M., Corral, A., Iribarne, L., and Criado, J. (2019). A progressive web application based on microservices combining geospatial data and the internet of things. *IEEE access*, 7:104577–104590.

Mohammed, S., Fiaidhi, J., Sawyer, D., and Lamouchie, M. (2022). Developing a graphql soap conversational micro frontends for the problem oriented medical record (ql4pomr). In *Proceedings of the 6th International Conference on Medical and Health Informatics*, pages 52–60.

Nishizu, Y. and Kamina, T. (2022). Implementing micro frontends using signal-based web components. *Journal of Information Processing*, 30:505–512.

Noppadol, N. and Limpiyakorn, Y. (2021). Application of micro-frontends to legal search engine web development. In *IT Convergence and Security: Proceedings of ICITCS 2021*, pages 165–173. Springer.

Pavlenko, A., Askarbekuly, N., Megha, S., and Mazzara, M. (2020). Micro-frontends: application of microservices to web front-ends. *J. Internet Serv. Inf. Secur.*, 10(2):49–66.

Peltonen, S., Mezzalira, L., and Taibi, D. (2021). Motivations, benefits, and issues for adopting micro-frontends: a multivocal literature review. *Information and Software Technology*, 136:106571.

Perlin, R., Ebling, D., Maran, V., Descovi, G., and Machado, A. (2023). An approach to follow microservices principles in frontend. In *2023 IEEE 17th International Conference on Application of Information and Communication Technologies (AICT)*, pages 1–6. IEEE.

Petcu, A., Frunzete, M., and Stoichescu, D. A. (2023). Benefits, challenges, and performance analysis of a scalable web architecture based on micro-frontends. *University Politehnica of Bucharest, Scientific Bulletin., Series C*, 85(3):319–334.

Pölöskei, I. and Bub, U. (2021). Enterprise-level migration to micro frontends in a multi-vendor environment. *Acta Polytechnica Hungarica*, 18(8):7–25.

Schäffer, E., Mayr, A., Fuchs, J., Sjarov, M., Vorndran, J., and Franke, J. (2019). Microservice-based architecture for engineering tools enabling a collaborative multi-user configuration of robot-based automation solutions. *Procedia CIRP*, 86:86–91.

Shakil, M. and Zoitl, A. (2020). Towards a modular architecture for industrial hmis. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 1, pages 1267–1270. IEEE.

Simões, B., Del Puy Carretero, M., Martinez Santiago, J., Muñoz Segovia, S., and Alcain, N. (2023). Twinark: A unified framework for digital twins based on micro-frontends, micro-services, and web 3d. In *Proceedings of the 28th International ACM Conference on 3D Web Technology*, pages 1–10.

Sorgalla, J., Wizenty, P., Rademacher, F., Sachweh, S., and Zündorf, A. (2021). Applying model-driven engineering to stimulate the adoption of devops processes in small and medium-sized development organizations: the case for microservice architecture. *SN Computer Science*, 2(6):459.

Stefanovska, E. and Trajkovik, V. (2022). Evaluating micro frontend approaches for code reusability. In *International Conference on ICT Innovations*, pages 93–106. Springer.

Taibi, D. and Mezzalira, L. (2022). Micro-frontends: Principles, implementations, and pitfalls. *ACM SIGSOFT Software Engineering Notes*, 47(4):25–29.

Tilak, P. Y., Yadav, V., Dharmendra, S. D., and Bolloju, N. (2020). A platform for enhancing application developer productivity using microservices and micro-frontends. In *2020 IEEE-HYDCON*, pages 1–4. IEEE.

Tosic, M., Petrovic, N., and Tosic, O. (2023). Semantic micro-front-end approach to enterprise knowledge graph applications development. In *WEBIST*, pages 488–495.

Velásquez-Durán, A. and Ramírez Montoya, M. S. (2018). Research management systems: Systematic mapping of literature (2007-2017).

Wang, D., Yang, D., Zhou, H., Wang, Y., Hong, D., Dong, Q., and Song, S. (2020). A novel application of educational management information system based on micro frontends. *Procedia Computer Science*, 176:1567–1576.

Wanjala, S. T. (2022). A framework for implementing micro frontend architecture. *International Journal of Web Engineering and Technology*, 17(4):337–352.

Yang, C., Liu, C., and Su, Z. (2019). Research and application of micro frontends. In *IOP conference series: materials science and engineering*, volume 490, page 062082. IOP Publishing.

Zhang, C., Zhang, D., Zhou, H., Wang, X., Lv, L., Zhang, R., Xie, Y., Liu, H., Li, Y., Jia, D., et al. (2023). Application business information interaction bus based on micro frontend framework. In *2023 7th International Symposium on Computer Science and Intelligent Control (ISCSIC)*, pages 306–310. IEEE.

Zhao, L., Kang, Y., and Wang, Y. (2023). Application of micro front-end technology in network public opinion system. In *2023 IEEE 7th Information Technology and Mechatronics Engineering Conference (ITOEC)*, volume 7, pages 227–230. IEEE.