International Research Journal of Advanced Engineering and Science
ISSN (Online): 2455-9024

a Development of Modern Web Application Frontend Structures Using Micro Frontends

I Wayan Krishna Dharma B. P.[1], Dina Anggraini[2]
[1,2]Department of Management Information System, Gunadarma University, Depok, West Java, Indonesia-16424

Abstract— *The development of user needs for a web application that continues to grow more rapidly recently coupled with the digital era and the covid-19 pandemic. Therefore, people are more dependent on internet technology services, which greatly encourages the development and growth of web applications that can be upgraded easier, uncomplicated maintenance, shorter time in repairments, and loosely coupled components. This research's purpose is to produce a new web application with micro frontends structures to enable and simplify future repairs, upgrades, and maintenance through times for the reason that that web application accommodates a large number of users and many types of services menus which will continue to grow and change over time. The research methodology consists of 8 starts from Determining the Research Objects and Subjects, Place and Time of The Research, and Problem Analysis up to Compose Research Reports and Conclusions. The research has produced a web application with micro frontends that the function of the components in micro frontends structure are running well-enough and get pretty good load test results 1.5 seconds in PageSpeed, 1.7s in Page Load Time chrome extension test and Lighthouse test got 82% in performance, 83% in accessibility, best practice got 75% and 70% in SEO.*

Keywords—*Web Application, Microservices Structure, Frontend, Micro Frontends Structure.*

I. INTRODUCTION

Lately, Information and Communication Technologies (ICT) have developed rapidly and become more advanced and complex. This rapid development especially can be seen in the Internet, Cloud Computing, and Web services. This is caused by more people using computers, laptops, smartphones, and the internet for browsing, accessing data, and using applications or even working online through internet networks than before, especially during this Covid-19 pandemic where a lot of work is done online. Therefore, people or users are more dependent on internet technology services. In today's technological developments where the need for data and information is very large and important, even in government agencies, it is very necessary to have a personnel information system to manage or administer the data and information for its personnel, thus causing the need to build a reliable and integrated staffing system for facilitating the huge amount of personnel.

The development of web applications ordinarily still uses a monolithic structure (traditional single-structure) [1]. One of the challenges that arise in monolithic structures is the ability to adapt to changes in system requirements (requirement changes), especially in managing code complexity and maintainability [1]. Applications that are built by using monolithic, all components are written in a single indivisible unit application. This regularly implies that the application has three core parts that trade data with one another: a user interface (UI), a server-side, and a database represented by a single large codebase and has nearly no modularity. Since it just has a solitary codebase, it can turn out to be so huge and subsequently hard to keep up with in future usage [2]. A single small change in the code will make its entire application required to be redeployed. Moreover, this structure is not truly dependable when there are some bugs in any piece of the code that can turn down the entire application and which is not suitable for future improvement and support of its system [2]. If one part of the system is under repair, all services should be suspended or out of service. Suspending all business offerings will make an awful client experience [2]. This is due to the redundant coupling between services in the application.

On the other side, a monolithic structure is a common way for making structural applications. Throughout the long term, utilizing a solitary codebase has made developments simpler and applications size generally little because of their little user population, simple architecture, little size, and low real-time communication requirements. However, the poor development of monolithic structure will slow application development and makes it more though to add fresh developers [3]. Consequently, this fits for small-scale applications.

A greater offer at present is to utilize a microservice to help the ongoing necessities that expand each time and solve existing problems in monolithic [4]. The microservice structures are an unavoidable style of distributed programming application structure regarding flexibility, high extensibility, reliability, high performance, scalability, dynamic scaling in the future, and modularity [4]. Microservices are needed in application development because of microservices goal to perceive a minor service system, every service can be used and updated separately on possibly various platforms and programming languages also runs in its processes and communicates through lightweight procedures like utilizing the REST API [5].

Alongside the ascent of microservice ideas, frontend and backend service detachment empowers system advancement to accomplish decoupling [6]. Based on microservice application innovation, a frontend group regularly develops and carries on a web application that utilizes the REST API to redeem data from backend services [7]. These results in the arising of micro frontend style design in web application development. In a large and complex monolith frontend application, the slightest change occurs in this application, and the entire application must perform a deployment ritual which may have the potential to cause side effects on a live website [7]. The concept of Micro Frontends is named after and inspired from microservices, it's to imagine through a website application frontend as a arrangement of many features which are a small independent part of another big application, or the application consists of many parts and features developed or released by different or separate teams [8]. Each group has particular areas of businesses either assignment it thinks often about and has practical experience in. This group is cross-practical and thrives its frontend attributes end-to-end.

This micro frontend implementation is applied in the web-based application of the government personnel service management information system. This web application is large and complex because it handles a huge amount of data, so this really requires distributed structure with high extensible, low coupling, flexibility, low maintenance, modularity, and dynamic scaling for utilization in the future which will handle a bigger number of service and data keeps up with the growing number of state employees. This web application is developed with ReactJS, Next.JS Framework, React Bootstrap, and ANT Design.

This paper is arranged as follows: Section II reveals state of the art, Section III explicates the research methodology, Section IV conducts results and discussions, and Section V reports conclusions and future work.

II. State of the Art

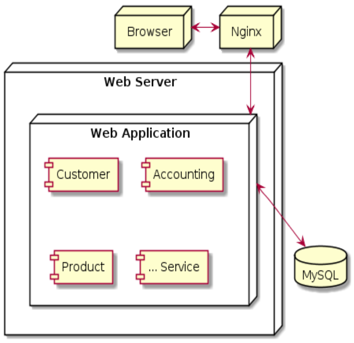


Fig. 1. Example of monolithic structure

In recent years, people are more dependent on applications and internet services. This causes a rise of the different structures of the web application to replace or enhance the traditional unified web structure called monolithic, as shown in figure 1, which is still commonly used and it's not suitable anymore for handling huge amounts of users and services.

Various studies have been conducted to search for better solutions for handling monolithic web structures. Recently, web developers have been discussing microservices and micro frontends as a new approach to replace monolithic in the web structure and increase the ability to upgrade or develop large and complex web applications, so it can catch up with the needs and requirements of users who continue to grow in the future.

Research by Isak Shabani, Endrit Mëziu, Blend Berisha, and Tonit Biba in 2021, the distributed web architecture called microservices is compared to a monolithic structure with the same application. This application with two different structures was tested with Apache JMeter [4]. The test results express that microservice can increase the app's average response time. This is caused by each microservice that works independently needing to exchange and communicate data with one another [9]. Other results present that the error rate in monolithic is better than in microservices. However, the application with microservices structure achieves better in the number of sent kb/sec and receive kb/sec (KiloBytes per second) [9], in the event that, the test scenario consists of post method [9].

One big benefit of microservices is the structure is not fixed with one programming language or one framework. The benefits of microservices can be gained depending on what type of application is and what the developers want to accomplish [9]. The big applications will get advantages from a microservices structure like scalability, modularity, loosely coupled, and can be deployed separately.

In accordance with a research was carried out by Y.R. Prajwal, Jainil Viren Parekh, and Dr. Rajashree Shettar in 2021. The micro frontends' structure concept comes from the extension of microservices used in the backend, as expressed in figure 2. Micro frontends can solve the monolith frontend problem generally used in web applications like restraint on the scalability and modularity of the web application development by multiple teams [10]. This problem can be defeated by decomposing the whole front ends into small front ends by its different functions or features maintained by self-contained teams which every team owns a specific sector of business [10]. Even though micro frontends have impressive benefits as well as significant prospects for frontend development, it has a few drawbacks such as performance latency issues, code redundancy, increased code size, and code divergence [10]. Thereupon micro frontends structure is not fit for web application development with the small number of developers. On the other hand, the micro frontends are good for scalable and medium to large web application development.

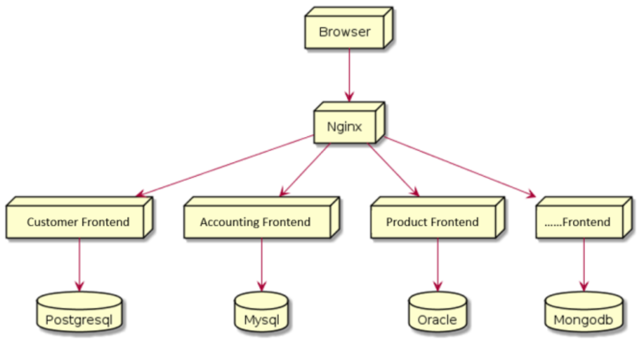


Fig. 2. Example of micro frontends structure

According to the journal by Caifang Yang, Chuanchang Liu1, and Zhiyuan Su in 2018. Micro frontends are great to build a complex large-sized or medium-sized multi-functional frontend web application. The monolith "single" frontend is difficult to maintain and hard to be scaled or deployed as the user traffic grows in the future [11]. The micro frontend structure enables the development teams to develop separately, independently test, and faster deployment, which helps with constant application integration, deployment, and delivery. Although, the micro frontends also bring some limitations; for instance: the micro frontends structure is not proper for the mooa framework system since it cannot perform different technology stacks [11]. Another drawback is the complexity of multiple sub-projects after the integration. This is caused by the increasing dependency redundancy between sub-projects [11].

Based on the research of Ricardo Alexandre Pinto da Silva in 2021. Micro frontends divide the big frontend web application into smaller modules by its functions or attributes, whereas each module is owned by the individual team with end-to-end management [12]. The following are some usefulness of micro frontends for developers: loosely coupled codebase, modular upgrades, overall scalability increment, technology divergence, and fault isolation improvement. Nonetheless, developers have to avoid and be aware of micro frontend structural challenges as it's still a new approach in web development structure [12]. Here are some challenges of micro frontends:

- Overall complexity escalation: This Micro Frontends will be challenging and will introduce the overall complexity of the system because of composing the distributed modules, monitoring all the small pieces, integrating a wide variety of technologies, and organizing different teams with different people [12].
- Scratchy User Experience (UX): The increase of autonomy and isolation, allowing for having a consistent UX and it can become challenging in Micro Frontends since it should not rely on a single shared stylesheet in maintaining the loose coupling [12].
- Environment variety: Each Micro Frontend can be improved, built, evaluated, and distributed independently with the environment. Behavior can be unexpected when shipping to production, and this emphasizes automation and fault isolation to mitigate this issue [12].
- Duplication and shared dependencies elevation: Each frontend is independent, so it needs to be managed carefully, such as code duplication and sharing dependencies [12].
- Enlarge Payload Size: It increases payload sizes and decreases performance as a result of technology variation, which another side can be a benefit in increasing the autonomy of teams [12].

III. RESEARCH METHODOLOGY

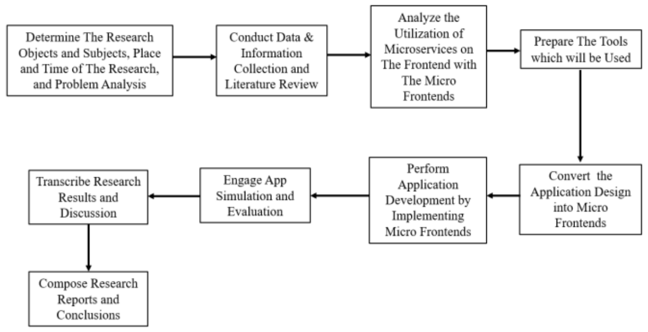

Determine The Research Objects and Subjects, Place and Time of The Research, and Problem Analysis
Conduct Data & Information Collection and Literature Review
Analyze the Utilization of Microservices on The Frontend with The Micro Frontends
Prepare The Tools which will be Used
Convert the Application Design into Micro Frontends
Perform Application Development by Implementing Micro Frontends
Engage App Simulation and Evaluation
Transcribe Research Results and Discussion
Compose Research Reports and Conclusions

Fig. 3. Research methodology stages

In this section will be discussed about the steps and procedures of research activities carried out, from determining the research objects and subjects to composing research reports and conclusions shown on figure 3. This section starts from common research phases then, followed by explanations in each phase below as follows:

1. The first step of this research is determining the research objects and subjects, place and time of the research, and problem analysis.
2. The next step is conducting data and information collection and literature review.
3. Then, analyze the utilization of microservices on the frontend with micro frontends.
4. After that, prepare the tools which will be used.
5. The fifth stage is converting the application with micro frontends.
6. Afterwards, perform application development by implementing micro frontends.
7. The seventh step is engaging app simulation and evaluation with micro frontends.
8. Next stage is transcribing the research results and discussion.
9. The final is composing research reports and conclusions.

A. *Research Objects*

The object of this research is the implementation of micro frontends structure which is relatively new in a web application structure and apply this web structure to the personnel management information system web application.

B. *Research Subjects*

The subjects in this study were ministries and government institutions personnel users of the personnel management information system web application.

C. *Problem Analysis*

There is already a previous personnel management information system web application, in spite of that, it is still using aged technology like flash which has already shut down its operational and support, old web structure which is hard to upgrade, and old frontend UI that's not user friendly anymore. These reasons result that this old web app doesn't qualify the users' needs and requirements currently and even for future use in which everything is digitized. Consequently, this requires developing a new web with a new more modern frontend structure i.e., micro frontends which are discussed in this research.

D. *Data Collection*

Data collection is done by conducting literature reviews to find and study the references for the research carried out in libraries, bookstores, related documents, recordings, and through the internet in the form of writing or pictures. Participant Observation and Content Analysis were carried out to observe the web application system that is running using micro frontends where the researcher participates directly and performs content analysis then receives and records the information about the web application system that suits the needs, supports a good user experience and uses micro frontends. Then note or record the observed results.

E. Prepare the Tools which will be Used

This stage is preparing the tools and installing the frameworks, programming languages and browser extensions that are used to build, in this research the author uses Visual Studio (VS) Code for text editing or writing the program codes, JavaScript, Node.JS, ReactJS, NextJS, Postman, GitHub Desktop, GitLab, Git Bash, Trello, React Developer Tools extension, Redux DevTools extension. In addition, programmers must have an account to use the services of the application, which must use a user account.

F. Convert the Application Design into Micro Frontends

This stage describes the transcribe of the frontend design of the application with micro frontends as seen in figure 4. However, before that, the design process is used to provide an initial description of the appearance of the web application that will be created while converting the design from the UI/UX team into the frontend team, after the design is approved by the clients.

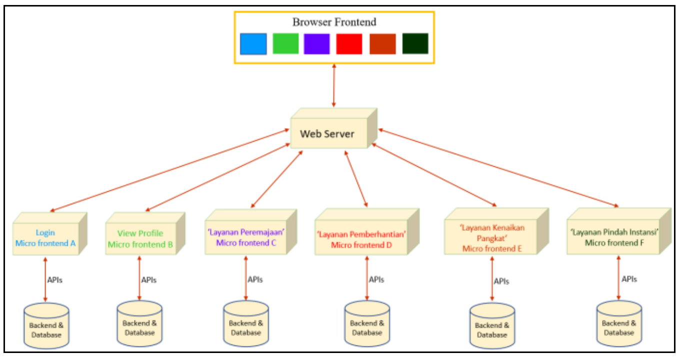

Browser Frontend
Web Server
Login Micro frontend A
View Profile Micro frontend B
'Layanan Peremajaan' Micro frontend C
'Layanan Pemberhantian' Micro frontend D
'Layanan Kenaikan Pangkat' Micro frontend E
'Layanan Pindah Instansi' Micro frontend F
APIs
Backend & Database

Fig. 4. Micro Frontends Design Model

G. Perform Application Development with Micro Frontend

This stage is building the apps by using the tools that have been prepared. The app development stages consist of:

1. Firstly, install those tools, then use VS Code to write the codes and build the directories, folders, structures, etc. and also make the documentation to the codes. As a result, it makes the codes easier to understand by other programmers, since the development involves quite a lot of people.
2. Secondly, use Postman to check the APIs that will be used which have been created by the backend team. If it is working then copy-paste the APIs link that works to the program files that are addressed in VS Code. If it isn't working then contact the backend team to get the APIs fixed.
3. Next step installs the NextJS on the program folder by typing the command 'npm i' on the VS code terminal command prompt, then to run the app use the NextJS command 'npm run dev'.
4. Thereafter the NextJS will run the app program on the URL link: http://localhost:3000. Thereon copy-paste the URL or simply type localhost:3000 on the search bar at the top of the browser like google chrome, Mozilla Firefox, etc. It will run the localhost and direct to the login page.
5. After that the folder program is ready to be used to develop the web app corresponding to the users' requirements. This web application was developed by a team of several people, so the GitLab repository is used to store and unify the source codes of the application that are written by different people, places, and times.
6. The developers while working in the team app development, all members must coordinate very well in the development of web applications to prevent conflicts in the repository and local folders or local files in each developer's device.
7. There are terms Pull, Commit, Push and Merge on the GitLab repository. Pull refers to grabbing any changes from the GitLab repository, then merging them into the user's local repository, Commit means committing the action which records changes in the repository.
8. Push implies pushing, which dispatches the latest commit history from the user's local repository into the GitLab repository. If there's only one working on a repository, pushing is fairly simple, but if there are others accessing the repository or working in a team, developers may need to pull and use Git Branch to create a new branch in the local directory before pushing.
   The GitHub Desktop is used when there is a change to the source code, to check and make sure in detail what the changes are before committing and pushing to the GitLab repository, after pushing the source code to the repository programmers have to merge requests in GitLab to check if there is a conflict in the files that are pushed, while doing these programmers use Git Bash command to be able to interact with GitLab repository, such as if there isn't any conflict in the program files it will continue merging to the GitLab repository, but if there's any conflict to the program files developers will notice that from the git and cancel the merge to avoid that conflict happened to other developers.
   Merge means combining multiple files or objects into one without changing their order. In making a better experience of the repository programmers have to write down the comments that have been changed in the source code in detail. So, other members of the repository can know what changes have occurred on the repository.

H. Engage App Simulation and Evaluation

At this stage, after the design and design stage is done then app simulation and evaluation are carried out for testing, running the simulations, collecting and processing the data, analyzing simulation results whether the coded program runs properly or not before being pushed to the GitLab repository. If it is not appropriate or an error occurs, it must be corrected before being pushed. If it is appropriate or no error occurs, it can be pushed to the repository.

I. Transcribe Research Results and Discussion

This phase contains the results of this research that have been conducted, as well as a discussion of the results of this research. The explanation of this step will be explained in detail in section IV on The Research Results and Discussion.

J. *Compose Research Reports and Conclusions*

This stage carries the results of the research that has been done and the conclusions will be taken as input for the development of the web application. The explanation of this stage will be discussed in detail in section V about Conclusions and Suggestions.

IV. RESULTS AND DISCUSSIONS

This section explains the results and discussions about the research that has been conducted in a personnel management information system web application using micro frontends structure.

A. *App Development and Server Implementation*

This stage is the implementation of the analysis to find out the stages used during the application development process by using the GitLab repository as a platform developer to implement the application to the server when the application program has been repaired or added the components so that the client can see the results of the development of the application, where the client sees the final result of the application on the server. Figure 5 is a diagram of the stages during the development process to the implementation of the application to the server.

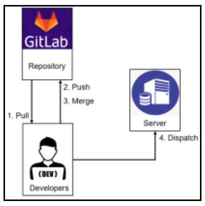
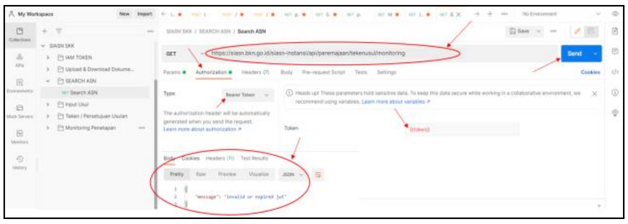


Fig. 5. Process Diagram of Development with the Repository and Server

B. *APIs Checking*

Frontend developers can check or test the APIs collection that has been given by the backend developers if the APIs are working properly as should be. This action can be taken by using the Postman app and it's free to use as seen in figure 6.



GET https://siasn.bkn.go.id/siasn-instansi/api/peremajaan/tekenusul/monitoring

The authorization header will be automatically generated when you send the request. Learn more about authorization ↗

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about variables ↗

{{token}}

```
{
    "message": "invalid or expired jwt"
}
```

Fig. 6. The API URL and Token

C. *The Web Micro Frontends Design*

The application of micro frontends in web applications in this study can be seen in figure 7 which shows the design diagram of the structure of micro frontends. The container frontend is a display that is visible in the user's browser, which is all of the micro frontends that are combined into a single web application frontend. This solution makes it easier for developers to build, improve, change independently and focus without disturbing other parts or large applications.

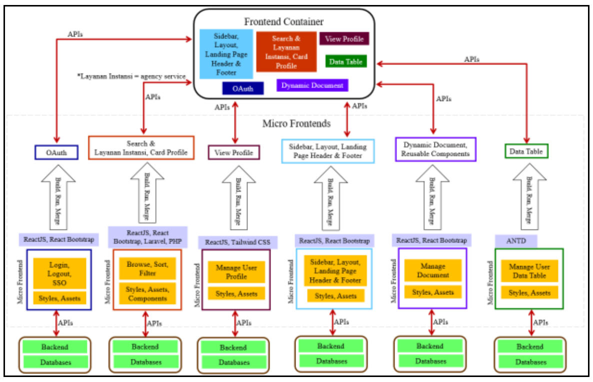

Frontend Container
Sidebar, Layout, Landing Page Header & Footer
Search & Layanan Instansi, Card Profile
View Profile
Data Table
OAuth
Dynamic Document
*Layanan Instansi = agency service
APIs
Micro Frontends
OAuth
Search & Layanan Instansi, Card Profile
View Profile
Sidebar, Layout, Landing Page Header & Footer
Dynamic Document, Reusable Components
Data Table
Build, Run, Merge
ReactJS, React Bootstrap
ReactJS, React Bootstrap, Laravel, PHP
ReactJS, Tailwind CSS
ANTD
Micro Frontend
Login, Logout, SSO
Browse, Sort, Filter
Manage User Profile
Sidebar, Layout, Landing Page Header & Footer
Manage Document
Manage User Data Table
Styles, Assets
Styles, Assets, Components
Backend
Databases

Fig. 7. The Micro Frontends Diagram

The micro frontend in this application communicates for data exchange with the backend and database using separate APIs, where the APIs used have been created by the backend team. The separation of frontend, backend, and database is very helpful in the application development process, each part can be done independently without disturbing the other parts such as when there is damage or system repair in one part of the database, this does not make the other part of the frontend or backend become disturbed or down.

D. *The App Running Flowchart*

A flowchart is a diagram that shows the steps and gives an idea of how a program goes from one process to another of a program. Thus, the flow of a program will be clearer, easier to understand, concise, and reduce the possibility of misinterpretation. Figure 8 shows the flowchart to run the web app in development mode from starting the localhost:3000 server using 'npm run dev' to perform all services provided.

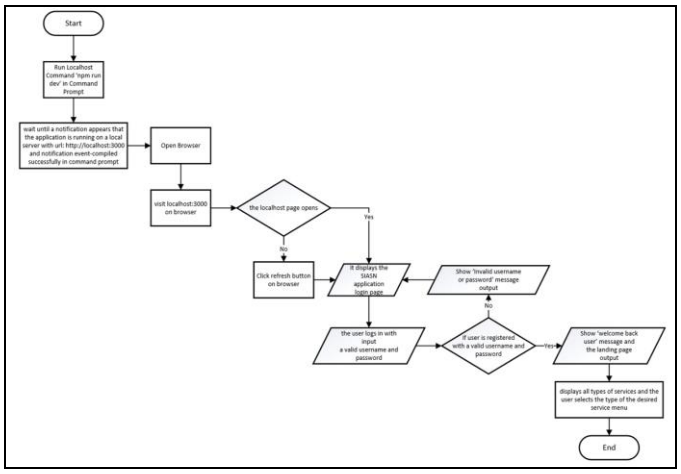

Start
Run Localhost Command 'npm run dev' in Command Prompt
wait until a notification appears that the application is running on a local server with url: http://localhost:3000 and notification event-compiled successfully in command prompt
Open Browser
visit localhost:3000 on browser
the localhost page opens
Yes
No
Click refresh button on browser
it displays the SIASN application login page
Show 'invalid username or password' message output
the user logs in with input a valid username and password
If user is registered with a valid username and password
Show 'welcome back user' message and the landing page output
displays all types of services and the user selects the type of the desired service menu
End

Fig. 8. The App Running Flowchart

E. *The Web Frontends*

The appearance of the application can be seen in the figure 9 below, which can be seen the breakdown of application looks using the micro frontend. The view at the top of the diagram that is seen in the browser is a combination of all the views that are broken down using the micro frontend which is on the its down side of the diagram in figure 9.

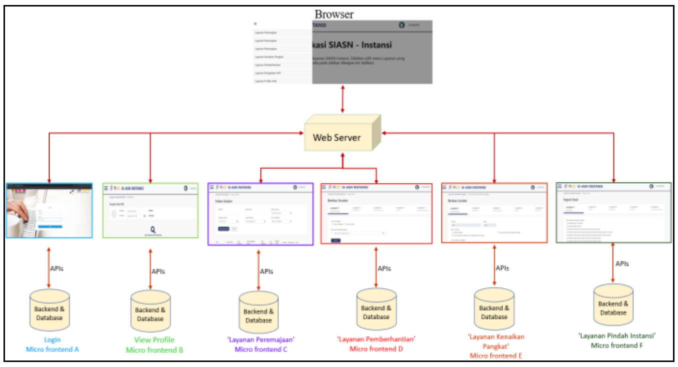


Fig. 9. The View Diagram

F. *The Utilization APIs in Frontend*

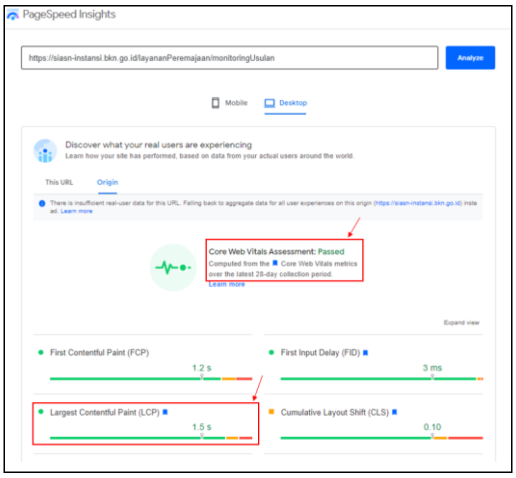
The use of APIs on the micro frontend is very important, which is used as a medium for exchanging data with the backend and database, without APIs the components on the micro frontend cannot exchange data, both sending and receiving data. This is because the micro frontend is a separate part from the backend and the database. The use of APIs in a web app by creating a special file for a separate set of APIs, to simplify naming can be given the name api.ts in the folder named configs (refers to configurations).

G. *Page Loading Time Check*

Website Page Load Time is the time needed for the browser to open or display a web page on a browser as a whole, so the visitor is able to see the overall appearance of the site. A website that has a very fast loading time, provides convenience to its visitors. A survey conducted by Akamai in conjunction with Gomes.com about website load times revealed that almost half of the respondents wanted the website to appear fully within 2 seconds. If the load time reaches more than 3 seconds, users will choose to visit another site that is faster. A DoubleClick study from Google found that as many as 53% of websites will be abandoned by visitors if the website takes more than 3 seconds to load. In the PageSpeed test, the web application got 1.5 seconds for the load test as shown in figure 10. Figure 11 shows the Page Load Time chrome extension test result which is got 1.7 seconds.

H. *Testing by Lighthouse Chrome Browser*

Other tests were carried out using Lighthouse by the chrome browser, the use of Lighthouse was done by inspecting elements of the web app on chrome browser being tested by right-clicking on the mouse or laptop touchpad ⇒ click 'inspect' ⇒ click the 'Lighthouse' tab button ⇒ click 'Generate report' button and wait until the Lighthouse process finishes doing the test. Figure 12 reveals the Lighthouse test result in chrome inspect element which is got 82% in performance, 83% in accessibility, best practice got 75%, and 70% in SEO.


PageSpeed Insights

https://siasn-instansi.bkn.go.id/layananPeremajaan/monitoringUsulan — Analyze

Mobile | Desktop

Discover what your real users are experiencing
Learn how your site has performed, based on data from your actual users around the world.

This URL | Origin

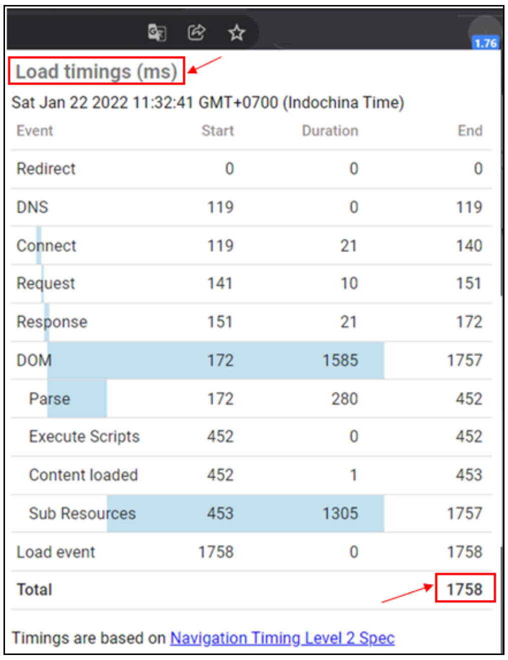
There is insufficient real-user data for this URL. Falling back to aggregate data for all user experiences on this origin (https://siasn-instansi.bkn.go.id) instead. Learn more

Core Web Vitals Assessment: Passed
Computed from the Core Web Vitals metrics over the latest 28-day collection period.
Learn more

Expand view

First Contentful Paint (FCP) 1.2 s
First Input Delay (FID) 3 ms
Largest Contentful Paint (LCP) 1.5 s
Cumulative Layout Shift (CLS) 0.10

Fig. 10. The PageSpeed by Google Test Result


1.76

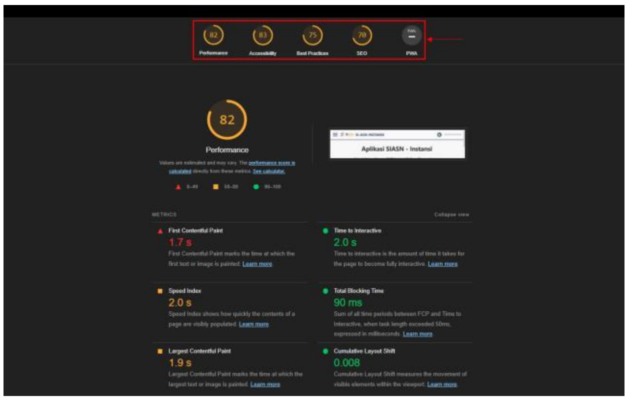
Load timings (ms)

Sat Jan 22 2022 11:32:41 GMT+0700 (Indochina Time)

Event	Start	Duration	End
Redirect	0	0	0
DNS	119	0	119
Connect	119	21	140
Request	141	10	151
Response	151	21	172
DOM	172	1585	1757
Parse	172	280	452
Execute Scripts	452	0	452
Content loaded	452	1	453
Sub Resources	453	1305	1757
Load event	1758	0	1758
Total			1758

Timings are based on Navigation Timing Level 2 Spec

Fig. 11. The Page Load Time Google Chrome Web Extension Test Result


82 Performance | 83 Accessibility | 75 Best Practices | 70 SEO | PWA

82
Performance
Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.
0–49 | 50–89 | 90–100

Aplikasi SIASN - Instansi

METRICS — Collapse view

First Contentful Paint
1.7 s
First Contentful Paint marks the time at which the first text or image is painted. Learn more

Time to Interactive
2.0 s
Time to interactive is the amount of time it takes for the page to become fully interactive. Learn more

Speed Index
2.0 s
Speed Index shows how quickly the contents of a page are visibly populated. Learn more

Total Blocking Time
90 ms
Sum of all time periods between FCP and Time to Interactive, when task length exceeded 50ms, expressed in milliseconds. Learn more

Largest Contentful Paint
1.9 s
Largest Contentful Paint marks the time at which the largest text or image is painted. Learn more

Cumulative Layout Shift
0.008
Cumulative Layout Shift measures the movement of visible elements within the viewport. Learn more

Fig.12. The Result Review of Lighthouse Testing

V. Conclusions

In accordance with the outcomes of the research that have been accomplished, it can be inferred that the process of developing a website-based SIASN application can be done by utilizing the application of micro frontends and the function of the components in micro frontends structure are running well-enough and get pretty good load test results: 1.5 seconds in PageSpeed, 1.7s in Page Load Time chrome extension test and Lighthouse test got 82% in performance, 83% in accessibility, best practice got 75% and 70% in SEO. Due to the fact that, this web application in this study is still in the development stage, the app still needs improvements in terms of performance, accessibility, best practices and SEO, so that it can produce web applications that satisfy its users. There are some benefits of micro frontends experienced by the author for instance: team organization autonomy, modular upgrades and independent app development. This also comes together with some drawbacks including scratchy user experience (UX), enlarged payload size, overall complexity escalation, and environment variety.

VI. Future Work

The future work for the next development on this research is to develop the government personnel management information system website application for much better such as optimizing its performance, codebase as well fixing the bugs, for the reason that this web application is still not optimal and is still in the development stage. On the other hand, over time, the use and application need to increase, especially for large application projects consisting of many teams. So, the application of the micro frontend in the development of this research in the future will add new features such as digital signature, users' performance graphs, education and training level charts, and so on. Besides that, it will also explore more in the use of other programming languages and frameworks including Ant Design, TailwindCSS, Python, Vue.js, Laravel, and all the rest that are tailored to the needs and developments of technology in programming.

In addition, further research could be conducted by studying more about other software quality attributes such as web app security, SEO, and JMeter testing which are very important in a web application. So that, in the end, it can deliver micro frontends solutions to increase productivity in app development, improve data security, create a better loosely coupled codebase and escalate team collaboration.

References

[1] F. Auer, V. Lenarduzzi, M. Felderer, and D. Taibi, "From monolithic systems to Microservices: An assessment framework," *Inf. Softw. Technol.*, vol. 137, p. 106600, 2021.

[2] D. Wang *et al.*, "A Novel Application of Educational Management Information System based on Micro Frontends," *Procedia Computer Science*, vol. 176, pp. 1567–1576, 2020.

[3] M. Gribaudo, M. Iacono, and D. Manini, "Performance Evaluation of Replication Policies in Microservice Based Architectures," *Electron. Notes Theor. Comput. Sci.*, vol. 337, pp. 45–65, 2018.

[4] Suthendra, J. A. and Pakereng, M. A. I. "Implementation of Microservices Architecture on E-Commerce Web Service," *ComTech: Computer, Mathematics and Engineering Applications*, vol. 11 issue 2, pp. 89–95, 2020.

[5] Irudayaraj, J. P. "Adoption Advantages Of Micro-Service Architecture In Software Industries", 2019.

[6] Swathi *and* Rashmi R. "Microservice Architectural Style", *International Research Journal of Engineering and Technology (IRJET)*, vol. 7 issue 8, pp. 1109– 1112, 2020.

[7] I. Manuel Kroiß and M. Kroiß Schahram Dustdar, "From Backend to Frontend - Case study on adopting Micro Frontends from a Single Page ERP Application monolith," 2021.

[8] S. Peltonen, L. Mezzalira, and D. Taibi, "Motivations, benefits, and issues for adopting Micro-Frontends: A Multivocal Literature Review," *Inf. Softw. Technol.*, vol. 136, 2021.

[9] Isak Shabani, Endrit Mëziu, Blend Berisha, Tonit Biba "Design of Modern Distributed Systems based on Microservices Architecture," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 12, no. 2, 2021.

[10] Y.R. Prajwal, Jainil Viren Parekh and Dr. Rajashree Shettar. "A Brief Review of Micro-frontends," *United International Journal for Research & Technology (UIJRT)*, vol. 02, issue 08, 2021.

[11] C. Yang, C. Liu, and Z. Su, "*Research and Application of Micro Frontends*," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 490, no. 6, 2019.

[12] Ricardo A. Pinto da Silva, "A Micro Frontends Solution Analyzing quality attributes," M.S. thesis, Department of Informatics Engineering, Instituto Superior de Engenharia do Porto, Porto, Brazil, 2021.