RES
IT

Research Report

RESERVATION SYSTEM FOR DIM
GROUP 2 S6-RB03

Preface

This document contains all findings regarding our research questions. The goal was to find out whether or not a no-code application, specifically Appsemble, could be used in combination with an enterprise architecture in a restaurant system. It should also satisfy the following requirements: security, performance, easy to use and in compliance with GDPR.

We were given the assignment to build open-source restaurant software. The added value is to make restaurants more crisis resilient. At the moment expensive vendors like Thuisbezorgd take a large commission and systems like Resengo do not allow for automatic exportation of data. Making software open-source and widely available helps restaurants get through crises more easily by saving on money.

A requirement from the client is to use Appsemble, a free and open-source low-code software platform. The problem however is that we are still uncertain whether or not Appsemble can fit within an enterprise architecture. This is what we are trying to figure out with the research questions.

Table of contents

Preface 2

Results 4

1. Can low-code applications be sufficiently performant? 4

2. Can low code applications, specifically Appsemble, be connected to messaging systems like a service bus? 5

3. How can messaging within an enterprise system be done reliably and securely? 5

4. Are there any other better technologies that can be used instead of Appsemble within the context? 7

Conclusion 9

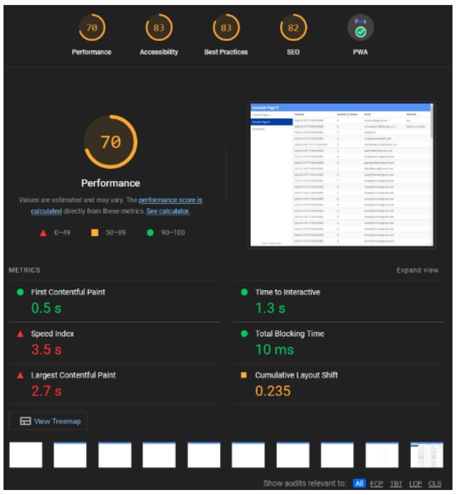
References 10

Results

1. Can low-code applications be sufficiently performant?

During our exploration of Appsemble as a low-code platform a few concerns arose when it comes to how it will be able to perform. The website on first load, takes over 5 seconds to load the page, because of a large JavaScript file being loaded. This happens until the file is cached, when the cache is removed, either by reloading via Alt + F5 or closing the browser down, the same happens when the page is opened.

Additionally, we ran a few Google Lighthouse tests to determine the performance of the website compared to others that have a similar page. We compared a page where data about reservation is being displayed on an Appsemble app versus a website where information about TV shows is displayed in a similar manner. Both pages contain a list of one hundred (100) entries to make the comparison as fair as possible. Comparing the results of both in the Performance category the TV shows website performs much better acquiring a score of 93, while the Appsemble app only has a score of 70. However, it is a PWA (Progressive Web App), which makes it better for mobile use. Furthermore, its accessibility, best practices and SEO are doing fairly well score wise, thought they could use an improvement to around 90 so it is in the good percentile, according to Lighthouse. However, as we were testing for performance, the biggest worry when it comes to Appsemble is the 3.5 seconds, in the speed index of Lighthouse, which is in the red – meaning it's performing poorly.



70 Performance · 83 Accessibility · 83 Best Practices · 82 SEO · PWA

70 Performance

Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

▲ 0–49 ■ 50–89 ● 90–100

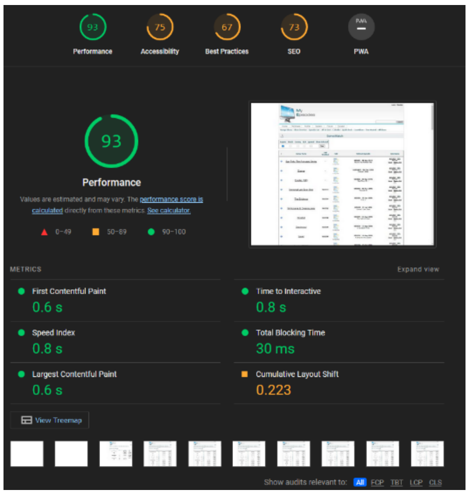
METRICS Expand view

● First Contentful Paint 0.5 s	● Time to Interactive 1.3 s
▲ Speed Index 3.5 s	● Total Blocking Time 10 ms
▲ Largest Contentful Paint 2.7 s	■ Cumulative Layout Shift 0.235

View Treemap

Show audits relevant to: All FCP TBT LCP CLS

Figure 1. An Appsemble page's, with registration data, results according to Google Lighthouse



93 Performance · 75 Accessibility · 67 Best Practices · 73 SEO · PWA

93 Performance

Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

▲ 0–49 ■ 50–89 ● 90–100

METRICS Expand view

● First Contentful Paint 0.6 s	● Time to Interactive 0.8 s
● Speed Index 0.8 s	● Total Blocking Time 30 ms
● Largest Contentful Paint 0.6 s	■ Cumulative Layout Shift 0.223

View Treemap

Show audits relevant to: All FCP TBT LCP CLS

Figure 2. MyEpisodes.com page's, displaying TV series and some details, results according to Google Lighthouse

Looking at the results it is fair to say that Appsemble struggles with some performance issues, sometimes it takes long to load, especially if it's for the first time, in other cases it doesn't perform to the high quality possible according to top-tier testing software. Additionally, there are some issues with Appsemble's API – like not being able to store a DateTime format properly, which hinders it as a viable full web app. However, if used purely for the front-end of the application and a lot of the logic is handled in the back-end, then it can perform good enough to support an application that matches the needs of our system.

2. Can low code applications, specifically Appsemble, be connected to messaging systems like a service bus?

Having carefully read through the documentation and scoured through the source code we found that as of right now it is not possible to connect the backend to a service bus. There are however several options to make this possible within the Appsemble environment.

1. Extend the source code of Appsemble
2. Connect a custom backend to the Appsemble app (as defined here)
3. Using a polling service

Extending the source code would mean creating a block where developers define which data should be published or subscribed to. The backend would need to be extended in a way that it can dynamically create consumers or producers. Taking on this challenge would be going outside our predefined scope and would deviate from our eventual goal. There is also the issue of having to make a pull request and it needing to get accepted. On top of this we need to get familiar with code that already has a large complexity to it.

Connecting to a custom backend would be a lot more convenient considering the time and scope. However, this does make it even less convenient to use Appsemble. Appsemble would only be used as the front-end. We would also be forced to follow the standard of Appsemble. The reasoning for using Appsemble was transferability, ease of use and being able easily deploy applications. These would be negated when opting for a custom backend.

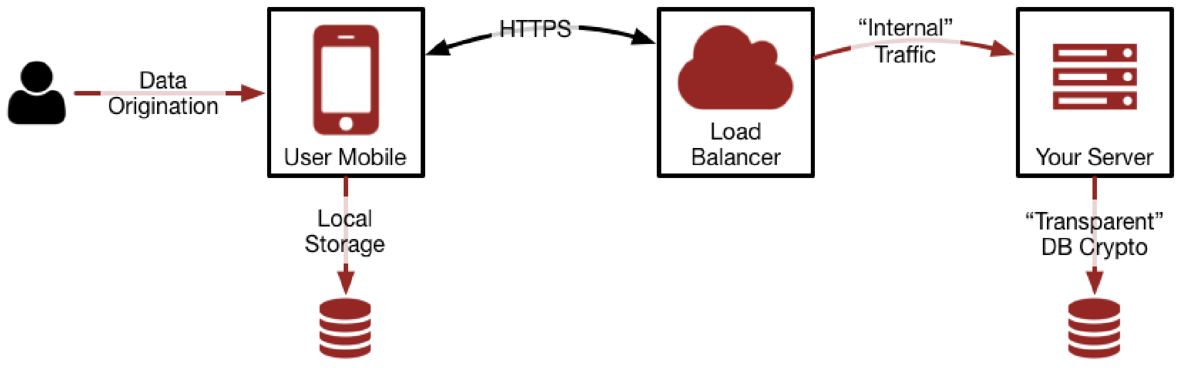
Using a polling service is technically possible but not a viable option for many reasons, it doesn’t scale well. It unnecessarily uses computing power and there is no place for it in an enterprise software architecture.

None of the options above are ideal, but the second one would definitely be preferable. The question that remains is whether or not Appsemble is the right tool for the job.

3. How can messaging within an enterprise system be done reliably and securely?

In an enterprise system we value the reliability and the security of the system. In order to improve the system on this area there are multiple things you should look into.

First of all, security processes should be established to protect and encrypt data at all levels of your enterprise. All communications and data transmission should be protected with end-to-end encryption to prevent data theft. As you can see in the image below, HTTPS alone is not enough. Black is HTTPS encrypted, however red is still plaintext. Implementing end-to-end encryption makes sure the data gets encrypted at the start and only gets decrypted at the point of use. This way you are not vulnerable at multiple points throughout your system anymore since your data will be encrypted throughout the system.



Besides encrypting data, it is also smart to encrypt secrets that the microservices use for communication like API keys, client secrets, or credentials for basic authentication.
If you make use of any third-party app, your encryption should extend to that app as well. Also, you should be aware of the vulnerabilities the libraries you are using are bringing with them, so frequently scan the source code, new code contributions, and deployment pipeline for vulnerable dependencies.

To further prevent other people from retrieving our data without permission it is important that authorization/authentication is included. There are a lot of different possible aspects to a microservices architecture, in order to deliver secure and effective authorization/authentication across the microservices it is important to put the right tools and protocols in place.

Besides thinking about all these ways to protect our system, it is also important to make sure your system works as intended. By including unit tests like SAST and DAST into your CI/CD pipeline you relieve the developers of manual security checks.

Attackers can always try to get authorized by brute forcing your authentication service, although this often takes some time you can slow this down even more. By implementing a rate limit, you limit them to only a few attempts per second. This way they might lose interest in this type of attack.

Also, it is important to give each role as limited amount permission as possible. This way you minimize the risk of unauthorized people gaining access to parts of the system you do not want them to.

4. Are there any other better technologies that can be used instead of Appsemble within the context?

Appsemble is not the only no-code/low-code framework there is, there are a lot of others as well. For this project the main purpose is that all the code must be open source. Not all the no-code/low-code frameworks are open source.

The context of this project is focused on open-source projects. The project needs to be transferred to other groups and they should be able to continue with the project. So, it needs to be easy to understand but also easy to maintain, people without coding experience should be able to understand what is going on. The project should be able to communicate with several applications and must be easy to extend, almost it must be plug & play. The communication part must work with a service bus like RabbitMQ or Kafka.

When looking at the top low-code/no-code frameworks there are some competitors for Appsemble that also fall in the scope, starting with ABP framework.

ABP is a framework that is written in ASP.NET Core. With this framework you can create web applications fast and easy. It is a low code framework, so it is not drag and drop and it has a small learning curve. One of the big advantages of ABP is that it is microservices compatible, the framework has the microservices architecture in mind. Also, the possibility to connect it to an event bus like Kafka of RabbitMQ is there which is a great plus point. It has a lot of NuGet packages and pre-built modules that you can use. Authentication and authorization are built in with the ASP.NET Core Identity & IdentityServer4. ABP can be self-hosted which can result in lower hosting costs.

Budibase is another low-code framework, but it is written in JavaScript and TypeScript. This platform comes with its own builder, in this builder you can use a simple drag and drop to build up the application. Budibase is also an open-source application so it can be self-hosted which can lower hosting costs. Budibase can also be deployed with Kubernetes, but it has no implantation with service bus applications like Kafka or RabbitMQ. The authentication and authorization are handled by Auth0.

Structr is a low-code framework based on graph technology. It is specially used for web-based enterprise applications. It can use REST and web sockets and it can be connected to a service bus like Kafka or RabbitMQ. It uses drag and drop technology to easily create a project. This results in a lower learning curve. For creating REST API, it uses a graph like technology to create an API. Structr is open source so it can be self-hosted which can result in lower hosting costs.

Appsemble uses a low code editor to create an application, you can build an application by creating a scheme. Appsemble has the option to host it yourself but also to host it by Appsemble. Appsemble is optimized to be used for PWA, React/Node.js, Docker and Kubernetes. The authentication and authorization part are done by JWT or with OAuth2. Appsemble uses building blocks that you can implement within your project. These building blocks can be created by yourself, or you can use building blocks from others. Communication with a service bus like Kafka or RabbitMQ is not supported. Because Appsemble is open source the code can be altered and the integration with a service bus can be made.

With this information a comparison table can be created to see the similarities and the differences between the low-code platforms.

	Appsemble	ABP	Budibase	Structr
Easy to learn	X	~	X	X
Works with service bus	-	X	-	X
Open source	X	X	X	X
Good support and documentation	X	X	X	X
Self-hosting capability	X	X	X	X
Good authentication and authorization	X	X	X	X

Looking at the comparison table the results are close. ABP is the only one that is more difficult to learn than the others because it uses coding instead of a drag and drop technology. Appsemble and Budibase cannot be connected to a service bus by default, the source code can be changed that it will work with a service bus, but this costs more time and effort. All the platforms are open source and have a good support and documentation which is good for the transferability to other groups. Because all the platforms are open source, they can be self-hosted and keep costs down. All platforms have good authentication like OAuth or other SSO services which is good.

In conclusion Structr has ticked all the boxes and comes out as the best. It is a simple low-code platform that works with drag and drop or with a graph like technology. It is included with a good documentation. The biggest plus point is that it can connect to a service bus, something Appsemble cannot do. There is also the possibility that Appsemble will be changed in the source code that it can use a service bus than is Appsemble the better choice.

Conclusion

References

L., & Broek, K. V. D. (2019, July 19). *Low-code app building platform*. App emblem.

Retrieved 17 March 2022, from https://appsemble.app/en/docs

***ABP Framework - Open Source Web Application Framework*. (2017, 4) May). ABP framework.**

Retrieved 17 March 2022, from https://abp.io/

Budibase *Open-source low-code platform*. (2019, February 26). Budibase.

Retrieved 17 March 2022, from https://budibase.com/

***Semantic Low-Code Platform based on Graph Technology | Structr*. (2010, May 15). Structr.**

Retrieved 1717 March 2022, from https://structr.com/

Musa, H. (2022, January 6). *20 Open-source Low-code platforms for 2022*. MEDevel.Com.

Retrieved 17 March 2022, from https://medevel.com/19-open-source-low-code/

Potoczny-Jones, I. (2022, Feb. 4). End-to-end encryption - Why HTTPS is not enough. Tozny.

Retrieved 17 March 2022, from https://tozny.com/blog/end-to-end-encryption-vs-https/

Untethered Labs, Inc. (2022, February 28). Principles to ensure a good enterprise system security architecture. GateKeeper Proximity passwordless 2FA.

Retrieved 17 March 2022, from https://gkaccess.com/principles-to-ensure-good-enterprise-system-security-architecture/

8 Ways to | Your Microservices Architecture Okta. (2020, May 22). Okta, Inc.

Retrieved 17 March 2022, from https://www.okta.com/resources/whitepaper/8-ways-to-secure-your-microservices-architecture/