A Collaborative Visual Database

by

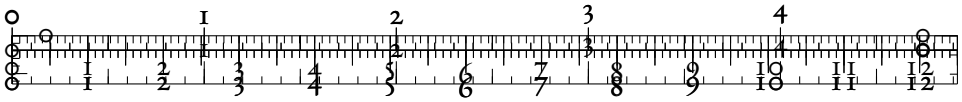
Imed Adel

A document submitted in partial fulfillment of the requirements for the degree of

Technical Report

at

Miskatonic University



I Presentation

1.1 Introduction

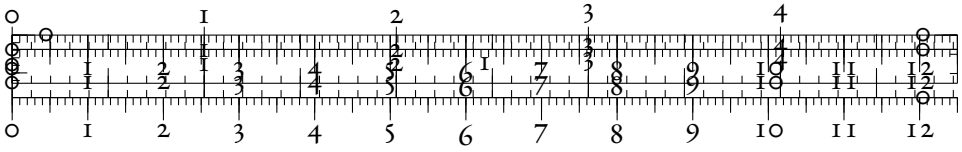
The aim of this chapter is to contextualize our work. We will start by introducing the hosting institution for the graduation project. Then, we will present the project, its motivations, and its objectives. Finally, we will discuss the development process used throughout the making of this project.

1.2 Host

This project is done as part of the final graduation project with the goal of obtaining the License Degree in Computer Science within the Higher School of Sciences and Technology of Hammam Sousse.

1.3 Project presentation

Our project's main idea and design choices stem from the problems we faced while trying to accomplish certain tasks using other tools.



1.3.1 Problematics

The continuous shift to Software as a Service (SaaS), coupled with the rise of remote work, uncovered a gap in the field of data and content management software. The gap is further exacerbated due to the accelerating adoption of web applications, which are mostly client-side applications without any server requirements. Nowadays, businesses are looking for easy and collaborative ways to allow stakeholders to manage data and content, and to connect the data to their different applications. The solution must respond to the needs of businesses from different backgrounds, with varying budgets, and minimal technical knowledge. The solution must also be easily integrable with other tools that these businesses might rely on. Furthermore, it must support recent technological advancements on the web, such as real-time collaboration and real-time queries. To ensure these requirements, we need to answer the following questions:

- How to support real-time collaboration?
- What level of collaboration is required for optimal productivity?
- How should we organize and share data between multiple users?
- What interface structure ensures the most accessible software?
- What data types should we support?
- How important is speed?
- How can we ensure a fast user experience?
- What are the bottlenecks of the existing solutions, and how can we solve them?
- How can we ensure a fast and easy on-boarding?

1.4 Preliminary study

Before starting the development process of our projects, it is of utmost importance to research the existing solutions in the field of data and content management that our potential users are currently relying on. It is necessary to understand what problems users are facing while using these solutions and what kind of tricks and shortcuts do they have to depend on to achieve their desired outcome. We should also focus on the points that users admire about their current choices, as these are the features keeping them from looking for another solution in the meantime.

With this goal in mind, we went on to research multiple applications and software systems with varying degrees of features and requirements. While some might require deep technical knowledge of databases, servers, and programming, others are more straightforward and require little to no technical knowledge. However, while some applications may require no programming skills, they still require some time for on-boarding and getting familiar with the software. This can be a significant roadblock for many enterprises that are already stuck with some other software system.

From this wide pool of data and content management software, we selected the most used and loved ones and put them to comparison. In particular, we chose to focus on Notion, Airtable, Contentful, Sanity, Webflow CMS, and Firebase.

1.4.1 Existing solutions

We will start by presenting the selected solutions.

Firebase

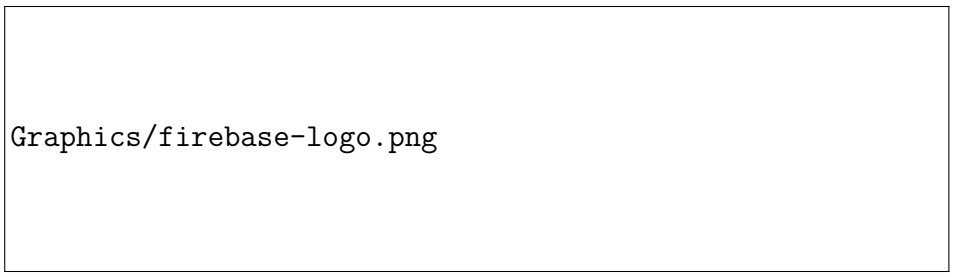


Figure 1.1: Firebase logo

Firebase is a platform developed by Google for creating mobile and web applications. It was initially released in 2012. It offers, among its products, a real-time database, in which, data is stored in JSON format and synced between all the connected clients. The database was not developed with non-technical users in mind, however, its real-time capabilities offer an example of what's desired in real-time database software. Firebase Realtime Database has been successfully used to develop highly demanding mobile applications.

Airtable

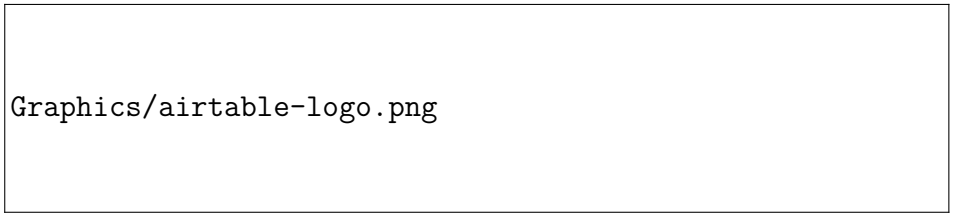


Figure 1.2: Airtable logo

Airtable is a visual database app inspired by the ease of spreadsheets and the wide adoption of software like Microsoft Excel. The company behind the app was founded in 2012.

Airtable comes with team collaboration out of the box. It also automatically generates a REST API from each database.

Pricing is done per team member. There are several limits to the size of storage and uploads.

Contentful

Graphics/contentful-logo.png

Figure 1.3: Contentful logo

Contentful is a headless [1] Content Management Software (CMS). It offers a flexible CMS editor and a configurable API. It also comes with multiple SDKs in multiple programming languages to make its integration easier.

Pricing is offered per package, with the lowest premium package starting at US$489 per month.

Sanity

Graphics/sanity-logo.png

Figure 1.4: Sanity logo

[1] Content is decoupled from the main application. It's made accessible through a set of APIs.

Sanity is another headless CMS. It competes directly with Contentful, offers an even more configurable editor, and its pricing starts at US$199 per month. It comes with real-time collaboration, a feature that Contentful lacks.

Webflow CMS

Graphics/webflow-logo.png

Figure 1.5: Webflow logo

Webflow is a website builder. It bundles a CMS and an e-commerce management system along with its visual website builder. The CMS is not usable outside of Webflow websites, however, it comes with an intuitive user interface.

Notion

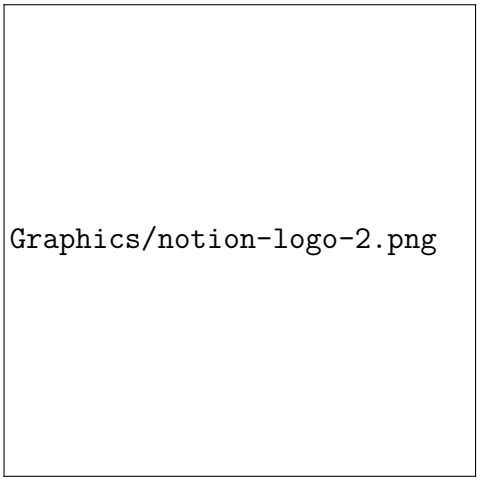


Figure 1.6: Notion logo

Notion is a new contender in the space of content management. It presents itself as a collaborative workspace for teams. Its use cases vary from product management and team documentation to note-taking and personal organization. The initial version of Notion was released in 2016. The second version, which received a lot of praise and media coverage, was released two years later in 2018. However, the largest surge in sign-ups happened during the pandemic, with 40% of sign-ups occurring from December 2020 to January.

Notion is built on the concept of blocks: A block is any single piece of content you add to your page, like a to-do item, an image, a code block, an embedded file, etc. This makes it easy to build complex pages and move content around.

Notion is also built as a collaborative web app—eliminating the need for saving and figuring out how to share one's documents as is the case in other apps.

Pricing is done per workspace member with unlimited storage starting from the free plan.

1.4.2 Comparison of the existing solutions

In order to have a better understanding of the different offerings of the selected solutions, and their features and shortcomings, we have to compare them side by side.

Methodology

For a fair and objective comparison of the different solutions, we will rely on Web.Dev and Google Chrome Lighthouse for measuring performance, accessibility, speed and security, and Capterra for aggregating user reviews and forming a consensus about the main strain points in the existing tools.

Web.Dev Web.Dev is a web application developed by Google that uses the Lighthouse tool to measure different websites and web applications metrics. It can only audit public web pages, however, it offers metrics totally unbiased by our browser setup.

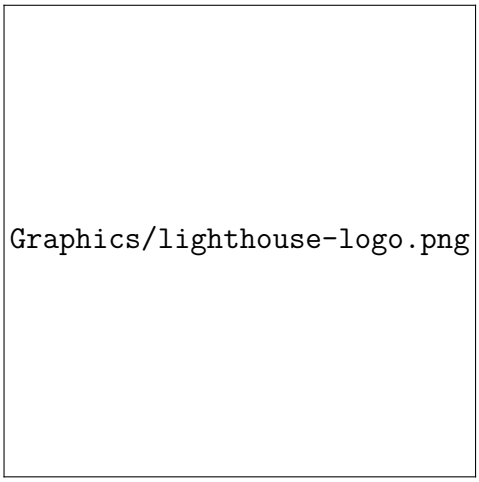


Figure 1.7: Lighthouse logo

Google Chrome Lighthouse Google Chrome Lighthouse is an extension available by default in Google Chrome browsers. It can measure different website and web application metrics. It can audit both public and private web pages. However, the results can be affected by any installed browser extensions. Which is why we run this tool in an isolated browser installed for this particular use case.

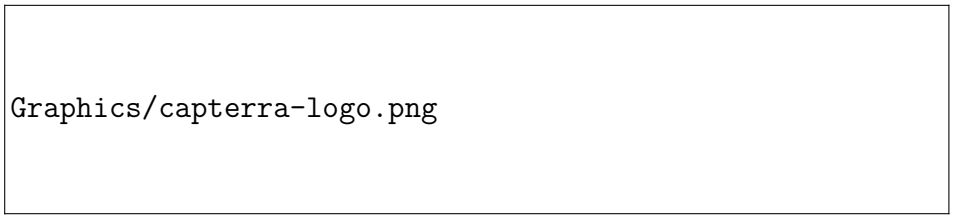
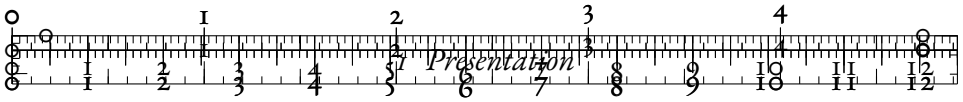


Figure 1.8: Capterra logo

Capterra Capterra is a free resource that helps businesses of all kinds compare available software and find the right software for their needs. It offers software ratings, reviews and buying guides.



Comparison table

Table 1.1 is an objective side-by-side comparison of our selected solutions. The code row specifies whether the software requires any technical knowledge to operate. The rating row is based on Capterra.

	Firebase	Airtable	Contentful	Sanity	Webflow CMS	Notion
Released	2016	2012	2016	2016	2012	2012
Type	Database	Spreadsheet	CMS	CMS	CMS	Wiki
Code	Yes	No	No	No	No	No
API	Yes	Yes	Yes	Yes	Yes	Beta
Rating	4.6	4.7	4.5	—	—	4.7

Table 1.1: Comparative table of the existing solutions

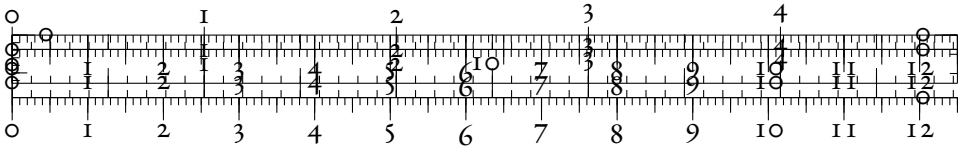
1.4.3 Critique

Multiple solutions are trying to focus on various use cases, however, all of them suffer from noticeable performance issues, a bad UX, and inadequate pricing for small and medium-sized businesses.

Notion is known for its slow performance and long loading times. Pages take on average between six and 12 seconds to load [2]. It also doesn't have an API, although one is being developed at the time of writing [2]. Furthermore, Notion is less structured than products like Airtable or Firebase.

Airtable is notable for its complexity, even for experienced users. It also suffers from some performance issues when loading large documents [1]. Furthermore, it doesn't have the same rich text capabilities as Notion. Finally, it lacks a real-time API, and it is relatively expensive [1].

[2] Notion's API was recently launched in beta.



Case study: Notion

Due to the popularity of Notion among various types of users, and due to the surge of sign-ups that it had seen, it seems to be the perfect product for a case study. While it is neither a database nor a CMS, it is often used as a limited content management solution for blogs.

Notion is a front-end React application, which means that the whole application has to be downloaded, parsed, and executed by the browser before it can load. Unfortunately, this happens nearly every time you visit their website [2]. On a fast 3G connection from Tunisia, the Notion page, depicted in figure 1.9, took 40 seconds to load. For the sake of comparison, the largest page ever on Wikipedia [4] took only 9 seconds to load.

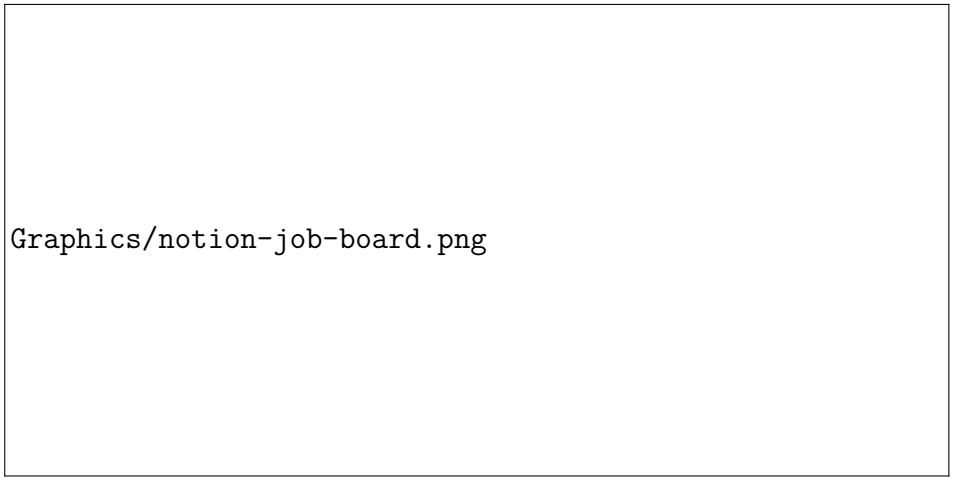
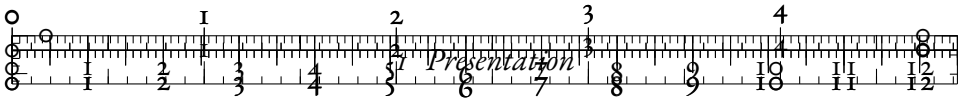


Figure 1.9: Notion page

While fetched resources can be cached for faster subsequent loads, processing JavaScript code and images is not cacheable and will always result in a slow application.



Further analysis shows that not much optimization has been done on Notion's part. The whole JavaScript code is loaded, regardless of whether it is needed on the requested page or not, unnecessary polyfills are loaded even for modern browsers, huge libraries are used—even when a smaller alternative exists—and images are served in their original format without optimization or minification.

The User Experience (UX), aside from the loading issues, is rather simple. Any person can start using Notion without any prior knowledge required.

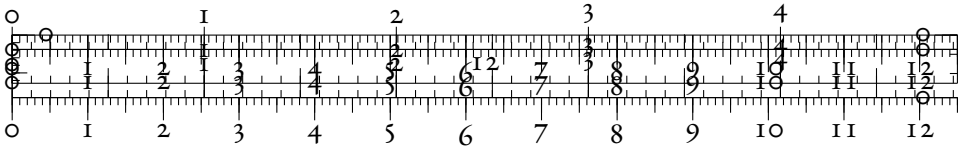
The takeaway from studying Notion is that performance should not be an afterthought. Instead, it should be part of every decision in the design process of our application. As for the interface, we should strive to keep it as simple and easy as Notion.

1.4.4 Proposed solution

Merebase is a collaborative visual database that can be used for data and content management. It's built with real-time collaboration, performance, and intuitiveness in mind. Thanks to years of innovation in the field of browser apps and high-performance real-time servers, it should be able to load instantaneously, while offering a smooth user experience with no glitching or slowdowns when loading large documents, and with the ability to effortlessly collaborate with other users.

1.5 Development process

To ensure the optimal use of time and energy, we chose to follow a development process throughout this project. That is, dividing work into smaller chunks according to a certain set of rules. In particular, we followed the principles of



Agile software development, which is an umbrella for multiple methodologies and frameworks. Of these methodologies, we adopted Feature-Driven Development (FDD) and the Kanban method for their practicality and ease-of-use, especially for projects with small teams.

1.5.1 Agile software development

Agile software development is an umbrella term for a set of frameworks and practices based on the set of principles popularized by the Manifesto for Agile Software Development in 2001. It advocates adaptive planning, evolutionary development, early delivery, and continual improvement, and it encourages flexible responses to change. It derives its values from a range of software development frameworks and methodologies.

Values

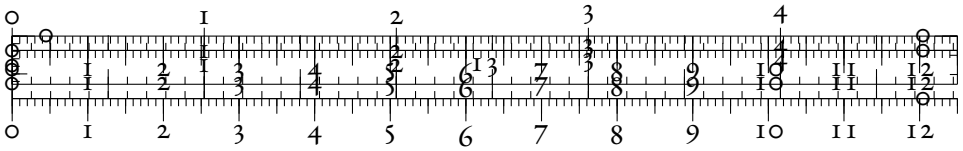
The Manifesto for Agile Software Development proclaims the following values:

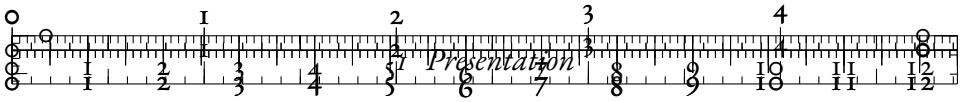
- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Principles

The Manifesto for Agile Software Development is based on twelve principles:

1. Customer satisfaction by early and continuous delivery of valuable software





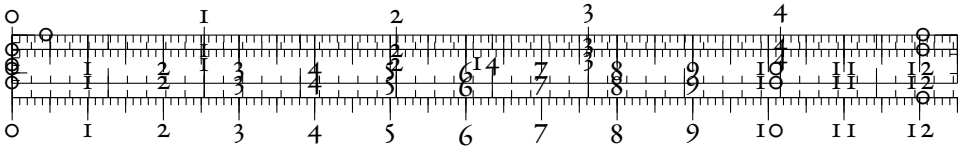
2. Welcome changing requirements, even in late development
3. Deliver working software frequently (weeks rather than months)
4. Close, daily cooperation between business people and developers
5. Projects are built around motivated individuals, who should be trusted
6. Face-to-face conversation is the best form of communication (co-location)
7. Working software is the primary measure of progress
8. Sustainable development, able to maintain a constant pace
9. Continuous attention to technical excellence and good design
10. Simplicity—the art of maximizing the amount of work not done—is essential
11. Best architectures, requirements, and designs emerge from self-organizing teams
12. Regularly, the team reflects on how to become more effective, and adjusts accordingly

1.5.2 Feature-Driven Development

Feature-Driven Development (FDD) is an Agile method for developing software iteratively and incrementally. It encourages planning, design, and development based on features.

1.5.3 Kanban

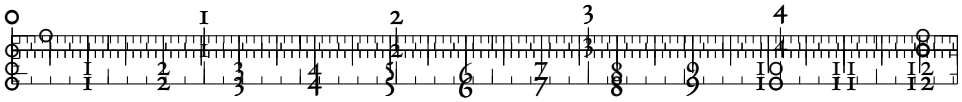
Kanban is an Agile method to manage work by balancing demands with available capacity, while uncovering bottlenecks. Work is divided into smaller tasks that are visualized on top of a Kanban board.



1.6 Conclusion

In summary, throughout this chapter, we have presented our project, the hosting institution, the motivations behind our choices, and our objectives. We also researched the existing solutions and we started drawing the picture for what our project strives to achieve.

Within the next chapter, we will be going into more details of this picture by focusing on the analysis and specification of needs for out project.



Acronyms

CmRDT Commutative Replicated Data Type

CMS Content Management Software

CRDT Conflict-free Replicated Data Type

CvRDT Convergent Replicated Data Type

DOM Document Object Model

FDD Feature-Driven Development

JSON JavaScript Object Notation

MVC Model-View-Controller

MVVM Model-View-ViewModel

OT Operational Transformation

P2P Peer-To-Peer

RBAC Role-Based Access Control

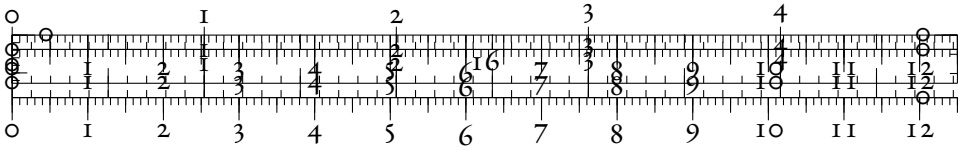
RDMBS Relational Database Management System

SaaS Software as a Service

SDK Software Development Kit

SSPL Server Side Public License

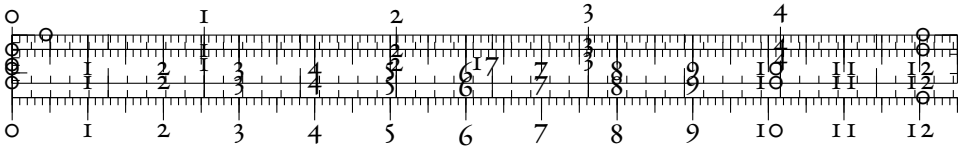
UI User Interface

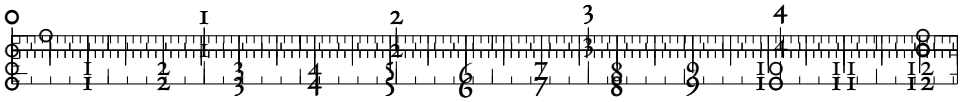


UX User Experience

WAI-ARIA Web Accessibility Initiative – Accessible Rich Internet Applications

WOOT WithOut Operational Transformation

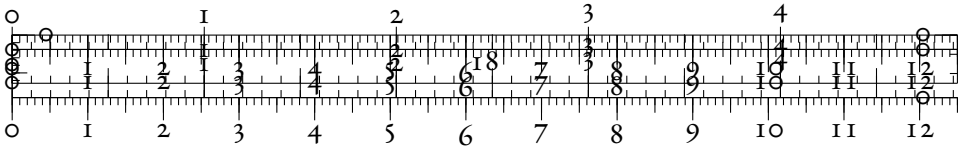


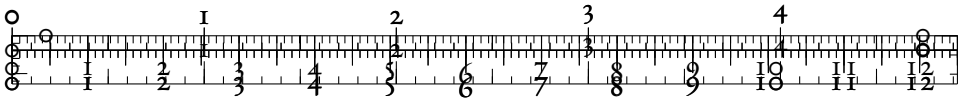


Glossary

cacheable A cacheable response is an HTTP response that is stored to be retrieved and used later, saving a new request to the server [3].

polyfill A piece of code used to provide modern functionality on older browsers that do not natively support it [5].





Bibliography

[1] *Airtable Pricing, Alternatives & More 2021 - Capterra*. URL: https://www.capterra.com/p/146652/Airtable/ (visited on 06/07/2021).

[2] I. Akulov. *Case study: Analyzing Notion app performance*. May 2020. URL: https://3perf.com/blog/notion/ (visited on 06/07/2021).

[3] *Cacheable - MDN Web Docs Glossary: Definitions of Web-related terms | MDN*. en-US. URL: https://developer.mozilla.org/en-US/docs/Glossary/cacheable (visited on 06/07/2021).

[4] *Long pages*. en. URL: https://en.wikipedia.org/wiki/Special:LongPages (visited on 06/07/2021).

[5] *Polyfill - MDN Web Docs Glossary: Definitions of Web-related terms | MDN*. en-US. URL: https://developer.mozilla.org/en-US/docs/Glossary/Polyfill (visited on 06/07/2021).