5-1-2023

Universal Back-End Design

Jason Kalili

Universal Back-End Design

by

Jason Kalili

A thesis presented to the

Faculty of the Department of
Computer Science
Loyola Marymount University

In partial fulfillment of the
Requirements for the Degree
Master of Science in Computer Science

May 1, 2023

I am submitting herewith a thesis written by Jason Ethan Kalili entitled "Universal Back-End Design." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science in Computer Science.

D. Yazdansepas

Dr. Delaram Yazdansepas, Thesis Advisor

We have reviewed the thesis and recommend its acceptance:

D. Yazdansepas

Dr. Delaram Yazdansepas, Thesis Committee Member

Robert WB Johnson

Dr. Robert Johnson, Thesis Committee Member

[illegible] 2023-05-10

Dr. John David N. Dionisio, Thesis Committee Member

TABLE OF CONTENTS

Acknowledgements

I wish to express my gratitude to Dr. Delaram Yazdansepas for her exceptional support and contributions as my advisor during the completion of my master's thesis. Her guidance was indispensable, and this project would not have been possible without her assistance.

I also want to extend my appreciation to B.J. Johnson and Dr. Dionisio for their roles as both thesis committee members and caring educators throughout my five-year academic journey.

To everyone involved, thank you.

Abstract

Accessibility in back-end development is often overlooked, with the majority of discussions and efforts centered on front-end design. To make applications usable for a wider audience, developers must also prioritize incorporating accessibility from the back-end. Back-end web accessibility encompasses the design and development of web-based systems and applications that are accessible to all users, including those with disabilities. This involves optimizing the underlying code and infrastructure for accessibility and implementing features that enable users with disabilities to navigate and interact with the site or application. Ensuring back-end web accessibility is crucial for creating an inclusive online environment accessible to everyone, regardless of their abilities. Presently, there is a significant gap in research regarding back-end accessibility specifics. This study investigates the challenges in implementing back-end accessibility, explores best practices, and aims to facilitate future research on its impact on user experience and system performance.

1. Introduction

Accessibility in software refers to the ability of digital systems, applications, and web-based platforms to be accessed and used by all individuals, regardless of their physical or cognitive abilities [12]. The goal of accessibility is to ensure that people with disabilities can effectively interact with digital content and services, and to promote inclusivity and equal access to information and technology. This is particularly important as society becomes increasingly reliant on technology for various aspects of daily life, from communication and education to entertainment and work.

Accessible software design involves taking into account the needs of users with disabilities, such as those with visual, hearing, or motor impairments and behavioral or emotional disabilities. For example, visually impaired users may require the use of screen readers or other assistive technologies to access digital content, while users with hearing impairments may require captioning or other forms of visual representation of audio content. To ensure accessibility, software developers follow established guidelines and standards, such as the Web Content Accessibility Guidelines[1] (WCAG) [16]. These guidelines provide specific recommendations for designing software that is accessible to users with disabilities, including recommendations for visual design, keyboard accessibility, and audio and video content.

Accessibility testing is also an important part of the software development process, as it allows developers to identify and address potential accessibility issues before the software is released to the public. This can involve the use of automated testing tools, as well as manual testing by individuals with disabilities who can provide feedback on the accessibility of the software. By engaging in accessibility testing, developers can ensure that their software meets

[1] https://www.w3.org/TR/WCAG21/

the needs of a diverse user base and complies with established accessibility standards. Despite the importance of accessibility in software design, there is a lack of research specifically focused on back-end accessibility. Nearly all research in this area has focused on front-end accessibility, which includes the visual and interactive components of software applications that users directly interact with. Front-end accessibility has been the primary focus, as these components are more visible to users and have a direct impact on their ability to access and use digital content.

As a result, many software development teams lack a comprehensive understanding of the requirements and best practices for ensuring accessibility in the back-end of their systems. Despite these challenges, it is crucial for software developers to prioritize back-end accessibility in their work.

To address this gap in research, there is a need for more studies focused specifically on back-end accessibility. This research explores various aspects of back-end accessibility, such as the challenges faced by software development teams in implementing accessibility, the best practices and guidelines for ensuring accessibility in the back-end, and the impact of back-end accessibility on the user experience for individuals with disabilities.

The topic of back-end accessibility in software is an important one, as it relates to the ability of individuals with disabilities to fully participate in the digital world. With the increasing reliance on technology in our daily lives, it is crucial for software developers to understand the importance of accessibility and to work towards creating inclusive and accessible digital systems for all users. By focusing on both front-end and back-end accessibility, developers can create software that is not only visually and interactively accessible, but also built on a foundation of inclusive design principles and practices.

2. Motivation

Despite the growing awareness of the importance of accessibility in the tech industry, many software development teams still struggle to implement back-end accessibility in their systems [1]. This is likely due to a lack of understanding of the requirements and best practices, as well as the added complexity and cost of ensuring accessibility in the back-end. Ensuring a robust and accessible back-end makes the internet a more inclusive space for users with disabilities, but can also benefit non-disabled users, as it can improve overall performance and security of the system [5].

This study aims to understand the challenges faced by software development teams in implementing accessibility, and to explore best practices and guidelines for ensuring accessibility in the back-end. This research can also examine the impact of back-end accessibility on the user experience for individuals with disabilities and how it can improve overall performance and security. The collaborative nature of software development can make it difficult to determine which teams are responsible for certain aspects. A specific guideline set for back-end accessibility can help ensure that accessibility is integrated into the back-end of all platforms. By establishing clear roles and responsibilities for back-end accessibility, software development teams can more effectively collaborate and coordinate their efforts, ultimately resulting in more inclusive and accessible digital systems.

In summary, this study seeks to address the challenges faced by software development teams in implementing back-end accessibility, explore best practices and guidelines, and potentially enable future researchers to examine the impact of back-end accessibility on user experience and system performance. This research aims to contribute to the ongoing efforts to make the digital world more inclusive and accessible for all users, regardless of their abilities.

2.1 The Importance a Universal Design for All Users

Accessibility is not only a matter of legal compliance and social justice but also a driving force behind the development of more usable applications for everyone. Universal Design is an approach to creating products, environments, and systems that are usable by as many people as possible, regardless of their age, ability, or status in life [17]. This concept seeks to eliminate the need for specialized solutions, instead opting for a single design that is adaptable, flexible, and inclusive.

By designing for diverse user needs, developers can create interfaces and systems that are easier to navigate and understand for all users. For example, incorporating larger and more legible fonts, as well as clear and consistent visuals, can enhance readability for users with visual impairments, while simultaneously benefiting older adults or those with cognitive challenges [18].

Accessible applications tend to be more flexible and adaptable, allowing users to tailor their experience according to their preferences and needs. This customization can enhance user satisfaction and facilitate more efficient interactions with the application. Adjustable text size, color contrast, and keyboard shortcuts not only benefit people with disabilities but also improve usability for non-disabled users who may have temporary impairments or situational limitations, such as operating a device in sunlight or with one hand [19].

Accessibility and Universal Design are essential in creating inclusive and equitable applications for individuals with disabilities, and also for enhancing the user experience for non-disabled users.

3. Literature/Interview Review

The history of web accessibility is a relatively short one, dating back to 1996 when the Web Accessibility Initiative (WAI) was conceived as a project by a few individuals from the W3C staff [6]. The issue of software accessibility only began gaining traction in 2006, and this is largely due to the high-profile court case against the Target Corporation. Target was sued by the National Federation of the Blind[2] (NFB) for violating the Americans with Disabilities Act[3] (ADA). The lawsuit was filed on behalf of blind individuals who could not access the company's website using screen readers. The NFB alleged that Target Corporation's website was not accessible to individuals with visual impairments. This did not only result in Target's legal obligation to integrate accessibility into their website, but also suggested that many other companies were not designing their software with accessibility in mind [15]. Since then, there has been a growing demand for software accessibility and an increasing awareness of the need to design software that is accessible to all individuals, including those with disabilities [14].

Finding reliable sources of information on back-end web accessibility can be challenging, as it is a relatively novel topic. To gather primary sources, the author conducted email and zoom interviews with front-end developers to understand how they believe back-end developers can build accessibility into joint applications from their perspective. The author also contacted back-end developers to learn about the practices they use to build accessibility into their platforms.

3.1 Front-End Design

In today's digital landscape, a company's online presence is extremely important. A website is the first impression that a visitor has of a company, so it is crucial that the website's structure and

[2] https://nfb.org/
[3] https://www.ada.gov/

design are attractive, robust, and clear, and that everything is working properly. Good front-end development can result in a better user experience by making navigation easier, which can keep the visitor engaged and interested. This is important for user retention, as a poor user experience can cause visitors to leave, resulting in a lost potential customer. Additionally, front-end development can help align branding and build trust with potential customers, which can boost brand awareness through word of mouth. While there is no specific research that directly supports the claim that effective front-end development can enhance business performance and subsequently drive sales and revenue growth, this notion is generally accepted as logical and plausible. By leveraging well-executed front-end development practices, companies can effectively communicate their value proposition, streamline user interactions, and create a lasting impression on customers [9]. Unlike literature that exclusively focuses on back-end accessibility, documentation and resources concerning front-end accessibility have become increasingly prevalent. Therefore, developers are now better equipped to ensure that their websites are accessible to all users, including those with disabilities, by adhering to established accessibility guidelines such as the aforementioned WCAG [16].

3.1.1 Perceivable

The visual design of a website or application is critical to its usability and appeal to users. Proper use of color, size, contrast, and other visual elements can make a website easier to navigate and more enjoyable to use. The size of visual elements should be carefully considered to ensure they are easily visible and don't overwhelm the user. Text, images, and other design elements should be appropriately sized for the device being used to view the website. The use of color can convey important information, create visual interest, and contribute to the overall aesthetic of a website. However, it is important to use colors that are easily distinguishable and avoid color combinations that may cause issues for users with color blindness. Proper contrast between background and foreground elements can help improve readability and reduce eye strain for users [11]. Contrast can also be used to highlight important information and call-to-action elements [16].

3.1.2 Operability

Interactions are the way users engage with the website or application, and they can greatly impact the user experience. Among other things, this includes buttons, user input fields, and navigation menus. Intuitive and well-designed interactions that can make the site more accessible and engaging [16]. WCAG recommends that all input fields should have visible labels that describe the purpose of the field. In addition to these guidelines, WCAG recommends that buttons and input fields should be easily navigable using the keyboard alone. This is important for users with motor disabilities who may not be able to use a mouse or touch screen to interact with the website [16].

3.1.3 Element Design

Proper element design makes a website more accessible and usable. Design considerations like HTML alt tags, hover over info, navigation, and tool tips can help users understand the purpose of different elements on the page and improve the overall user experience. Providing descriptive alternative tags for images and other visual elements can help users with visual impairments better understand the content of the page. Providing additional information through hover over text can help users understand the purpose of different elements on the page. Navigation should be intuitive and easy to use. Providing clear and concise navigation labels can help users find the information they are looking for more quickly. Tooltips can provide additional information about different elements on the page and help users understand the purpose of specific elements [16].

3.1.4 Integrations

Screen readers, third-party tools, and other accessibility technologies can help users with disabilities better interact with a website or application. These technologies can help users interact with an interface by providing alternative methods for accessing and navigating web content. Screen readers are designed to read out the contents of a website or application to users who are usually visually impaired. Screen readers rely on HTML attributes and proper labeling of buttons and input fields (refer to section 3.1.3). WCAG outlines specific recommendations for making content accessible to screen readers, and other assistive technology [16].

In order for third-party tools like web page translators or speech-to-text software to effectively integrate with a given webpage, they must meet specific requirements. These requirements often include adherence to the accessibility guidelines set forth by the WCAG[4], which ensures that digital content is compatible with a variety of assistive technologies.

3.1.5 Aesthetics

Contrary to what many people believe, adhering to best practices for front-end accessibility usually does not alter the appearance of an interface. Even when visual changes are necessary, they do not necessarily have to compromise the aesthetics of the design - that is ultimately the responsibility of the front-end designer. The goal of web accessibility is to create content that functions in a predictable and understandable way for every user, regardless of the methods they use to browse the web [12]. Apple[5] has long been recognized as a leader in usable and accessible technology, and their products are widely regarded as having some of the most accessible interfaces in the industry. Apple's commitment to accessibility is reflected in their development of specific accessibility features and guidelines, which are intended to make their products

[4] https://www.w3.org/WAI/tutorials/
[5] https://www.apple.com/

accessible to users with a wide range of disabilities. Despite their focus on accessibility, Apple has also managed to maintain an aesthetic that has enabled them to sell a combined 1.3 billion iPhones worldwide [2]. This combination of usability and aesthetics has helped to make Apple products popular among users of all abilities. As a result of their commitment to accessibility, Apple has become a model for other technology companies that are seeking to improve the accessibility of their own products. By providing clear and detailed human interface guidelines[6], Apple has made it easier for developers to design and implement accessible interfaces that are usable by all users, including those with disabilities [4][5]. Apple's success in balancing accessibility and aesthetics serves as an example to other tech companies looking to improve the accessibility of their developments. Their products highlight the importance of creating software that is usable for everyone, regardless of their ability level.

3.2 Back End Design

The back-end of a software application is a crucial component that often receives less attention than the front-end, even though it plays an essential role in ensuring seamless functionality and performance. By neglecting accessibility in the back-end, developers risk creating errors that can adversely impact user experience and limit the overall reach of the product. Despite its importance, there is a lack of research and information available on the topic of back-end design. To address this gap in knowledge, this study conducted interviews with experts in the field to gather information about the key considerations in back-end design. Through analysis of this data, the study reached conclusions about how back-end design can be leveraged to improve customer experience and increase software usability. Interestingly, many of the aspects that are traditionally associated with front-end design, such as screen readers, can be mirrored in the

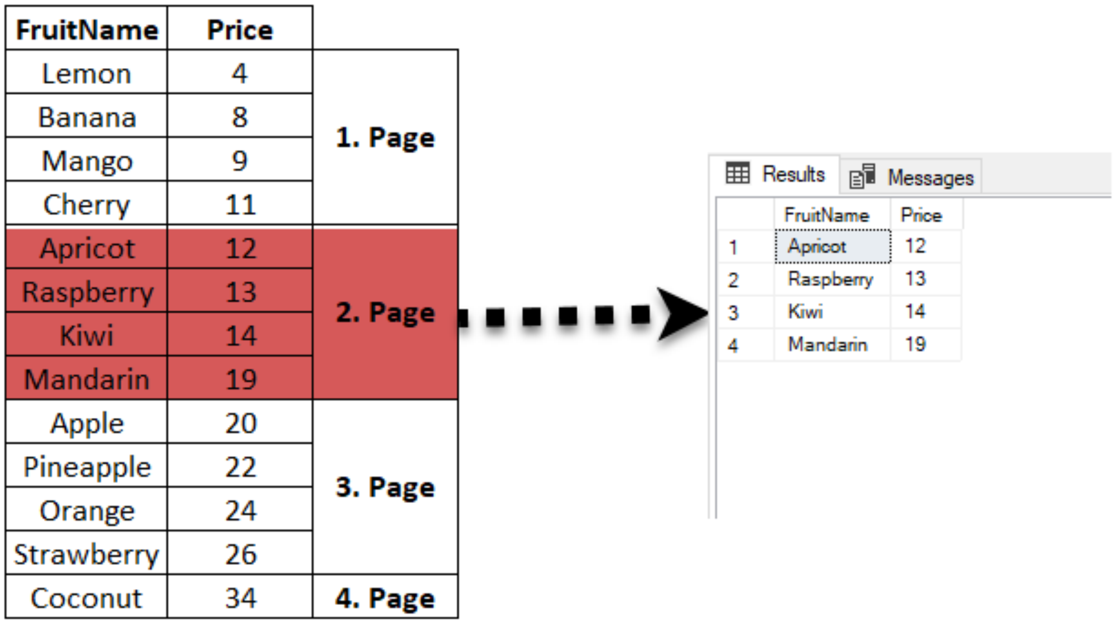
[6] https://developer.apple.com/design/human-interface-guidelines/guidelines/overview/

back-end. This paper will explore how back-end design can be optimized to facilitate the seamless integration of these front-end features, ultimately leading to a more effective and user-friendly software product.

3.2.1 Inconsistency

Inconsistency often arises when various components of a system do not align, such as when a backend lacks accessibility features, subsequently impacting the front-end user experience. Upon being questioned about this matter, an interviewee provided an explanation that in a project aimed at designing a dashboard to display organizational member information, a dense set of data was included, such as email, dates of last login, and more. Feedback revealed that presenting all the information at once made it challenging for users to filter and understand the data, even with adherence to WCAG guidelines. The project team identified the need to implement full-text search and pagination (Refer to Figure i for a more straightforward illustration of this concept [21]) on certain endpoints to improve the user experience, reduce information overload, and enhance accessibility [10].

In the context of localization, the most frequent issue encountered is non-localized messages originating from the back-end. This creates a challenge for localization on the front-end. While this issue may be related to accessibility, particularly if the back-end provides user interface elements, it has not yet been directly encountered [7].



FruitName	Price	
Lemon	4	1. Page
Banana	8	
Mango	9	
Cherry	11	
Apricot	12	2. Page
Raspberry	13	
Kiwi	14	
Mandarin	19	
Apple	20	3. Page
Pineapple	22	
Orange	24	
Strawberry	26	
Coconut	34	4. Page

	FruitName	Price
1	Apricot	12
2	Raspberry	13
3	Kiwi	14
4	Mandarin	19

Figure i. Simplified example of the pagination of data (for reducing information overload)

3.2.2 Pure Back-End

At a web security company where an expert worked, some crucial accessibility features depended primarily on the back-end, including translation and billing tools. The system retrieved translations based on the user's browser language preference and stored them locally. Additionally, the billing tool converted service prices to the user's local currency for easier payment.The application integrated with major identity providers like Apple[7] and Github[8], and billing services like Stripe[9] to enable users to use their existing credentials with the service. At the time, the system did not incorporate a purchasing power parity tool, which would have been a valuable addition for accessibility. This tool would have adjusted charges according to each

[7] https://www.apple.com/
[8] https://github.com/
[9] https://stripe.com/

country's economic status. For instance, a user in the USA might have paid $5 for a service, while a user in Mexico would have paid $0.50, and a user in Indonesia, $0.05 [10].

Another expert explained a back-end issue involving adding attributes or elements to the generated HTML, to determine the necessary information needed to fulfill a request. In a particular case, some requests were straightforward, requiring only the editing of the code that generates the HTML. However, in other cases, additional data was necessary, and as a result, the server-side renderer needed to be augmented with additional functions to retrieve and provide the required data [7].

3.2.3 Combining Front-End and Back-End

In the context of infrastructure accessibility, relying on a single point of failure can pose a significant risk. For instance, if the translation services of a company that primarily serves American customers are unavailable, they will be unable to cater to non-English speaking users. While tools such as translation are crucial for end-users, it is equally important to ensure the resilience of the entire infrastructure. In the event of a cache outage, the system's performance can be impacted, resulting in increased latency and an elevated risk of client timeouts. This can ultimately render the service inaccessible, emphasizing the need for a robust and resilient infrastructure to enhance accessibility [10].

3.2.4 Developer Accessibility

When it comes to server and database management, some tools can be challenging to use, which can be a barrier for end-users, especially developers. To enhance accessibility, selecting tools with excellent documentation and an active community ensures that there are adequate resources available to support individual users. Additionally, the platform allows for the use of third-party management tools with internal servers, databases, and more. For instance, tools like k9s[10] can make using Kubernetes[11] more accessible for individuals with dyslexia or color blindness than using kubectl[12]. Dedicated database clients such as Datagrip[13] or Datasette[14] can make it easier for developers to understand their databases than using the psql[15] command line interface. Allowing developers to customize their setup to their preferences is a crucial aspect of enhancing accessibility. Furthermore, the platform is kept open to allow other developers to use the services in the easiest way possible for them, which can improve accessibility by accommodating different workflows and preferences [10].

3.2.5 Testing and Validation

Testing and monitoring are critical for ensuring the reliability and performance of back-end integrations. Back-end developers need to implement testing and monitoring strategies to identify and fix issues that may arise during integration. This includes unit testing individual components, integration testing the entire system, and monitoring the integration in production to identify and address issues proactively [8].

Automated testing is an essential practice in ensuring accessibility by minimizing the effort required to interact with a system. For this reason, a platform should focus on system availability and integrity, and various measures should be taken to ensure that these aspects are not compromised. The two most important factors considered are the latency added by (in the interviewee's case) Cloudflare[16] and response size. Tests are conducted at every level of the stack

[10] https://k9scli.io/
[11] https://kubernetes.io/
[12] https://kubernetes.io/docs/reference/kubectl/
[13] https://www.jetbrains.com/datagrip/
[14] https://datasette.io/
[15] https://www.postgresql.org/docs/current/app-psql.html
[16] https://www.cloudflare.com/

to monitor traffic and prevent any blockage or drop [10]. To accomplish this, the platform conducts end-to-end (E2E) tests to simulate requests flowing through the network, and integration tests to ensure that the caches, queues, and databases are working together seamlessly. This multi-level approach ensures that the system is functioning as intended and that users can interact with it without experiencing any delays or other issues that could affect accessibility [10].

Currently, programming with awareness and utilizing a prioritization hierarchy are the methods employed for handling back-end accessibility testing and validation, based on information obtained from expert interviews [7][10]. There are automated scoring mechanisms like Google Lighthouse[17] for simple things, especially with HTML, however a manual audit is frequently necessary. A raw back-end testing framework has yet to exist, and issues like these are only handled by indirect tests through the front-end, with back-end rendered elements [7].

3.2.6 User Provided Input

In the context of ensuring comprehensive accessibility within software applications, one critical aspect that requires thorough investigation is the management of user-provided input on the backend. An interviewee explained that at a previous job, they worked on a tool that provided users with a search functionality that included an auto-completion feature. The tool relied on a user-metadata store operated by another team that contained customer-provided names, email addresses, and characteristics. Instead of hitting the database directly, the tool used a filtering API to enable pagination and query across the data. However, there was a lot of concern about the impact of the autocomplete functionality on the service, and the other team wanted the developer to only allow searches after the user had clicked a "Submit" button. This attention to

[17] https://developer.chrome.com/docs/lighthouse/overview/

detail significantly improved the user experience and was critical for accessibility, allowing all users the necessary time and control to input their search criteria and exit the process as soon as they found what they needed. This product decision led to a significant architectural decision [10].

3.2.7 API Endpoints

A key consideration for back-end integrations is API design. APIs are the standard way for different services to communicate with each other, and back-end developers need to design and implement APIs that are easy to use, secure, and scalable. This involves defining the API endpoints, request and response formats, and authentication and authorization mechanisms [2]. One of the best ways to ensure the accessibility of an API is by making sure it is open, complete, and free. When an API is well-documented and maintained, it allows developers to choose their own front-end, customizing everything from command-line arguments to mapped keyboard presses. An open, or public, API enables consumers to use the product in the way that is easiest for them, promoting accessibility [10].

3.2.8 Security and Accessibility

When implementing back-end accessibility, there is an increased risk for potential attacks, especially when adding new back-end services like a cache or database. To ensure security and minimize the risk of such attacks, it's crucial to follow best practices for distributed system development. For example, Cloudflare's Identity and Access Management team is responsible for all authentication and authorization of the control plane, with a focus on thorough testing of both the happy path (accessing what one should be able to) and the bad path (accessing what one shouldn't be able to). Additionally, the company has a dedicated Security Incident Response Team available at all times to help triage and address any security incidents that may arise [10].

The process of identifying and addressing vulnerabilities in a system is typically carried out by hiring a penetration tester, according to established security practices. Based on the findings of the tester, appropriate measures are taken to secure the system and close off any vulnerabilities that were identified. It is generally expected that developers have a basic understanding of common threats, such as code injection, to help prevent such issues from arising in the first place [10].

In terms of user testing, one professional mentioned that there have been cases where users with disabilities are asked to test front-ends that are generated by back-ends. In such cases, the testing process is similar to standard user testing, where users are observed while they interact with the system and feedback is gathered. However, it should be noted that such testing may not necessarily uncover all accessibility issues, particularly those that are not immediately apparent to users. Therefore, additional accessibility testing may be necessary to ensure that the system is fully accessible to all users [7].

3.2.9 Integrations

Integrations play a pivotal role in back-end development as they allow various systems and services to interact with each other, thus improving the functionality and user experience of web applications. When integrating with external systems, it's important to ensure that only authorized users and systems can access the application's data and functionality. Back-end developers need to implement authentication and authorization mechanisms to secure their APIs and prevent unauthorized access. This involves integrating industry-standard protocols like OAuth[18] or implementing custom authentication mechanisms, depending on the specific requirements of the integration. Error handling is another important consideration for back-end

[18] https://oauth.net/

integrations. Integrations can be complex, and errors can occur at any stage of the process. Back-end developers need to design robust error handling mechanisms to detect and handle errors that may arise during integration. The expert suggested logging errors, providing informative error messages, and implementing retry mechanisms to handle transient errors [10].

4. Conclusion

The lack of literature on back-end accessibility hindered the ability to establish a definitive and universally applicable back-end design. However, insight from expert interviews facilitated drawing recommendations on certain facets of back-end design and can serve as a basis for formulating a set of guidelines specifically tailored for the field of back-end development. The following recommendations were built upon the information gathered and reviewed:

4.1 Addressing Inconsistency

To enhance user experience and accessibility, back-end developers should implement full-text search and pagination on specific endpoints. This will help reduce information overload and improve overall user experience. Additionally, localization of messages from the back-end should be ensured to facilitate front-end localization efforts, making the application accessible to a broader audience.

4.2 Focusing on Pure Back-End Functionality

Utilize technologies like translation and billing tools to improve accessibility for a diverse user base. Consider implementing a purchasing power parity tool to charge users according to their country's economic status, promoting fairness and inclusivity.

4.3 Combining Front-End and Back-End Development

Ensure the resilience of the entire infrastructure to minimize the risk of single points of failure and maintain accessibility in case of service outages. A strong and robust infrastructure will benefit both front-end and back-end systems.

4.4 Enhancing Developer Accessibility

Select tools with excellent documentation and active communities to support individual users. Allow for the use of third-party management tools to cater to different workflows and preferences, and keep the platform open for customization. This flexibility will enable developers to create front-end experiences that meet diverse user needs.

4.5 Implementing Testing and Validation

Implement testing and monitoring strategies to ensure the reliability and performance of back-end integrations. Utilize automated testing to minimize user effort and conduct end-to-end tests and integration tests to ensure seamless functioning of the system, which will positively impact front-end experiences.

4.6 Prioritizing User Provided Input

Pay close attention to user experience and accessibility when designing features like autocomplete search functionalities. Make architectural decisions that prioritize user needs and preferences, ensuring that both front-end and back-end designs cater to users effectively.

4.7 Designing Accessible API Endpoints

Design and implement APIs that are easy to use, secure, and scalable. Ensure that APIs are open, complete, and well-documented to allow developers to customize their front-end interactions, promoting flexibility and adaptability in the system.

4.8 Balancing Security and Accessibility

Follow best practices for distributed system development and ensure thorough testing of both access paths. Use penetration testing to identify and address system vulnerabilities, and conduct additional accessibility testing when necessary. This ensures a secure system that remains accessible to users with varying abilities.

4.9 Streamlining Integrations

Implement authentication and authorization mechanisms to secure APIs and prevent unauthorized access. Design robust error handling mechanisms to detect and handle errors that may arise during integration, including logging errors, providing informative error messages, and implementing retry mechanisms for transient errors. This will contribute to a more reliable and resilient system, ultimately benefiting front-end user experiences.

5. Future Work

While there is a wealth of information on creating an accessible front-end, it is important to recognize that there is a significant knowledge and resource gap for back-end developers. Despite the important role back-end design plays in creating a universally accessible website or application, there are currently no specific guidelines that back-end developers must follow. Therefore, an important area of future research is to develop a comprehensive set of guidelines for back-end developers. These guidelines should address unique challenges and considerations related to back-end design, such as database management systems, server configuration, and API integration.

One possible approach to this problem is for an organization like WCAG to create a separate section specifically for back-end development. This section may contain specific guidelines and recommendations, and can cite relevant case studies and best practices from industry experts.

The development of a comprehensive set of back-end design guidelines will help ensure that websites and applications are universally accessible. This would represent a major step forward in the ongoing efforts to promote web-accessibility of all users.

5.1 Limitations of the Study

One limitation of this study was the lack of differentiation between good code and accessible code. The theory of universal design suggests that good design is accessible design, and vice versa. However, it is possible that there are aspects of good code that are not related to accessibility, and vice versa. Thus, a clearer differentiation between good code and accessible code could help in identifying the unique aspects of each and developing more comprehensive design guidelines that address both accessibility and overall code quality.

This study acknowledges a number of limitations that impact the interpretation of its findings on back-end accessibility and design. Firstly, the small sample size consisting of full-stack developers (which was a convenience sample) may not accurately represent the broader population of software engineers, thereby limiting the generalizability of the study's results. A wider range of interviews was challenging due to the time constraints of the study.

Secondly, the potential for self-report bias due to data collection through interviews needs to be mentioned. The limited literature available on back-end accessibility and design also poses a major challenge, as most of the information was derived from these interviews. Although efforts were made to incorporate diverse perspectives, the findings might not be comprehensive and could be influenced by interviewees' biases.

While the present study offers valuable insights into back-end accessibility and design, its limitations warrant careful consideration when interpreting the findings. To build upon and expand the current insights, future research must involve larger sample sizes, more diverse participants, and a wider focus on accessibility and design in software engineering.

6. References

[1] "2020 Full Year Report – ADA Digital Accessibility Lawsuits," Accessed April 17, 2023.

[2] Amazon. "What Is an API? - Application Programming Interfaces Explained - AWS." Accessed April 17, 2023. https://aws.amazon.com/what-is/api/.

[3] Apple. "Human Interface Guidelines - Human Interface Guidelines - Design - Apple Developer." Accessed April 17, 2023. https://developer.apple.com/design/human-interface-guidelines/guidelines/overview/.

[4] Apple. "Layout - Foundations - Human Interface Guidelines - Design - Apple Developer." Accessed April 11, 2023. https://developer.apple.com/design/human-interface-guidelines/foundations/layout/.

[5] Bureau of Internet Accessibility. "Accessibility Is Privacy and Security." Accessed April 11, 2023. https://www.boia.org/blog/accessibility-is-privacy-and-security.

[6] Dardailler, Daniel. "WAI History." Accessed April 17, 2023. https://www.w3.org/WAI/history.

[7] Dionisio, John David. Universal Backend Design Interview. Email, March 4, 2023.

[8] IBM. "What Is Software Testing and How Does It Work? | IBM." Accessed April 17, 2023. https://www.ibm.com/topics/software-testing.

[9] Inc, Novateus, and Shahzaib Munawar. "Top 8 Reasons Why Front End Development Is Important For Your Business Success." Novateus, March 17, 2022. https://novateus.com/blog/top-8-reasons-why-front-end-development-is-important-for-your-business-success/.

[10] Mobbs, Ian. Universal Backend Design Interview. Email, January 25, 2023.

[11] Okabe, Masataka, and Kei Ito. "Color Universal Design (CUD) / Colorblind Barrier Free," November 20, 2002. https://jfly.uni-koeln.de/color/.

[12] Petrie, Helen, Andreas Savva, and Christopher Power. “Towards a Unified Definition of Web Accessibility.” In Proceedings of the 12th International Web for All Conference, 1–13. W4A ’15. New York, NY, USA: Association for Computing Machinery, 2015. https://doi.org/10.1145/2745555.2746653.

[13] Shvartsman, Daniel. “Apple Inc.: Our Modern-Day Symbol of Knowledge, Power, and Wealth.” Investing.com. Accessed April 17, 2023. https://www.investing.com/academy/statistics/apple-facts/.

[14] TheExpressWire. “Worldwide Demand For Website Accessibility Software Market Is Growing At A CAGR of 11.3% By 2028 | 98 Report Pages.” Digital Journal, May 31, 2022. https://www.digitaljournal.com/pr/worldwide-demand-for-website-accessibility-software-market-is-growing-at-a-cagr-of-11-3-by-2028-98-report-pages.

[15] W3C Web Accessibility. “Target Corporation - A Cautionary Tale of Inaccessibility | Web Accessibility Initiative (WAI) | W3C.” W3C Web Accessibility Initiative (WAI). Accessed April 17, 2023. https://www.w3.org/WAI/business-case/archive/target

[16] W3C Web Accessibility. “WCAG 2 Overview.” Web Accessibility Initiative (WAI). Accessed April 11, 2023. https://www.w3.org/WAI/standards-guidelines/wcag/.

[17] Connell, B. R., Jones, M., Mace, R., Mueller, J., Mullick, A., Ostroff, E., ... & Vanderheiden, G. (1997). The principles of universal design. North Carolina State University.

[18] Czaja, Sara J., Neil Charness, Arthur D. Fisk, Christopher Hertzog, Sankaran N. Nair, Wendy A. Rogers, and Joseph Sharit. “Factors Predicting the Use of Technology: Findings From the Center for Research and Education on Aging and Technology Enhancement (CREATE).”

Psychology and Aging 21, no. 2 (June 2006): 333–52. https://doi.org/10.1037/0882-7974.21.2.333.

[19] Newell, Alan F., and Peter Gregor. "'User Sensitive Inclusive Design'— in Search of a New Paradigm." In Proceedings on the 2000 Conference on Universal Usability, 39–44. CUU '00. New York, NY, USA: Association for Computing Machinery, 2000. https://doi.org/10.1145/355460.355470.

[20] W3C Web Accessibility. "Tutorials." Web Accessibility Initiative (WAI). Accessed April 27, 2023. https://www.w3.org/WAI/tutorials/.

[21] Figure i. From "Pagination in SQL Server." by Erkec, Esat. SQL Shack - Articles about Database Auditing, Server Performance, Data Recovery, and More (blog), April 14, 2020. https://www.sqlshack.com/pagination-in-sql-server/.