communicated by:
Lehr- und Forschungsgebiet Informatik 9

Prof Dr. Ulrik Schroeder

The present work was submitted to Learning Technologies Research Group
Diese Arbeit wurde vorgelegt am Lehr- und Forschungsgebiet Informatik 9

Accessibility and Usability Enhancements for Educational Gamebooks

Verbesserungen der Zugänglichkeit und Benutzerfreundlichkeit von Spielbüchern für den Unterricht

Master-Thesis
Masterarbeit

presented by / von

Shanker, Ritwik
441957

Prof. Dr.-Ing. Ulrik Schroeder

Dr. rer. nat. Svenja Noichl

Aachen, February 26, 2025

Contents

List of Figures

List of Listings

Abstract

In an era of digital transformation, educational gamebooks have emerged as innovative platforms that combine narrative-driven content with interactive learning experiences. However, despite their potential to engage diverse audiences, many such platforms neglect critical aspects of accessibility and usability, thereby excluding users with disabilities. This thesis addresses the gap by enhancing the accessibility and usability of DiGaBo, an educational gamebook platform developed at RWTH Aachen University, through the integration of advanced, user-centered design principles. The research is driven by the central question of how to effectively implement accessibility features in gamified environments.

A mixed-methods approach was adopted, incorporating a comprehensive literature review, heuristic evaluation, and iterative design cycles. The methodology involved a detailed analysis of existing accessibility standards, such as WCAG 2.1, and an exploration of theoretical frameworks to underpin the design rationale. Key technical interventions included the development of a multilanguage support system using the ngx-translate library, the creation of a customizable theming engine offering five distinct visual modes (Light, Dark, High Contrast, Sepia, and Low Animation) tailored to diverse visual needs, and the implementation of speech-to-text and text-to-speech functionalities leveraging the Web Speech API. These features were rigorously tested using both automated tools (e.g., WAVE and Lighthouse) and manual assessments with assistive technologies to ensure compliance and practical usability.

The results demonstrate a significant improvement in both the accessibility and usability of the platform. Automated testing revealed enhanced compliance with accessibility standards. The findings highlight that integrating comprehensive accessibility features within gamified educational platforms fosters a more inclusive digital learning environment. Ultimately, this research provides a replicable framework for developers and educators, offering practical guidelines for embedding accessibility into the design of educational technologies, and underscores the broader imperative of ensuring equal access to educational opportunities in the digital age.

Chapter 1 Introduction

1.1 Background and Context

In the rapidly advancing technological landscape today, the synergy between education and technology is reshaping traditional learning paradigms. Digital platforms have emerged as key tools that offer innovative solutions to make learning more engaging, accessible, and personalised. Figure 1.1 illustrates the steadily increasing adoption of online education platforms. Within this domain, educational gamebooks have gained prominence for their ability to combine narrative-driven content with interactive elements, fostering an immersive learning experience. An example of such a platform is **DiGaBo**, developed by Dr. rer. nat. Svenja Noichl at RWTH Aachen University, which exemplifies the integration of narrative and interactivity in educational contexts. Despite their potential, these platforms often do not address key aspects of accessibility and usability, leaving significant user groups underserved.

Online Education Growth

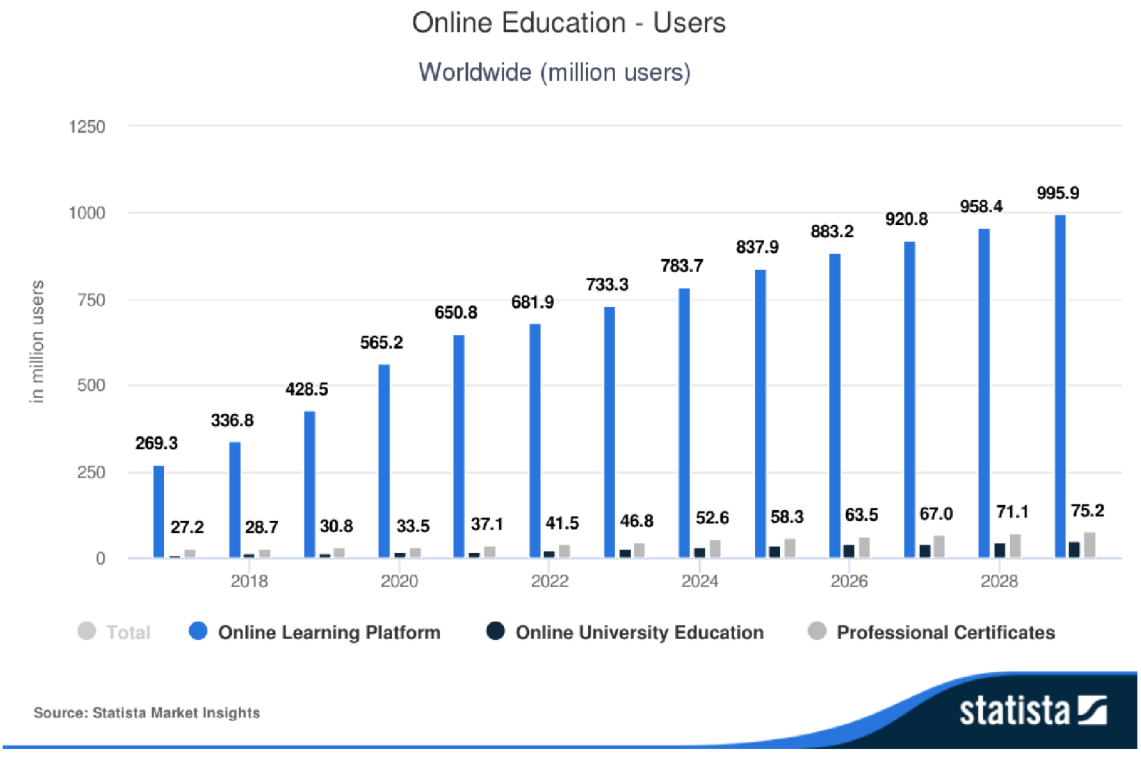

Figure 1.1: Online Education - Total Number of Users [Sta]

Accessibility, as defined by the World Wide Web Consortium's Web Content Accessibility Guidelines (WCAG)[1], refers to the ability of a system to be usable by individuals with disabilities, encompassing visual, auditory, motor and cognitive impairments. Usability, on the other hand, is the ease with which users can navigate and interact with a platform. Both are essential for creating inclusive educational tools that cater to a diverse audience, including young learners, elderly users, and individuals with disabilities.

Defining Accessibility and Usability

The critical need for accessible and user-friendly educational tools has been amplified by the global shift towards digital learning, further accelerated by recent global events. Platforms that do not address accessibility and usability risk alienating users and hindering their educational progress. This thesis, therefore, aims to bridge the gap between gamified educational platforms and the principles of accessibility and usability.

Inclusivity Imperative

1.2 Motivation

The motivation behind this thesis stems from the recognition that accessibility and usability are not just desirable features, but essential components of any educational tool intended for diverse audiences. As education becomes increasingly digital, it is crucial to ensure that these tools are designed with inclusivity in mind. The ultimate goal is to ensure that educational gamebooks can serve as effective learning tools for everyone, regardless of their abilities or background.

Educational gamebooks have the potential to revolutionise learning by making it more interactive and engaging. This thesis aims to bridge the gap between the engaging nature of gamified learning and the need for accessibility and usability in educational tools. By focussing on these aspects, the thesis seeks to create a more inclusive learning environment that can benefit a wider range of users.

1.3 Problem Statement

Although gamified educational platforms have shown promise in improving engagement and knowledge retention, they often neglect the needs of diverse user groups. This is particularly problematic for users with disabilities, who often face significant barriers when interacting with digital learning tools. For example, Elderly users may find themselves excluded due to complex interfaces, users with visual impairments may struggle with poorly designed interfaces that do not support screen readers, while those with motor disabilities may find it difficult to navigate platforms that lack proper keyboard support. **Existing solutions in the field tend to address either gamification or accessibility in isolation, leaving a void where these aspects intersect.**

The current state of educational gamebooks, exemplified by platforms such as DiGaBo, highlights this issue. While DiGaBo's gamified nature appeals to many users, its lack of comprehensive accessibility features, such as keyboard navigation, screen reader compatibility, and multi-language support, limits its reach and effectiveness. Without addressing these shortcomings, the educational potential of DiGaBo remains restricted, and its promise of inclusivity is unfulfilled.

[1] WCAG 2.1, `https://www.w3.org/TR/WCAG21/`

1.4 Objectives

The primary objective of this thesis is to enhance the accessibility and usability of educational gamebooks, making them more inclusive for diverse user groups, to achieve this, the thesis focuses on the following specific objectives:

1. **Identify Accessibility Challenges:** Conduct a detailed analysis of our DiGaBo gamebook to pinpoint barriers faced by users with diverse needs.
2. **Implement Accessibility Features:** Implement a range of accessibility features, including keyboard navigation, screen reader compatibility, high-contrast modes, and text-to-speech functionality.
3. **Introduce Multilanguage Support:** Create a robust language framework to make the platform more accessible to non-native speakers, which would be achieved by using localization files to replace hardcoded strings and enable seamless language switching.
4. **Conducting Comprehensive Testing:** Apply user-centered design principles and usability heuristics to ensure the platform is intuitive, user-friendly and aligns the development process with established accessibility standards to ensure the platform's inclusivity and test if all the features meet the required standards. This involves both automated and manual testing, as well as cross-browser and cross-platform testing to ensure consistency across different devices and browsers.
5. **Documenting the Process:** Document the entire process, from the initial research and development to the final testing and evaluation. This documentation will serve as a guide for future developers who wish to create accessible and usable educational tools.

1.5 Scope of the Study

The scope of this thesis encompasses the development and evaluation of accessibility and usability enhancements within the context of educational gamebooks. Specifically, this study focuses on enhancing DiGaBo, a platform developed at RWTH Aachen University, but the findings and recommendations can be applied to other digital learning tools too. It focuses on enhancing the accessibility and usability of DiGaBo, using it as a case study to demonstrate how modern web technologies can be used to create inclusive learning environments. Key areas of focus include:

1. **Technological Framework:** Leveraging Angular and Node.js to implement accessibility features such as theme customization and localization.
2. **Enhanced Features:** Introducing five customizable themes (Light, Dark, High Contrast, Sepia, and Low Animation) using WCAG-compliant colours, as well as a speech-to-text functionality for text input, a text-to-speech functionality and introducing multilanguage support and multiple other features.
3. **Testing and Evaluation:** Utilizing automated tools and heuristic evaluations to ensure compliance with accessibility standards and optimal user experience.
4. **Documentation:** Providing comprehensive guidelines and documentation to facilitate the adoption and further development of the enhanced platform.

1.6 Methodology

The methodology for this thesis is divided into four main phases: research and familiarization, development, testing and evaluation, and documentation and thesis writing which has also been represented in Figure 1.2.

1. **Research and Familiarization:** The first phase involves conducting a comprehensive literature review on accessibility, usability, and gamification in educational platforms. This phase also includes familiarizing with WCAG (Web Content Accessibility Guidelines) standards, usability heuristics, and relevant frameworks. The goal of this phase is to develop a theoretical framework that will guide the design and evaluation processes in later phases.
2. **Development:** The development phase focuses on implementing accessibility and usability features in the educational gamebook platform. This includes implementing keyboard navigation, screen reader compatibility, high-contrast modes, text-to-speech functionality, and multi-language support. The development phase also involves redesigning forms and interactive controls to ensure they are fully accessible.
3. **Testing and Evaluation:** The testing and evaluation phase involves conducting comprehensive testing to ensure that all accessibility and usability features meet the required standards. This includes both automated and manual testing, as well as cross-browser and cross-platform testing. The goal of this phase is to identify and address any issues that may arise during testing.
4. **Documentation and Thesis Writing:** The final phase involves documenting the entire process, from the initial research and development to the final testing and evaluation. This includes writing the thesis, preparing detailed code documentation, and creating user and developer guides. The goal of this phase is to provide a comprehensive resource for future developers who wish to create accessible and usable educational tools.

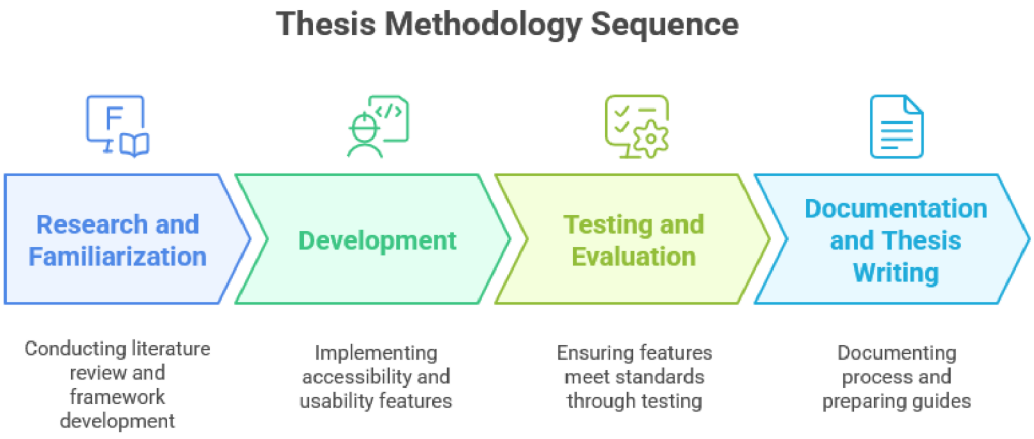

Figure 1.2: Thesis Methodology

1.7 Structure of the Thesis

This thesis is organized into multiple chapters, each addressing a critical aspect of the research:

1. **Introduction:** This chapter provides an overview of the thesis, including the background, motivation, problem statement, objectives, scope, methodology, and contributions. It also discusses the theoretical framework that guides the design and evaluation processes.
2. **Literature Review:** An exploration of existing research on accessibility, usability, and gamification in educational platforms. It also discusses the theoretical framework that guides the design and evaluation processes.
3. **Methodology:** This chapter provides a detailed description of the methodology used in the thesis, including the research and familiarization, development, testing and evaluation, and documentation and thesis writing phases.
4. **Implementation:** This chapter describes the implementation of accessibility and usability features in the educational gamebook platform. It includes detailed technical descriptions of the features implemented, as well as the challenges encountered during the development process.
5. **Evaluation and Results:** An analysis of the testing outcomes, highlighting the platform's compliance with accessibility standards and its usability improvements.
6. **Discussion and Conclusion:** A synthesis of the findings, their implications, and recommendations for future research, including potential improvements such as audio descriptions for video content.

Through these chapters, this thesis aims to provide a comprehensive narrative of the challenges, solutions, and insights encountered in enhancing the accessibility and usability of educational gamebooks. By achieving these goals, the study seeks to make a meaningful contribution to the ongoing efforts of creating inclusive digital learning environments.

1.8 Conclusion

The outcomes of this thesis hold significant implications for the fields of e-learning, web development, and accessibility engineering. By addressing accessibility and usability within DiGaBo, this work contributes to creating tools that are not only engaging but also inclusive. The findings and methodologies derived from this research can serve as a blueprint for developers and educators aiming to design more equitable learning environments.
In addition, this study aligns with the broader goal of education as a universal right, highlighting the importance of equal access to knowledge regardless of physical, cognitive, or technical limitations. By making educational gamebooks more accessible, this thesis aspires to extend their benefits to a wider audience, thereby fostering a more inclusive learning ecosystem.

Chapter 2 Literature Review

This comprehensive literature review explores the intersection of gamification, accessibility, and usability in the context of educational technologies, with a specific focus on educational gamebooks. The review aims to provide a solid foundation for understanding the potential of gamified, accessible, and usable educational tools in enhancing learning experiences for diverse student populations.

2.1 Gamification in Education

Gamification, the application of game-design elements and game principles in non-game contexts, has emerged as a powerful tool in educational settings over the past decade [DDKN11]. This section explores the concept of gamification, its theoretical underpinnings, and its effectiveness in educational contexts.

2.1.1 Definition and Basic Concepts

Gamification in education refers to the integration of game elements such as points, badges, leaderboards, narratives, and challenges into learning environments [Kap12]. The primary goal is to increase student engagement, motivation, and ultimately, learning outcomes by leveraging the intrinsic appeal of games. Key elements of gamification in education include:

Gamification Defined

- **Points and Scoring Systems:** Used to quantify progress and achievements.
- **Badges and Achievements:** Visual representations of accomplishments.
- **Leaderboards:** Competitive elements that showcase relative performance.
- **Levels and Progression:** Structured advancement through content or skills.
- **Narratives and Storytelling:** Contextualizing learning within engaging storylines.
- **Challenges and Quests:** Goal-oriented tasks that promote active learning.
- **Feedback Mechanisms:** Immediate and constructive responses to user actions.

These elements, when thoughtfully implemented, can transform traditional learning experiences into more engaging and interactive processes.

2.1.2 Theoretical Frameworks

Several theoretical frameworks underpin the use of gamification in education, providing a solid foundation for its implementation and study:

Self-Determination Theory (SDT)

Self-determination theory, developed by Ryan and Deci, posits that intrinsic motivation is driven by three basic psychological needs: autonomy, competence, and relatedness [RD00]. Gamification can address these needs in the following ways:

- **Autonomy:** Providing choices in learning paths and decision-making opportunities.
- **Competence:** Offering progressive challenges and skill-building opportunities.
- **Relatedness:** Incorporating social elements and collaborative tasks.

A study by Sailer et al. (2017) found that specific game design elements can satisfy psychological needs, with badges, leaderboards, and performance graphs positively influencing competence need satisfaction [SHMM17].

Flow Theory

Developed by Csikszentmihalyi, Flow Theory describes an optimal state of immersion and focus where an individual is fully absorbed in an activity [Csi90]. In educational gamification, the goal is to create this state by:

- Balancing challenge and skill level
- Providing clear goals and immediate feedback
- Eliminating distractions and promoting concentration

Gamification Theories

Research by Hamari and Koivisto (2014) demonstrated that gamification can induce flow experiences, leading to increased enjoyment and engagement in learning activities [HK14].

Theory of Gamified Learning

Proposed by Landers, the Theory of Gamified Learning suggests that gamification can positively impact both instructional content and learning outcomes by influencing learner behaviour and attitudes [Lan14]. This theory posits that:

- Gamification elements can directly affect learning outcomes
- Gamification can moderate the relationship between instructional design and learning outcomes
- The effectiveness of gamification depends on both the quality of the educational content and the appropriateness of the gamification elements

Landers and Landers (2014) found that gamification elements, when aligned with learning objectives, can significantly improve time-on-task and learning outcomes [LL14].

2.1.3 Effectiveness of Gamification in Education

Numerous studies have investigated the effectiveness of gamification in educational contexts. A meta-analysis by Sailer and Homner (2020) of 82 independent samples found that gamification in education has a significant positive effect on cognitive, motivational, and behavioural learning outcomes [SH20].

Key findings from various studies include several important aspects. Gamification elements such as points, badges, and leaderboards have been shown to increase student engagement and participation in learning activities [DSdNDM+13]. A study by Hanus and Fox (2015) found that gamification can enhance intrinsic motivation, particularly when it promotes autonomy and competence [HF15]. Research by Tsay et al. (2018) demonstrated that gamification can lead to improved academic performance, especially in STEM subjects [TKL18]. Additionally, gamified learning environments have been shown to promote the development of 21st-century skills such as problem-solving, critical thinking, and collaboration [QC16].
However, it's important to note that the effectiveness of gamification can vary depending on factors such as the specific gamification elements used, the context and subject matter, the characteristics of the learners, the duration of the gamified experience, and the quality of implementation.
For example, a study by Hanus and Fox (2015) found that certain gamification elements, such as compulsory participation in leaderboards, can have negative effects on motivation and academic performance [HF15]. This highlights the importance of thoughtful design and implementation of gamification strategies.

2.1.4 Challenges and Considerations in Educational Gamification

While gamification shows promise in education, several challenges and considerations must be addressed. Over-gamification can be an issue, as excessive use of game elements can distract from learning objectives [TVI18]. Individual differences play a role, as not all students respond equally to gamification, necessitating personalized approaches [BD17]. Sustainability is another concern, as the novelty of gamification may wear off over time, requiring strategies to maintain long-term engagement [KH19]. Ethical concerns, such as data privacy and the potential for manipulation, need to be carefully considered [KSLB18].
Future research in educational gamification should focus on addressing these challenges and developing best practices for effective implementation across diverse learning contexts.

2.2 Accessibility in Educational Technologies

Accessibility in the context of educational technologies refers to the design of digital learning tools and environments that can be used by all students, including those with disabilities. This section explores the importance of accessibility, key guidelines, and strategies for implementation.

2.2.1 Definition and Basic Concepts

Accessibility Defined

The World Wide Web Consortium's Web Content Accessibility Guidelines (WCAG) defines accessibility as the ability of a system to be usable by individuals with disabilities, including visual, auditory, motor, and cognitive impairments [W3C18]. In educational technology, accessibility ensures that all students, regardless of their abilities, can access and interact with digital learning materials and tools.
Key principles of accessibility, as outlined by WCAG 2.1, include:

- **Perceivable:** Information and user interface components must be presentable to users in ways they can perceive.
- **Operable:** User interface components and navigation must be operable.
- **Understandable:** Information and the operation of the user interface must be understandable.
- **Robust:** Content must be robust enough that it can be interpreted reliably by a wide variety of user agents, including assistive technologies.

2.2.2 Importance of Accessibility in Educational Technologies

Ensuring accessibility to educational technologies is crucial for several reasons:

Legal Compliance

Many countries have laws requiring educational institutions to provide accessible learning materials and technologies. For example:

- In the United States, Section 508 of the Rehabilitation Act and the Americans with Disabilities Act (ADA) mandate accessibility in educational institutions.
- The European Union's Web Accessibility Directive requires public sector websites and mobile applications, including those in education, to be accessible.

Equal Opportunities

Accessible technologies ensure that all students, regardless of their abilities, have equal access to educational content and opportunities [Sea13]. This is particularly important in the context of online and distance learning, where digital technologies play a central role in education delivery.

Universal Design for Learning (UDL)

The Universal Design for Learning framework emphasizes the importance of providing multiple means of engagement, representation, and action/expression to accommodate diverse learners [CAS18]. Accessibility is a key component of UDL, as it ensures that learning materials can be accessed and used by students with various abilities and learning styles.

2.2.3 Implementing Accessibility in Educational Technologies

Implementing accessibility in educational technologies involves addressing various types of disabilities and providing appropriate accommodations:

Visual Accessibility

For students with visual impairments, key considerations include

- **Screen Reader Compatibility:** Ensuring that all text content can be read aloud by screen reading software.

- **Alternative Text:** Providing descriptive text alternatives for images, graphs, and other visual elements.
- **Color Contrast:** Using sufficient colour contrast for text and important visual elements.
- **Resizeable Text:** Allowing users to resize the text without loss of functionality. [W3C18]

Auditory Accessibility

For students with hearing impairments:

- **Captions:** Providing accurate captions for all audio content, including videos and podcasts.
- **Transcripts:** Offering text transcripts for audio-based learning materials.
- **Visual Alerts:** Using visual cues in addition to auditory signals [W3C18].

Motor Accessibility

For students with motor impairments:

- **Keyboard Navigation:** Ensuring all functionality is accessible via keyboard input.
- **Large Click Targets:** Providing sufficiently large and well-spaced interactive elements [W3C18].
- **Alternative Input Methods:** Supporting various input devices, such as switch controls or eye-tracking systems.

Cognitive Accessibility

For students with cognitive or learning disabilities:

- **Clear Structure:** Organizing content in a logical, consistent manner.
- **Simplified Language:** Using clear, concise language and avoiding jargon.
- **Multimedia Support:** Providing multiple representations of information (text, audio, video).
- **Customizable Interface:** Allowing users to adjust settings such as text size, colour schemes, and content density [W3C18].

2.2.4 Challenges in Implementing Accessibility

Despite the importance of accessibility, several challenges exist in its implementation. Lack of awareness is a significant issue, as many educators and developers may not be fully aware of accessibility requirements or best practices [LFH17]. Technical complexity poses another challenge, as implementing accessibility features can be technically challenging, especially for complex interactive content [CCJ12]. Resource constraints are also a concern, as retrofitting existing content for accessibility can be time-consuming and costly [Sea13]. Additionally, there can be perceived conflicts between accessible design and visually appealing design, creating a challenge in balancing accessibility and aesthetics [PHK09].

2.2.5 Future Directions in Accessibility Research

Future research in accessibility for educational technologies should focus on several key areas. There is a need for developing more efficient tools and methodologies for implementing accessibility features. Investigating the impact of accessibility on learning outcomes for students with and without disabilities is another important direction. Exploring the intersection of accessibility and emerging technologies such as virtual and augmented reality in education will be crucial as these technologies become more prevalent in educational settings. Finally, addressing the unique accessibility needs in specific educational domains, such as STEM subjects or language learning, will be essential for ensuring inclusive education across all disciplines.

2.3 Usability in Educational Technologies

Usability refers to the ease with which users can navigate and interact with a platform or tool. In the context of educational technologies, usability is crucial for ensuring that students can focus on learning content rather than struggling with the interface.

2.3.1 Definition and Basic Concepts

ISO 9241-11 defines usability as "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use" [Int18]. In educational technology, usability encompasses:

Usability Defined

- **Learnability:** How easy it is for students to accomplish basic tasks the first time they encounter the interface.
- **Efficiency:** How quickly students can perform tasks once they have learned the design.
- **Memorability:** How easily students can re-establish proficiency after a period of not using the system.
- **Errors:** How many errors students make, how severe these errors are, and how easily they can recover from them.
- **Satisfaction:** How pleasant it is to use the design.

2.3.2 Usability Heuristics in Educational Contexts

Jakob Nielsen's 10 Usability Heuristics for User Interface Design provide a framework for evaluating and improving usability [Nie94]. These heuristics can be adapted for educational technologies:

- **Visibility of system status:** Keep students informed about their progress and current location within the learning material.
- **Match between system and the real world:** Use language and concepts familiar to students, following real-world conventions.
- **User control and freedom:** Provide clear navigation options and allow students to undo actions or exit unwanted states.
- **Consistency and standards:** Follow platform conventions and maintain consistency across the learning environment.

- **Error prevention:** Eliminate error-prone conditions or provide confirmation options before students commit to an action.
- **Recognition rather than recall:** Make objects, actions, and options visible to minimize cognitive load.
- **Flexibility and efficiency of use:** Provide shortcuts for experienced users while still guiding novices.
- **Aesthetic and minimalist design:** Avoid cluttering the interface with irrelevant information.
- **Help users recognize, diagnose, and recover from errors:** Provide clear error messages and suggest constructive solutions.
- **Help and documentation:** Offer easily accessible help and documentation focused on the student's task.

2.3.3 Importance of Usability in Educational Technologies

Usability in educational settings is crucial for several reasons. Learning efficiency is enhanced as a usable interface allows students to focus on content rather than struggling with the technology, leading to more efficient learning [ZP09]. Well-designed interfaces minimize extraneous cognitive load, allowing students to dedicate more cognitive resources to the learning material [SVMP98]. User-friendly educational technologies are more likely to keep students engaged and motivated [Kel10]. Studies have shown a positive correlation between usability and learning outcomes in e-learning environments [ACDM+06]. Additionally, usable designs often overlap with accessible designs, making educational content more inclusive for all learners [Bur15].

2.3.4 Usability Evaluation in Educational Technologies

Evaluating usability in educational technologies involves various methods and approaches:

Heuristic Evaluation

Experts evaluate the interface against established usability principles. In educational contexts, this may include additional heuristics specific to learning, such as learner control, allowing students to control their learning pace and sequence; feedback, providing timely and constructive feedback on student actions and progress; and consistency with learning objectives, ensuring the interface supports and aligns with the intended learning outcomes [SP00].

User Testing

Involving actual students in usability testing can provide valuable insights. Methods include think-aloud protocols, where students verbalize their thoughts as they navigate the learning environment; task analysis, observing students as they complete specific learning tasks; and eye-tracking, analysing students' visual attention patterns during interaction with the interface [TA08].

Questionnaires and Surveys

Standardized usability questionnaires can be adapted for educational contexts. Examples include the System Usability Scale (SUS), a quick, reliable tool for measuring perceived usability [Bro96]; the Website Analysis and Measurement Inventory (WAMMI), which assesses user satisfaction with web-based systems [Kir98]; and the Questionnaire for User Interaction Satisfaction (QUIS), which evaluates specific interface aspects [CDN88].

2.3.5 Challenges in Usability for Educational Technologies

Despite its importance, implementing usability in educational technologies faces several challenges. Educational technologies often need to cater to a wide range of users with varying levels of technical proficiency and learning styles [ZP09]. Balancing simplicity and functionality can be difficult, as educational software often requires complex features to support various learning activities, which can conflict with the need for a simple, intuitive interface [ACDM+06]. The fast-paced evolution of technology can make it challenging to maintain usability standards across different platforms and devices [Nie12]. Additionally, educational institutions may lack the resources or expertise to conduct thorough usability evaluations and implement improvements.

2.3.6 Future Directions in Usability for Educational Technologies

As educational technologies continue to evolve, several areas warrant further research and development. These include developing adaptive interfaces that can adapt to individual student needs and preferences [BM07], exploring usability considerations for virtual reality, augmented reality, and artificial intelligence in education [Ded09], investigating how cultural differences impact usability perceptions and needs in global educational contexts [Hal04], and studying how usability impacts student engagement and learning outcomes over extended periods [Nie12].

2.4 Intersection of Gamification, Accessibility, and Usability in Educational Technologies

The convergence of gamification, accessibility, and usability in educational technologies presents both opportunities and challenges. This section explores how these three domains interact and influence each other in the context of educational gamebooks and similar learning tools.

Figure 2.1 illustrates the intersection of gamification, accessibility, and usability in educational technologies. These three domains interact in various ways to enhance user engagement, inclusivity, and ease of use. While gamification improves motivation and learning outcomes, accessibility ensures that digital platforms are usable by all individuals, including those with disabilities. Usability principles further refine the user experience, ensuring that both gamified and accessible features are intuitive and efficient to use.

Figure 2.1: Conceptual Relationship Between Gamification, Accessibility, and Usability

2.4.1 Synergies and Tensions

Gamification and Accessibility

Gamification can potentially enhance engagement for users with disabilities, but it also introduces new accessibility challenges [Sea13].

- **Opportunities:** Gamified elements like achievements and progress tracking can provide additional motivation for learners with disabilities [GCdMGL17].
- **Challenges:** Visual or time-based game elements may create barriers for users with visual impairments or motor disabilities [YFHJ11].

Usability and Accessibility

Usability and accessibility often go hand in hand, as many usability principles also improve accessibility. However, there can be instances where they conflict [PHK09].

- **Complementary Aspects:** Clear navigation, consistent layout, and error prevention benefit both usability and accessibility [W3C18].
- **Potential Conflicts:** Some accessibility features, such as high contrast modes or text-only versions, may impact the aesthetic appeal or functionality preferred by non-disabled users [LFH17].

2.4.2 Integrated Approaches

To effectively combine gamification, accessibility, and usability in educational technologies, several integrated approaches have been proposed:

Universal Design for Learning (UDL)

UDL principles provide a framework for designing educational experiences that are accessible, engaging, and usable for all learners [CAS18].

- **Multiple Means of Representation:** Presenting information in various formats to accommodate different learning styles and abilities.
- **Multiple Means of Action and Expression:** Allowing learners to demonstrate their knowledge in diverse ways.
- **Multiple Means of Engagement:** Using gamification elements to motivate and involve all learners.

Inclusive Game Design

This approach focuses on creating game-like educational experiences that are inherently accessible and usable for a wide range of learners [YFHJ11].

- **Customizable Interfaces:** Allowing users to adjust game elements, such as speed, difficulty, or input methods.
- **Multi-modal Feedback:** Providing game feedback through visual, auditory, and haptic channels.
- **Flexible Narratives:** Designing branching storylines that accommodate different player abilities and preferences.

User-Centered Design (UCD)

UCD involves users throughout the design process to ensure that educational technologies meet the needs of diverse learners [AMKP04].

- **Participatory Design:** Involving students, educators, and accessibility experts in the design process.
- **Iterative Testing:** Continuously evaluating and refining the design based on user feedback.
- **Persona Development:** Creating detailed user profiles to guide design decisions and ensure inclusivity.

2.4.3 Challenges in Integration

Despite the potential benefits, integrating gamification, accessibility, and usability in educational technologies presents several challenges:

- **Technical Complexity:** Implementing accessible and usable gamified features often requires advanced technical skills and resources [CCJ12].

- **Balancing Act:** Finding the right balance between engaging game elements, accessibility features, and usable interfaces can be difficult [DDKN11].
- **Diverse User Needs:** Catering to the wide range of abilities, preferences, and learning styles present in educational settings is challenging [Bur15].
- **Evaluation Metrics:** Developing comprehensive metrics to assess the effectiveness of integrated approaches in terms of engagement, accessibility, and learning outcomes is complex [SHMM17].

2.4.4 Future Research Directions

As the fields of gamification, accessibility, and usability continue to evolve, several areas warrant further investigation:

- **Adaptive Gamification:** Exploring how gamification elements can dynamically adapt to individual user needs and preferences.
- **Cross-cultural Perspectives:** Examining how cultural differences impact the perception and effectiveness of gamified, accessible, and usable educational technologies.
- **Long-term Impact:** Conducting longitudinal studies to assess the sustained effects of integrated approaches on learning outcomes and user engagement.
- **Emerging Technologies:** Investigating the implications of virtual reality, augmented reality, and artificial intelligence for gamification, accessibility, and usability in education.

2.5 Conclusion

The integration of gamification, accessibility, and usability in educational technologies, particularly in the context of educational gamebooks, presents exciting opportunities and significant challenges. Using the synergies between these domains and addressing potential tensions, educators and designers can create more engaging, inclusive, and effective learning experiences. As research in these areas continues to evolve, it is crucial to maintain a user-centred approach that considers the diverse needs of all learners. Future developments in adaptive technologies, inclusive design practices, and comprehensive evaluation methods will play a vital role in shaping the next generation of educational technologies that are not only engaging and usable but also accessible to all.

Chapter 3 Methodology

This chapter outlines the research design, methods, and approaches used in developing and evaluating the accessible educational gamebook platform. The methodology is grounded in the principles of universal design and web accessibility, with a focus on creating an inclusive learning experience for all users, including those with disabilities.

3.1 Research Design

The research follows a mixed-methods approach, combining qualitative and quantitative methods to address the research questions comprehensively. This design allows for a thorough exploration of both the technical implementation of accessibility features and their impact on user experience and learning outcomes.

3.1.1 Research Questions

The methodology is designed to address the following primary research questions:

Research Focus

1. How can accessibility features be effectively implemented in an educational gamebook platform?
2. What is the impact of these accessibility features on user experience and engagement?
3. How does the implementation of accessibility features affect the overall usability of the platform?

3.1.2 Approach

The research adopts an iterative, user-centered design approach, consisting of the following phases:

1. **Initial analysis and requirements gathering:** This phase involved a comprehensive exploration of existing literature, accessibility guidelines, and user needs to inform the design and implementation of accessibility features. Specifically, this included:
    - Conducting a thorough review of the WCAG 2.1 guidelines to understand the technical requirements for web accessibility [W3C18].
    - Reviewing existing literature on accessibility and usability in e-learning environments to identify best practices and common challenges [Sea13].

- Performing a heuristic evaluation of the existing DiGaBo platform to identify potential accessibility issues.

The outcome of this phase was a detailed set of requirements and design considerations that guided the subsequent phases of the research.

2. **Design and implementation of accessibility features:** Based on the requirements gathered in the initial phase, this phase focused on the design and implementation of specific accessibility features in the DiGaBo platform. This included:

   - Implementing a theming system to allow users to customize the visual appearance of the platform.
   - Integrating the ngx-translate library to provide multilanguage support.
   - Implementing speech-to-text and text-to-speech functionality to improve accessibility for users with motor and visual impairments.
   - Enhancing keyboard navigation and screen reader compatibility throughout the platform.
   - Ensuring all content met WCAG 2.1 colour contrast requirements.

   Accessibility Features

   This phase involved iterative cycles of design, coding, and testing to ensure that each feature was implemented effectively and met the identified accessibility requirements.

3. **Usability testing and evaluation:** This phase focused on evaluating the effectiveness of the implemented accessibility features and their impact on user experience. This involved:

   - Conducting automated accessibility testing using tools such as WAVE and Lighthouse to identify potential accessibility violations.
   - Performing manual testing with assistive technologies (e.g., screen readers, keyboard-only navigation) to identify usability issues.

   The results of this phase were used to inform further refinements and improvements to the platform.

This iterative approach allows for continuous improvement and ensures that the final product meets the needs of diverse users [AMKP04]. The feedback from each phase informed the subsequent phase, creating a cycle of refinement and enhancement. This user-centered approach ensured that the accessibility features were not only technically sound but also met the real-world needs of the target audience.

3.2 Implementation of Accessibility Features

The implementation of accessibility features in the gamebook platform focused on five main areas: Multilanguage support, Themes, speech-to-text functionality, text-to-speech functionality, and general accessibility improvements. The following subsections detail the methods and rationale for each area.

3.2.1 Multilanguage Support

To make the platform accessible to a wider audience and comply with WCAG 2.1 guideline 3.1.2 (Language of Parts) [W3C18], multilanguage support was implemented. The process involved:

1. Extracting all hardcoded strings from the codebase
2. Creating language JSON files (en.json and de.json) to store translations
3. Implementing a translation service using the ngx-translate library

Choice of Translation Library and Comparison with Alternative Libraries

After careful consideration, the ngx-translate library was chosen for implementing multilanguage support. While there are several options available for Angular applications, ngx-translate stood out due to its robust features, extensive community support, and seamless integration with Angular [ntt].
To justify this choice, we considered two other popular internationalization libraries for Angular: Angular's built-in i18n [tea] and Transloco [JSV]. Each option presented unique advantages and drawbacks that were carefully weighed against the specific needs and constraints of the DiGaBo platform.
Angular's built-in i18n solution offers the advantage of being an official Angular product, ensuring long-term support and compatibility. Its compilation-based translation approach also provides excellent performance and strong typing support. However, this approach necessitates separate builds for each supported language. For DiGaBo, this would significantly increase build times and deployment complexity, which was deemed impractical given the iterative nature of the development process and the desire for rapid updates. Furthermore, Angular's i18n has limited runtime language switching capabilities. The ability to switch languages on the fly without a page reload was deemed crucial for maintaining a seamless user experience in the gamebook environment, making this a significant drawback.
Transloco presents a modern API with excellent TypeScript support, along with features like lazy loading of translations, translation scoping, and fallback languages. These features align well with modern development practices and offer flexibility in managing translations. However, Transloco is a relatively new library with a smaller community compared to ngx-translate. This posed a risk in terms of ongoing support and the availability of community-contributed resources. Additionally, the setup and configuration of Transloco were found to be more complex than ngx-translate, potentially increasing the development overhead.
Ultimately, ngx-translate was selected for its pragmatic balance of features, ease of use, and community support. Its large and active community ensures good support and regular updates, which is critical for a long-term project like DiGaBo. The simple and intuitive API made it easy to implement and use, minimizing the learning curve for the development team. Crucially, ngx-translate supports runtime language switching without a page reload, a key requirement for maintaining a seamless user experience. While it may not be as type-safe as Angular's built-in solution or offer all the advanced features of Transloco, the benefits of ngx-translate in terms of ease of use and community support outweighed these drawbacks for the DiGaBo project. The

availability of extensive documentation and numerous examples further solidified its suitability for our needs.
Compared with the alternatives, the ngx-translate library allows for easy addition of new languages and ensures that users can access content in their preferred language, enhancing comprehension and user experience.

3.2.2 Theming

To enhance visual accessibility and accommodate user preferences, a flexible theming system was developed, offering five distinct themes:

- Light theme
- Dark theme
- High Contrast theme
- Sepia theme
- Low Animation theme

Rationale for a Custom Theming Solution

While existing third-party libraries provide pre-built theming solutions, they often impose rigid structures and include unnecessary overhead. Many generic theming libraries lack comprehensive accessibility features tailored to users with diverse needs. By designing a custom theming system, this implementation ensures:

- Full compliance with WCAG 2.1 guidelines for contrast, readability, and accessibility.
- A lightweight approach with only the necessary styles, optimizing performance and reducing dependency bloat.
- Greater flexibility in modifying or expanding themes as per evolving user needs.
- Seamless integration with the existing architecture, ensuring smooth application-wide theme transitions.

Theme Characteristics and User Benefits

The DiGaBo platform's theming system was designed with inclusivity in mind, offering five distinct visual configurations tailored to different user needs. Each theme addresses specific accessibility requirements while maintaining aesthetic appeal:
The **Light Theme** serves as the default configuration, optimized for general use in well-lit environments. Its neutral colour palette and high contrast ratios ensure readability for the majority of users. The balanced combination of white backgrounds with dark text minimizes eye strain while maintaining a clean, professional appearance. This theme particularly benefits users who work in bright environments or require minimal visual adjustments.
For users in low-light conditions, the **Dark Theme** provides a more comfortable viewing experience by reducing screen brightness. This configuration is particularly beneficial for users with light sensitivity or those working in dimly lit spaces. The dark background with light text also helps conserve battery life on OLED displays, making it practical for mobile users. The visual hierarchy remains clear through careful contrast ratios between text and interactive elements.

The **High Contrast Theme** was developed specifically for users with visual impairments. By amplifying the contrast between text and background elements, this theme improves readability for individuals with low vision or colour blindness. The stark colour combinations meet WCAG 2.1 accessibility standards, ensuring that content remains distinguishable even for users with limited visual acuity. This theme also benefits users who require enhanced focus on textual content without visual distractions.
The **Sepia Theme** introduces a warmer colour palette that reduces glare and eye fatigue. This configuration is particularly suitable for users who experience discomfort with stark black-and-white contrasts. The muted, earthy tones create a softer visual experience while maintaining sufficient contrast for readability. This theme is ideal for users who prefer a more relaxed aesthetic or have light sensitivity issues.
Finally, the **Low Animation Theme** minimizes motion-based effects to accommodate users with vestibular disorders or motion sensitivity. By reducing animations, this theme creates a more stable interface that avoids triggering discomfort from excessive movement. The static layout ensures predictable interactions, making it particularly beneficial for users who require a calm and predictable visual environment. This configuration aligns with accessibility guidelines that recommend reducing unnecessary animations to improve usability.

Implementation Strategy

The process of integrating theming within the application involved the following key steps:

1. **Code Cleanup**: Eliminating hardcoded colours from stylesheets to ensure centralized theme management.
2. **Centralized Color Definition**: Defining colour variables in a single file to facilitate easy modifications and ensure consistency across the themes.
3. **Theme Management Service**: Developing a custom `ThemeService` in Angular to manage theme switching dynamically and store user preferences.
4. **CSS Class Implementations**: Creating dedicated CSS classes for each theme and applying them based on user selection.

Ensuring Accessibility Compliance

The colour palettes and contrast ratios were meticulously chosen to align with WCAG 2.1 standards. Specifically:

- Contrast ratios were tested to meet or exceed the minimum requirement of 4.5:1 for normal text and 3:1 for large text.
- The high contrast theme was verified to ensure distinct separation of UI elements, particularly for users with colour vision deficiencies.
- The low animation mode adheres to WCAG recommendations for reducing motion that could trigger vestibular disorders [W3C18].

This tailored theming approach enhances the overall usability of the application, ensuring inclusivity for a broad spectrum of users while maintaining a high degree of customization and performance efficiency.

3.2.3 Speech-to-Text Functionality

To improve accessibility for users with motor impairments or those who prefer voice input, speech-to-text functionality was implemented for all text input fields. This feature aligns with WCAG 2.1 guideline 2.1.1 (Keyboard) by providing an alternative input method [W3C18].
The implementation leverages the Web Speech API [1] to facilitate real-time speech recognition within the application. This approach offers a customizable solution without relying on external libraries, ensuring seamless integration and control over the functionality.
Key steps in the development process include:

1. **Initialization of Speech Recognition**: Utilizing the `webkitSpeechRecognition` object, the application initializes the speech recognition service, setting parameters such as language preference and result handling mechanisms.
2. **Language Configuration**: The system retrieves the user's language preference from local storage or defaults to the browser's language setting, supporting both English and German locales.
3. **Event Handling**: Event listeners are established to manage various states of the speech recognition process, including capturing results, handling errors, and monitoring the start and end of speech input.
4. **User Interface Feedback**: The application provides real-time feedback to users, indicating active listening status and displaying transcribed text, thereby enhancing user engagement and confidence in the system's responsiveness.

Accessibility Considerations

Implementing speech-to-text functionality offers substantial benefits:

- **Assistance for Motor Impairments**: Users with conditions that limit manual dexterity can input text through voice commands, reducing dependence on traditional input devices.
- **Enhanced Usability**: Voice input can streamline interactions, making the application more user-friendly and efficient for a broader audience.
- **Compliance with Accessibility Standards**: By providing alternative input methods, the application adheres to established accessibility guidelines, promoting inclusivity.

Browser Compatibility and Limitations

The Web Speech API's support varies across browsers:

- **Chromium-Based Browsers**: Browsers like Google Chrome and Microsoft Edge offer robust support for the Web Speech API, enabling full functionality of the speech-to-text feature.
- **Mozilla Firefox**: Currently, Firefox does not support the speech recognition component of the Web Speech API [moz].

[1] `https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API`

- **Safari**: Safari provides partial support, primarily for speech synthesis, with limited functionality for speech recognition.

Given these disparities, users are advised to utilize Chromium-based browsers to fully experience the speech-to-text capabilities.

Offline Functionality

The Web Speech API typically relies on cloud-based services for processing speech input, necessitating an active internet connection. While certain platforms offer on-device speech recognition, such implementations often require specific configurations and may not be universally supported [Ram20]. Therefore, the current implementation is optimized for online use, ensuring access to the most accurate and up-to-date speech recognition services.

Conclusion

By integrating a custom speech-to-text feature using the Web Speech API, the application enhances accessibility and user experience without the overhead of external libraries. This approach provides a tailored solution that accommodates users' diverse needs, aligns with accessibility standards, and maintains flexibility for future enhancements.

3.2.4 Text-to-Speech Functionality

The text-to-speech (TTS) feature was implemented to further enhance the accessibility of the platform by providing an alternative mode of content consumption. This functionality is particularly beneficial for users with visual impairments, reading difficulties, or cognitive disabilities, as well as for those who prefer to receive information audibly. By converting on-screen text into natural-sounding speech, the TTS feature supports diverse learning styles and improves overall usability.

Overview and Rationale

In developing this feature, a custom solution was preferred over existing third-party libraries. While libraries offer pre-built components, they may introduce unnecessary dependencies, lack fine-grained control over the speech output, or fail to seamlessly integrate with the existing system architecture. The custom implementation leverages the native Web Speech API, which provides direct access to the browser's speech synthesis engine. This approach results in a lightweight, efficient, and easily maintainable solution.

One of the significant advantages of this approach is the ability to dynamically adjust the spoken language based on user preferences. The system retrieves the user's language setting from local storage or by detecting the browser's default language and applies this setting to the speech synthesis engine. This ensures that the speech output is delivered in the native accent and intonation corresponding to the selected language. In cases where the desired language is not explicitly set or available, the implementation defaults to a standard accent (e.g., British English with the code `en-GB`). Such adaptability is critical in a multilingual context and enhances the inclusivity of the application.

Benefits and Accessibility Impact

The TTS functionality contributes significantly to the accessibility of the platform in several ways:

- **Enhanced Accessibility for Disabled Users**: For users with visual impairments or reading disabilities, the ability to listen to the content can dramatically improve the user experience. It reduces reliance on visual input, making it easier to navigate and comprehend the material.
- **Support for Multilingual Audiences**: By dynamically setting the language based on user preferences or browser settings, the TTS feature ensures that non-native speakers can hear content in their preferred language. This localization of audio output contributes to a more inclusive experience for a diverse user base.
- **Improved Cognitive Accessibility**: Users who experience fatigue or cognitive overload from reading lengthy texts can benefit from auditory delivery. The spoken word can often be easier to digest, particularly in educational contexts or when the content is dense.
- **Seamless Integration**: The custom-built nature of the feature means that it integrates smoothly with the overall application architecture. This eliminates compatibility issues that might arise with external libraries and ensures that the TTS functionality is robust and adaptable.

Potential Enhancements and Future Directions

While the current implementation meets the essential accessibility and usability requirements, there is scope for future improvements. Potential enhancements include:

- **Customization Options**: Allowing users to further customize parameters such as speech rate, pitch, and volume could provide a more personalized experience.
- **Voice Selection Interface**: Implementing a user interface for selecting from multiple available voices would enable users to choose the voice that they find most pleasant or easiest to understand.
- **Context-Aware Speech**: Future iterations could explore context-aware speech synthesis, where the system intelligently pauses, adjusts emphasis, or changes intonation based on the structure and importance of the content.

Conclusion

The text-to-speech functionality described in this section represents a critical advancement in making digital content accessible to all users. By utilizing the browser's native speech synthesis engine, the implementation is both lightweight and highly adaptive, catering to multilingual audiences and providing a natural, native accent based on the user's language settings. The feature not only complies with accessibility standards such as WCAG 2.1 but also significantly enhances the user experience for individuals with disabilities. Through ongoing enhancements and refinements, this TTS system has the potential to serve as a cornerstone for accessible web applications, ensuring that information is available in multiple formats to meet the diverse needs of its user base.

3.2.5 General Accessibility Improvements

Ensuring accessibility in web applications is essential to creating an inclusive digital environment for all users, including those with disabilities. In addition to the specific accessibility features discussed in previous sections, several general improvements were systematically implemented. These improvements enhance usability for individuals who rely on assistive technologies and improve overall user experience by ensuring compliance with established accessibility standards.

Heading Structure for Screen Readers

A well-defined heading structure is critical for users who navigate through content using screen readers. Many assistive technologies rely on proper heading hierarchies (H1, H2, H3, etc.) to provide an organized and logical reading order. This methodology involved:

- Structuring content with semantic HTML elements to improve document readability.
- Ensuring that headings are used consistently and in the correct order to convey content hierarchy.

ARIA Labels for Interactive Elements

The implementation of ARIA attributes was driven by the need to bridge the gap between dynamic web content and assistive technologies like screen readers. As highlighted in research on web accessibility, ARIA (Accessible Rich Internet Applications) provides a standardized way to make interactive elements understandable to users who rely on assistive technologies [Sea13]. The methodology involved three key strategies:

1. **Meaningful Descriptions via `aria-label`:** Buttons, links, and form elements were enhanced with `aria-label` attributes to provide context that screen readers could interpret. For example, a "Submit" button might have an `aria-label` like "Submit registration form" to clarify its purpose beyond generic text. This approach aligns with best practices for improving screen reader compatibility [Sea13].

2. **Reducing Verbiage with `aria-hidden`:** Decorative elements like icons or background images were flagged with `aria-hidden` to prevent screen readers from announcing non-essential content. This decision was informed by research emphasizing the importance of semantic markup and reducing cognitive load for assistive technology users [Sea13].

Keyboard Navigation for All Interactive Elements

Keyboard accessibility was prioritized to ensure users with motor impairments or those preferring keyboard navigation could interact fully with the platform. The implementation strategy focused on:

1. **Tab Navigation Accessibility:** All interactive elements were made reachable via the `Tab` key, ensuring users could navigate without a mouse. This aligns with WCAG guidelines for keyboard navigation.
2. **Logical `tabindex` Order:** The `tabindex` attribute was used to define a natural flow of focus, mirroring the visual hierarchy of the interface. This approach was validated through testing with keyboard-only navigation tools, as recommended in accessibility literature.
3. **Key Activation Support:** Interactive elements were configured to respond to `Enter` and `Space` keys, enabling activation without mouse clicks. This implementation addressed findings from studies on keyboard navigation barriers in web applications.
4. **Focus Management:** Testing revealed issues like missing focus rings or inconsistent focus states, which were resolved by implementing CSS focus styles and ensuring proper focus trapping in modal dialogs. This process was informed by research on improving keyboard navigation usability.

Text Alternatives for Non-Text Content

To support users who rely on screen readers, alternative text descriptions were provided for all non-text content, including images, icons, and multimedia elements. The methodology for text alternatives involved:

- Adding `alt` attributes to all meaningful images to describe their function or content.
- Providing detailed captions for data visualizations, graphs, and charts.
- Ensuring that decorative images were appropriately marked with empty `alt` attributes to avoid unnecessary verbosity.

These general accessibility improvements were implemented in adherence to WCAG 2.1 guidelines and best practices in web accessibility [W3C18]. By systematically addressing structural, navigational, and visual accessibility barriers, the platform ensures an inclusive and user-friendly experience for all individuals, regardless of their abilities.

3.3 Evaluation Methods

To assess the effectiveness of the implemented accessibility features and their impact on the overall user experience, a multi-faceted evaluation approach was adopted. Given the constraints of limited funding and the inability to recruit a substantial user base of individuals with disabilities, the evaluation primarily relied on online automated tools and simulated manual testing. This section describes the evaluation strategy in detail, including the tools used, the testing procedures employed, and the interpretation of the results.

3.3.1 Automated Accessibility Testing

Automated accessibility testing serves as a crucial first step in identifying common accessibility issues and ensuring compliance with the Web Content Accessibility Guidelines (WCAG) 2.1. The following online tools were employed to conduct a thorough evaluation:

- **WAVE (Web Accessibility Evaluation Tool)**[2]: WAVE analyzes web pages for a range of accessibility issues, including missing alternative text, improper semantic structures, and other markup-related problems. It provides visual feedback by overlaying annotations directly onto the web page, which assists developers in pinpointing and addressing specific issues.
- **Lighthouse (Google Chrome DevTools)**[3]: Lighthouse is an open-source, automated tool that audits web pages for performance, accessibility, best practices, and SEO. The accessibility audit within Lighthouse evaluates various aspects such as colour contrast, usage of ARIA attributes, and the presence of alternative text, providing a composite score that reflects the overall accessibility of the platform.

Each tool was run on multiple pages across the platform to ensure that accessibility standards were uniformly met. The automated tests were executed repeatedly after major updates to verify that the new code did not introduce regressions or new issues. The quantitative data from these tools provided an objective baseline for further qualitative assessments.

3.3.2 Manual Online Testing

While automated tools are invaluable for identifying many common issues, they cannot capture all usability aspects. Therefore, manual online testing was also conducted using simulated user interactions. Although direct user testing with a diverse group of participants was not feasible, the following manual evaluation techniques were implemented:

[2] https://wave.webaim.org/
[3] https://developers.google.com/web/tools/lighthouse/

- **Screen Reader Emulation**: Virtual screen reader software like NVDA (NonVisual Desktop Access) [4] was used to navigate the platform. This free, open-source screen reader for Windows helped identify issues like reading order inconsistencies, missing labels, and content grouping problems that automated tools might overlook. NVDA provides a realistic simulation of how screen reader users interact with the platform, revealing usability gaps in navigation and content structure.
- **Keyboard-Only Navigation Testing**: To ensure that the platform is fully navigable without a mouse, tests were conducted using keyboard-only input. This involved verifying that all interactive elements (links, buttons, form fields) were accessible via the `Tab` key and that the focus order was logical and intuitive.
- **Cross-Browser Testing**: The platform was evaluated on different browsers to ensure compatibility and consistent performance across various user environments. Tests were primarily conducted on the latest versions of Google Chrome [5], Mozilla Firefox [6], and Safari [7] to cover the majority of the user base. While Chromium-based browsers like Chrome provide robust support for accessibility features and developer tools, Safari and Firefox were also included to identify and address any browser-specific discrepancies and limitations. By testing on these three major browsers, the platform aims to offer a consistent and accessible experience for all users, regardless of their preferred browser.

These manual tests provided qualitative insights, complementing the quantitative data from automated tools. They helped identify subtle usability issues and informed iterative improvements to enhance overall accessibility.

3.3.3 Analysis and Interpretation of Results

The data collected from both automated and manual testing was analyzed to provide a comprehensive view of the platform's accessibility:

- **Quantitative Metrics**: Accessibility scores from tools like Lighthouse and WAVE were used as key performance indicators. These scores helped in tracking improvements over successive iterations and in setting benchmarks for compliance.
- **Qualitative Feedback**: Observations from manual testing, including screen reader navigation and keyboard accessibility, were documented in detail. This qualitative feedback was essential for understanding the real-world usability of the accessibility features.
- **Issue Prioritization**: Identified issues were classified based on their severity and impact on the user experience. Critical issues affecting navigation and content comprehension were prioritized for immediate remediation, while minor issues were logged for future updates.

[4] https://www.nvaccess.org/
[5] https://www.google.com/chrome/
[6] https://www.mozilla.org/en-US/firefox/new/
[7] https://www.apple.com/safari/

Overall, the evaluation confirmed that the platform met the key WCAG 2.1 requirements, though some areas such as dynamic content focus management and browser-specific behavior-were noted as opportunities for further refinement.

3.3.4 Limitations of the Evaluation Approach

Despite the comprehensive nature of the evaluation, several limitations must be acknowledged. First, the evaluation did not include extensive direct feedback from users with disabilities. This limitation means that certain context-specific usability challenges, which might only become apparent through real-world interactions with assistive technologies, may remain undetected. Second, the approach relied heavily on automated tools like WAVE and Axe DevTools, which while efficient and effective for identifying common accessibility issues, cannot fully capture the nuances of human interaction. Some complex accessibility issues may only be identified through in-depth, real-world user studies. Finally, the evaluation focused predominantly on Chromium-based browsers like Google Chrome and Microsoft Edge, with limited testing on other browsers such as Mozilla Firefox and Safari. This browser support variability could influence the generalizability of the findings, as accessibility features may behave differently across browsers.

3.3.5 Future Work in Evaluation

Future evaluations should aim to address these limitations through several enhancements. First, expanded user testing with a diverse group of participants, including individuals with disabilities, would provide valuable insights and help identify issues not captured by automated tools. This would require additional funding or institutional support to recruit participants effectively. Second, integrating advanced user interaction analytics could help monitor how users engage with accessibility features over time, offering a data-driven approach to ongoing improvements. Third, incorporating automated accessibility tests into a continuous integration (CI) pipeline would enable regression testing, ensuring that updates do not compromise accessibility standards. Finally, broader cross-browser evaluations should be conducted to include a wider array of browsers and devices, thereby enhancing the robustness and inclusivity of the accessibility features.
In summary, the evaluation methods employed in this study—though primarily online and automated—offer a thorough assessment of the platform's accessibility. The combination of automated tools and manual testing has provided a solid foundation for iterative improvements. While the current approach is limited by the lack of direct user feedback from the disabled community, the insights gained will guide future efforts to create a more inclusive and user-friendly educational platform.

3.4 Conclusion

The methodology outlined in this chapter provides a comprehensive framework for both implementing and rigorously evaluating accessibility features within an educational gamebook platform. Through a detailed, multi-phased approach-encompassing initial research, development, testing, and iterative refinement - this work has established a solid foundation for creating an inclusive digital learning environment.

By leveraging best practices and standards such as WCAG 2.1, the implementation phase introduced innovative solutions including customized theming, speech-to-text, and text-to-speech functionalities, alongside general accessibility enhancements such as semantic HTML structuring, ARIA labelling, and keyboard navigation improvements. These strategies not only ensure that users with diverse abilities can interact with the platform effectively, but also contribute to an overall more intuitive and engaging user experience.

The evaluation methods adopted ranging from automated tools like WAVE and Lighthouse to simulated manual testing techniques - provided both quantitative and qualitative insights into the platform's accessibility performance. Although the evaluation was primarily conducted using online tools due to budget constraints and the challenge of recruiting a diverse user base, the results have been instrumental in identifying areas of strength and opportunities for further improvement.

Moreover, the analysis of test outcomes has allowed for the prioritization of critical issues, thereby establishing a feedback loop that enhances both usability and compliance with accessibility guidelines. The lessons learned from this rigorous evaluation process not only underscore the importance of continual testing and refinement but also pave the way for future enhancements - such as expanded user testing and advanced analytics -that could further enrich the learning experience.

In summary, the methodology presented in this chapter demonstrates a robust and systematic approach to integrating accessibility into digital learning tools. It combines innovative implementation techniques with thorough evaluation practices to ensure that the educational gamebook platform is not only compliant with established accessibility standards but is also tailored to meet the diverse needs of its users. The insights gained from this research serve as a blueprint for future work in the field, emphasizing the necessity of creating truly inclusive educational environments in an increasingly digital world.

Chapter 4 Implementation

This chapter details the technical implementation of accessibility and usability enhancements within the DiGaBo educational gamebook platform. Given the pre-existing project structure, our work focused on augmenting the platform with features that promote inclusivity and a user-friendly experience. The enhancements span several core areas: multilanguage support, dynamic theming, speech-to-text functionality, text-to-speech functionality, and a series of general accessibility improvements. Each section discusses the motivation behind the feature and provides an in-depth look at its technical implementation.

4.1 Development Environment and Technology Stack

The DiGaBo educational gamebook platform leverages a modern web technology stack, chosen for its robustness, maintainability, and suitability for creating interactive web applications. This stack was already in place when this project began, providing a solid foundation upon which to build accessibility enhancements:

- Frontend: Angular 14
- Backend: Node.js with Express.js
- Database: MongoDB
- Version Control: Git with GitLab
- Containerization: Docker

Docker was strategically used to containerize the application components, standardizing the deployment process and mitigating environment-specific issues. This approach ensures that the application behaves consistently across different machines and simplifies the deployment pipeline. Listing 4.1 shows the Docker Compose configuration used for the project.

```
version: "3.7"
services:
  editor:
    build:
      context: ./digabo-editor
      dockerfile: editor.Dockerfile
    ports:
      - "4201:4201"
  ui:
    build:
      context: ./digabo-ui
      dockerfile: ui.Dockerfile
```

```
    ports:
      - "4200:4200"
    volumes:
      - "./digabo-editor:/app/ui"
  backend:
    build:
      context: ./digabo-backend
      dockerfile: backend.Dockerfile
    ports:
      - "3000:3000"
    volumes:
      - "./digabo-backend/images:/app/backend/images/"
    links:
      - database
  database:
    image: mongo
    ports:
      - "27017:27017"
    volumes:
      - "./mongo:/data/db"
```

Listing 4.1: Docker Compose configuration for DiGaBo

This Docker Compose configuration defines several services to encapsulate the application components:

- editor: This service handles the visual editor part of the gamebook, which allows developers and content creators to modify the gamebook. It's built using the 'editor.Dockerfile' located in the './digabo-editor' directory, ensuring a controlled and reproducible build process for the editor.
- ui: This service represents the frontend of the gamebook, which will be presented to the users. It is built using 'ui.Dockerfile' from the './digabo-ui' directory. To enable live updates without needing a full image rebuild, it also mounts a volume, improving the development workflow.
- backend: This is the service for the backend, responsible for handling API requests, managing user data, and serving dynamic content. This is built using 'backend.Dockerfile' from the './digabo-backend' and also mounts a volume to persist uploaded images and files.
- database: This is the simplest service, responsible for the MongoDB database. It mounts a volume to persist the data, ensuring that database contents are preserved across container restarts.

Dockerization Benefits

This containerized setup streamlines deployment by packaging each component with all its dependencies, ensuring consistent performance across different environments and simplifying scaling efforts. By isolating each component in its own container, we can ensure that changes to one part of the application do not inadvertently affect other parts. This modularity also simplifies maintenance and updates, as individual services can be updated and redeployed without impacting the entire system.

4.2 Multilanguage Support

Ensuring that an educational tool can cater to users from diverse linguistic backgrounds is crucial for accessibility and inclusivity. To achieve this, the DiGaBo platform was extended with multilanguage functionality, allowing seamless switching between English and German. The key goals of this implementation were:

- Extracting all hardcoded text strings from the codebase to facilitate easy translation.
- Enabling on-the-fly language switching without requiring a page reload.
- Providing a scalable approach that allows adding new languages with minimal effort.

To meet these goals, we selected `ngx-translate` as the translation library due to its runtime translation capabilities, strong community support, and ease of integration with Angular (as justified in Section 3.2.1).

4.2.1 Implementation Details

The process of implementing multilanguage support consisted of three primary steps: installing dependencies, extracting and externalizing text, and configuring runtime language switching.

Installing Dependencies

To enable translation functionality, the `ngx-translate` package and its HTTP loader were installed using the following command (Listing 4.2).

```
1 npm install @ngx-translate/core @ngx-translate/http-loader
```

Listing 4.2: Installing the Translation Libraries

The `@ngx-translate/core` module provides the translation service, while the `@ngx-translate/http-loader` module allows translations to be loaded from external JSON files.

Extracting Hardcoded Strings

One of the most labour-intensive steps was manually identifying and extracting all hardcoded UI strings from the application code. This was necessary to ensure that all textual content could be dynamically translated based on the user's language preference.
The extracted strings were moved to structured JSON files within the `assets/i18n/` directory (Listing 4.3). This approach enables easy expansion of the language set without modifying the source code.

```
1 {
2   "LANGUAGE_SELECT": "Sprache auswaehlen",
3   "LOGIN": "Einloggen",
4   "REGISTER": "Registrieren",
5   "LOGOUT": "Abmelden",
```

```
  "PLACEHOLDER_EMAIL": "E-Mail",
  "ERROR_EMAIL": "Bitte eine E-Mail eingeben."
}
```

Listing 4.3: Excerpt from de.json translation file

Configuring the Translation Module

To integrate translation capabilities into the application, the translation module was configured in app.module.ts (Listing 4.4).

```
import { HttpClient } from '@angular/common/http';
import { TranslateLoader, TranslateModule } from '@ngx-translate/core';
import { TranslateHttpLoader } from '@ngx-translate/http-loader';

export function HttpLoaderFactory(http: HttpClient) {
    return new TranslateHttpLoader(http, './assets/i18n/', '.json');
}

@NgModule({
    imports: [
        TranslateModule.forRoot({
            loader: {
                provide: TranslateLoader,
                useFactory: HttpLoaderFactory,
                deps: [HttpClient]
            }
        })
    ]
})
export class AppModule { }
```

Listing 4.4: Translation Module Configuration

This configuration defines a factory function (HttpLoaderFactory) that loads translations dynamically from JSON files, ensuring that additional languages can be incorporated with minimal modifications.

Enabling Runtime Language Switching

A critical requirement was the ability to switch languages without requiring a page reload. This was achieved by dynamically updating the language setting within the TranslateService (Listing 4.5).

```
constructor(private translate: TranslateService) {
    const savedLang = localStorage.getItem('language');
    const browserLang = translate.getBrowserLang();
    const defaultLang = savedLang || (browserLang.match(/en|de/) ? browserLang
        : 'en');
    this.translate.setDefaultLang(defaultLang);
    this.translate.use(defaultLang);
    this.selectedLanguage = defaultLang;
}
```

```
9
10 changeLanguage(lang: string) {
11     this.selectedLanguage = lang;
12     this.translate.use(lang);
13     localStorage.setItem('language', lang);
14 }
```

Listing 4.5: Language Switching Logic

This logic achieves:

- Automatic Language Detection: The application initializes using either the saved user preference or the browser's default language.
- Persistent Language Preference: User selections are stored in local storage to maintain consistency across sessions.
- Immediate Language Switching: Updates occur dynamically without requiring a page refresh.

User Interface for Language Selection

To provide a user-friendly mechanism for selecting languages, a dropdown menu was implemented using Angular Material (Listing 4.6).

```
1 <mat-select placeholder="{{ 'LANGUAGE_SELECT' | translate }}" [(value)]="
      selectedLanguage"
2             (selectionChange)="changeLanguage($event.value)">
3     <mat-option *ngFor="let lang of languages" [value]="lang.code">
4         {{ lang.label }}
5     </mat-option>
6 </mat-select>
```

Listing 4.6: Language Selection in the UI

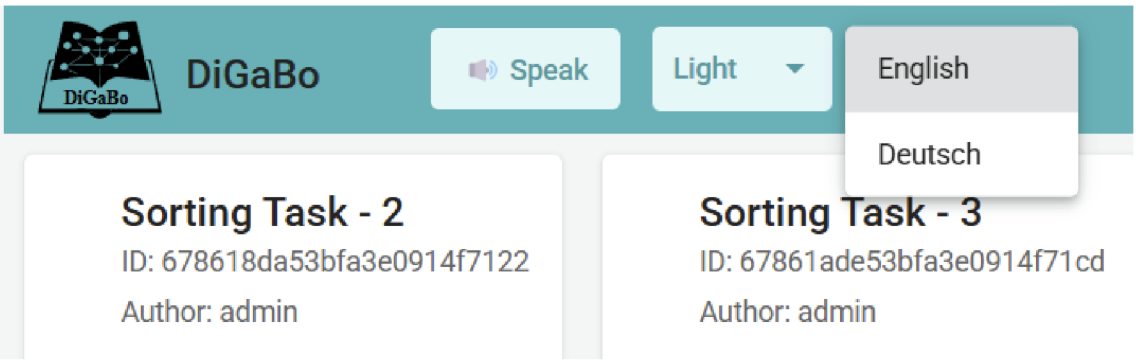

Figure 4.1: Dropdown for Selecting Language

This dropdown allows users to switch languages easily, with translations applied instantly.

4.2.2 Results and Observations

After implementing the multilanguage functionality, the effectiveness of the approach was evaluated based on the following criteria:

1. Ease of Adding New Languages: New JSON files can be created and referenced in the UI without requiring code changes.
2. Performance: Since translations are loaded only when required, there is minimal impact on application speed.
3. User Experience: Users can seamlessly switch between languages, and their preference is maintained across sessions.

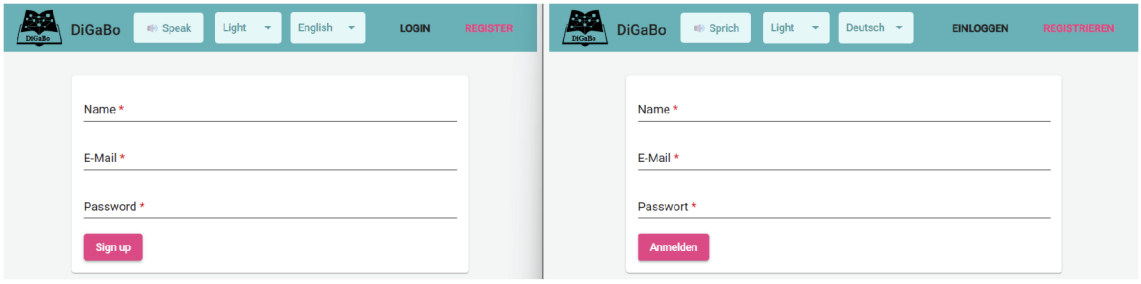

Figure 4.2: Comparison of the Same Page in English and German

Overall, the implemented multi-language support significantly enhances accessibility and user experience, allowing a diverse audience to interact with the platform in their preferred language.

4.2.3 Conclusion

The implementation of multilanguage support in DiGaBo successfully addressed the accessibility challenges associated with linguistic diversity. By selecting `ngx-translate`, extracting hardcoded text, and enabling runtime switching, we created a scalable and user-friendly localization solution. The approach ensures that additional languages can be incorporated efficiently, making the platform more inclusive for a global audience.

4.3 Theming Support

A flexible theming system was developed to enhance visual accessibility and accommodate diverse user preferences. Five themes were implemented: Light, Dark, High Contrast, Sepia, and Low Animation. These themes provide users with the ability to personalize the interface for optimal readability, eye comfort, and accessibility.

4.3.1 Implementation Details

The theming system was implemented as a modular solution, ensuring ease of maintenance and scalability. The primary steps in the implementation included:

1. Extracting hardcoded colours from UI components.
2. Defining a centralized theme structure using CSS variables.
3. Implementing a `ThemeService` to manage theme selection and persistence.
4. Ensuring WCAG compliance for colour contrast and animation reduction.

Extracting Hardcoded Colors

Initially, colours were hardcoded into individual components, making theme modifications cumbersome. To facilitate theming, all colours were extracted into SCSS variables and grouped into separate theme files. This approach improved maintainability and allowed for dynamic theme switching.

Centralized Theme Definition

Each theme was defined using CSS variables within separate SCSS files. A sample of the light theme colour definitions is provided in Listing 4.7.

```
body.light-theme {
  --background-color: #f5f5f5;
  --text-color: #212121;
  --text-color-light: #636363;
  --link-color: #1e88e5;
  --header-bg-color: #00B1B7;
  --hover-bg-color: #e8e8e8;
  --hover-bg-color-light: #abe7e8;
  --background-color-quiz-options: #cccccc;
  --card-bg-color: #ffffff;
  --add-card-bg-color: #dfdfdf;
  --required-color: red;
  --editor-color: #212121;
  --drag-editor-color: #f5f5f5;
  --border-color: #636363;

  background-color: var(--background-color);
  color: var(--text-color);
}
```

Listing 4.7: An excerpt of light-theme color definitions

Each theme follows a similar structure, with variables controlling text colours, backgrounds, and interactive elements. By defining colours in variables, any changes to the theme require modifications in a single file, simplifying maintenance.
In addition, a colour palette visualization was created to illustrate the differences between four themes as shown in Figure 4.3.

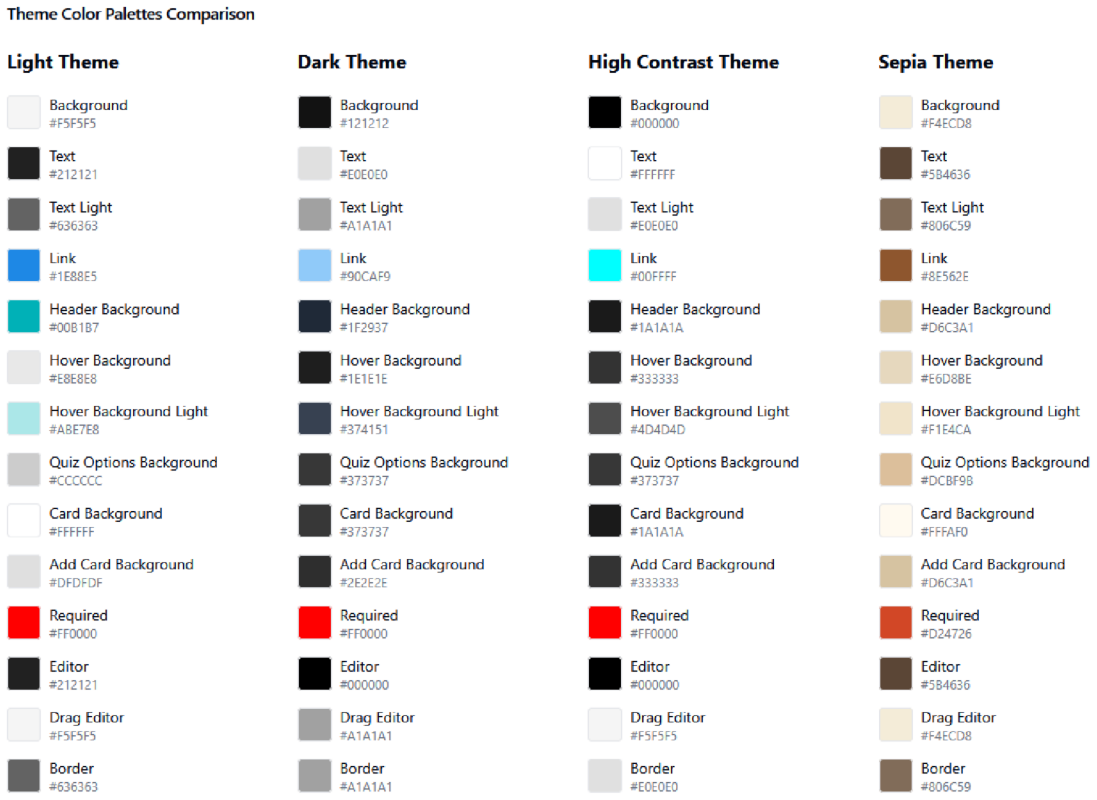

Figure 4.3: Color Palette for Light, Dark, High Contrast, and Sepia Themes

Theme Management Service

To facilitate theme selection and persistence, a dedicated `ThemeService` was implemented. The service manages theme changes by dynamically modifying the `body` class and storing the user's preference in local storage.

```
import { Injectable } from '@angular/core';

@Injectable({
  providedIn: 'root',
})
export class ThemeService {
  private activeTheme: string;
  constructor() {
    this.activeTheme = localStorage.getItem('theme') || 'light-theme';
    this.applyTheme(this.activeTheme);
  }
  applyTheme(theme: string): void {
    document.body.classList.remove(this.activeTheme);
    this.activeTheme = theme;
    document.body.classList.add(theme);
    localStorage.setItem('theme', theme);
  }
}
```

Listing 4.8: Theme Service Implementation

This service ensures that the selected theme is applied on page load and remains consistent across sessions.

User Interface for Theme Selection

A dropdown menu was provided to allow users to select a preferred theme. The interface utilizes Angular Material components for a consistent user experience.

```
<div class="theme-container">
  <mat-select placeholder="Select Theme" [(value)]="selectedTheme"
              (selectionChange)="changeTheme($event.value)">
    <mat-option *ngFor="let theme of themes" [value]="theme.code">
      {{ theme.label }}
    </mat-option>
  </mat-select>
</div>
```

Listing 4.9: Theme Selection UI

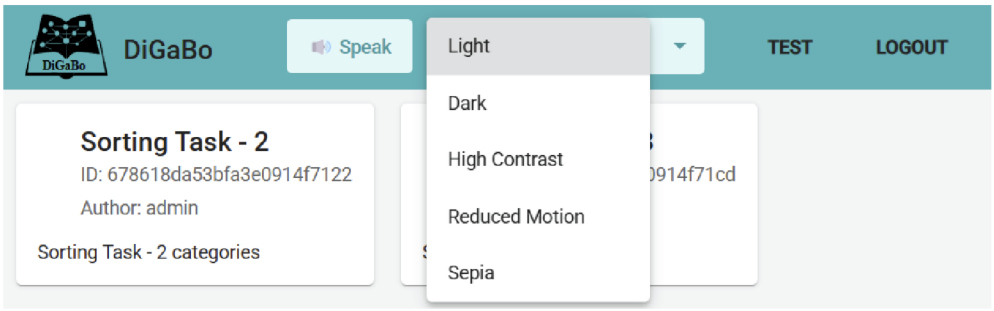

Figure 4.4: Theme Selection Dropdown

Ensuring Accessibility Compliance

Each theme was designed with accessibility considerations, particularly for contrast and readability. Figure 4.5 demonstrates a comparison between the Light and Dark themes.

Key accessibility enhancements included:

- Contrast Compliance: The High Contrast theme meets WCAG 2.1 contrast guidelines (minimum 4.5:1 for text).
- Reduced Animation: The Low Animation theme minimizes motion-based effects to aid users with vestibular disorders.
- Color Differentiation: The Sepia theme provides a warmer colour scheme to reduce eye strain.

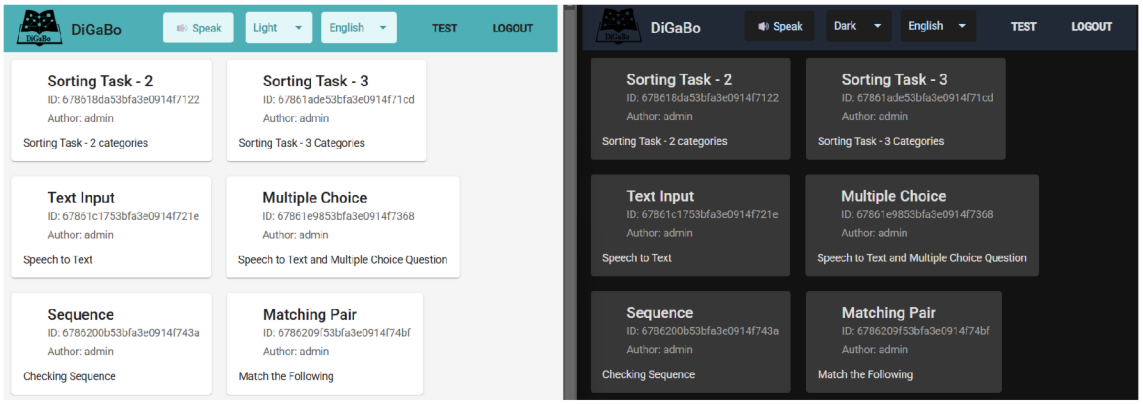

Figure 4.5: Comparison of Light Theme and Dark Theme

4.3.2 Conclusion

The custom theming solution provided a balance between flexibility, performance, and accessibility. By leveraging CSS variables and a centralized theme management approach, the system ensures consistency and ease of modification. The adherence to WCAG guidelines further enhances the inclusivity of the platform, allowing users to customize their experience according to their needs.

4.4 Speech-to-Text Functionality

To enhance accessibility for users with motor impairments or those who prefer voice input, a speech-to-text (STT) feature was implemented. This functionality allows users to dictate text directly into input fields, eliminating the need for manual typing.

4.4.1 Implementation Details

The STT functionality was built using the browser-native Web Speech API, which enables real-time speech recognition. A key design consideration was choosing a cost-effective approach. While third-party services such as Google Cloud Speech-to-Text or Microsoft Azure Speech offer superior accuracy and multi-browser support, they require API keys and impose usage costs. Given the project's constraints, the Web Speech API was selected despite its limited cross-browser compatibility.

Browser Limitations and Trade-offs

One of the primary challenges in implementing STT functionality was browser support inconsistency:

- **Supported Browsers**: The Web Speech API is fully supported in Chromium-based browsers (Google Chrome, Microsoft Edge), enabling seamless speech recognition.
- **Limited Support**: Safari offers partial support, primarily for speech synthesis rather than recognition.

- **Unsupported Browsers**: Mozilla Firefox and some other non-Chromium browsers do not support the Web Speech API, rendering the STT feature unavailable in these environments.

Given these limitations, users attempting to use STT in an unsupported browser receive a clear message instructing them to switch to a supported browser.

Speech Recognition Initialization

The speech recognition component initializes the Web Speech API and configures it to use the user's preferred language. Listing 4.10 demonstrates this process.

```
initSpeechRecognition(): void {
    const { webkitSpeechRecognition } = window as any;
    if (!('SpeechRecognition' in window) && !webkitSpeechRecognition) {
        alert('Speech recognition is not supported in this browser. Please use
            Chrome or Edge.');
        return;
    }
    this.recognition = new (webkitSpeechRecognition)();
    this.recognition.lang = this.selectedLanguage || 'en-US';
    this.recognition.interimResults = false;
    this.recognition.maxAlternatives = 1;

    this.recognition.onresult = (event: any) => {
        const transcript = event.results[0][0].transcript;
        this.userInput = this.userInput ? `${this.userInput} ${transcript}` :
            transcript;
        console.log('Speech recognized: ', transcript);
        this.cdr.detectChanges();
    };
    this.recognition.onend = () => {
        this.isListening = false;
        this.cdr.detectChanges();
    };
    this.recognition.onerror = (event: any) => {
        this.isListening = false;
        console.error('Speech recognition error: ', event.error);
        alert('An error occurred during speech recognition: ' + event.error);
        this.cdr.detectChanges();
    };
}
```

Listing 4.10: Speech Recognition Initialization

This implementation:

- Checks if the Web Speech API is available and provides an error message if unsupported.
- Configures the API with the user's selected language.
- Handles speech recognition events, updating the text input dynamically.
- Implements error handling for better user experience.

User Interface for Speech Recognition

A simple and accessible UI was designed to activate speech recognition. The button with a microphone icon provides a clear visual indicator when speech recognition is active.

```
<input matInput placeholder="{{data.input_example}}" [(ngModel)]="userInput"
    name="userInput" #myinput>
<button mat-icon-button matSuffix aria-label="Activate microphone" (click)="
    startSpeechRecognition()">
    <mat-icon [class.mic-active]="isListening">mic</mat-icon>
</button>
```

Listing 4.11: Speech-to-Text UI Element

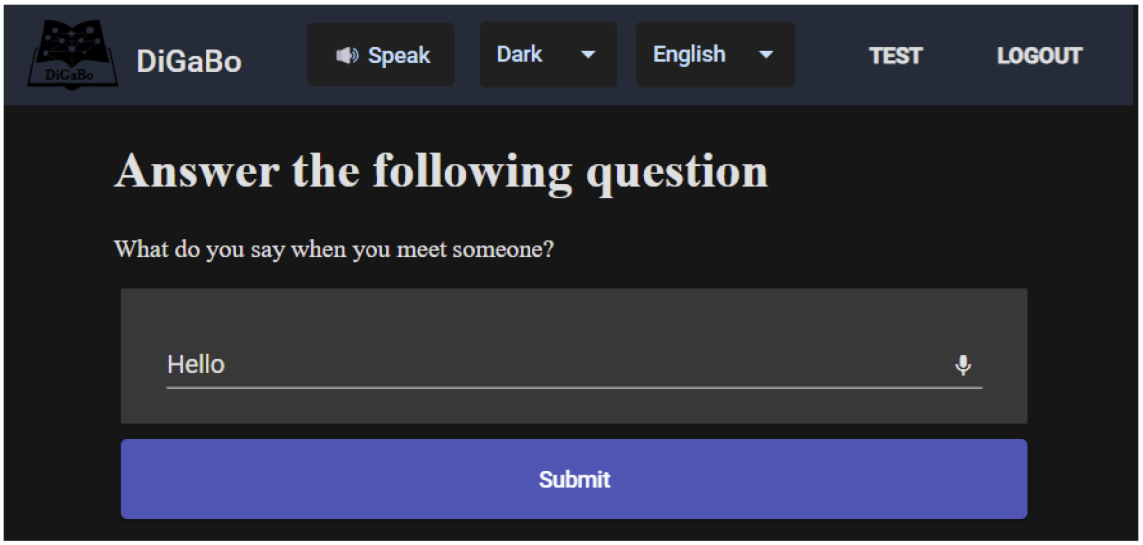

Figure 4.6: Speech-to-Text Input Field with Microphone Button

The UI incorporates several key elements for user interaction. It features a standard input field for text entry, accompanied by a button to activate speech recognition and a microphone icon that provides visual feedback when the listening function is active.

Error Handling and User Guidance

Regarding error handling and user guidance, the system implements comprehensive measures to handle situations where speech recognition capabilities may be limited. When users attempt to use speech recognition in unsupported browsers, the system actively detects the browser type and provides appropriate warning messages. If speech recognition encounters any issues, clear error messages are displayed to inform the user. Additionally, the system offers helpful guidance by recommending that users switch to a Chromium-based browser to access full functionality.

4.4.2 Future Enhancements

While the current implementation effectively provides STT functionality, several areas for improvement have been identified:

- **Improved Cross-Browser Support**: Investigating third-party services like Mozilla DeepSpeech or Vosk could enable STT in non-Chromium browsers.
- **Offline Recognition**: Exploring on-device speech models could allow users to dictate text without requiring an internet connection.
- **Enhanced Language Support**: The current implementation supports English and German, but expanding to additional languages would improve accessibility.
- **Real-time Transcription Feedback**: Implementing interim results display could provide a smoother user experience.

4.4.3 Conclusion

The implementation of speech-to-text functionality using the Web Speech API successfully enhances accessibility by offering an alternative input method for users with motor impairments. Despite browser limitations, this approach remains a cost-effective and efficient solution. Future improvements could expand browser compatibility, enable offline functionality, and refine user experience further.

4.5 Text-to-Speech Functionality

To enhance accessibility and provide an alternative content consumption method, a text-to-speech (TTS) feature was implemented. This functionality benefits users with visual impairments, reading difficulties, cognitive overload, or those who prefer auditory learning.

4.5.1 Implementation Details

The TTS functionality was built using the native Web Speech API's Speech Synthesis feature, which allows dynamic conversion of text content into speech. The primary challenge in implementing this feature was ensuring that only relevant content was read aloud while excluding repetitive or non-essential elements, such as navigation headers and footers, which are already covered by screen readers.

Extracting the Relevant Content for Speech

Unlike traditional screen readers that read an entire page sequentially, this TTS functionality was designed to focus on the dynamic content of the gamebook interface. The system targets the `<main>` section of the webpage, ensuring that only game-related text is spoken.
Listing 4.12 demonstrates the implementation, which dynamically extracts and reads the main content.

```
1 speakPageContent() {
2     const synth = window.speechSynthesis;
3     const mainElement = document.querySelector('main');
4 
5     if (!mainElement) {
6         console.log('Main section not found.');
7         return;
8     }
```

```
9
10      const mainText = mainElement.innerText;
11
12      if (!mainText.trim()) {
13          console.log('No content to read in the main section.');
14          return;
15      }
16
17      const utterance = new SpeechSynthesisUtterance(mainText);
18      utterance.lang = this.selectedLanguage || 'en-GB';
19      utterance.rate = 1;
20      utterance.onerror = (event) => {
21          console.error('SpeechSynthesis error:', event);
22      };
23
24      synth.speak(utterance);
25  }
```

Listing 4.12: Text-to-Speech Function

This function:

- Extracts only the text within the `<main>` section, avoiding repetitive speech of headers, footers, or navigation elements.
- Ensures content is non-empty before initiating speech synthesis.
- Sets the speech-language dynamically based on user preference.
- Handles errors gracefully to prevent unexpected interruptions.

User Interface for Speech Activation

A simple button was added to trigger text-to-speech functionality. Listing 4.13 shows its implementation.

```
1 <div class="tts-container">
2     <button mat-button (click)="speakPageContent()">{{ 'SPEAK' | translate }}</
          button>
3 </div>
```

Listing 4.13: Text-to-Speech Button

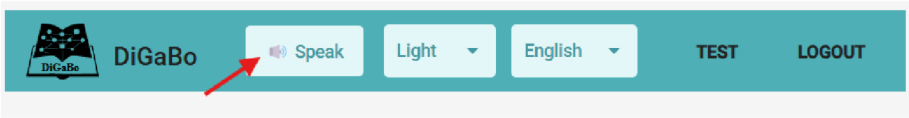

Figure 4.7: TTS Activation Button

The button label is translated dynamically to support multiple languages, ensuring accessibility for a diverse user base.

Customization and Future Enhancements

Due to time constraints, certain features were not implemented but are planned for future iterations:

- **Highlighting Spoken Words**: Synchronizing text highlighting with speech to provide visual feedback to users.
- **Speech Customization**: Allowing users to control pitch, speed, and volume for a personalized experience.
- **Multi-Voice Selection**: Enabling users to choose between different voice options for better engagement.
- **Language and Accent Selection**: Allowing users to fine-tune accents for better comprehension.

4.5.2 Conclusion

The implementation of text-to-speech functionality significantly enhances the accessibility of the platform by providing an auditory mode of content consumption. By leveraging the native Web Speech API, this approach ensures a lightweight and efficient solution while avoiding external dependencies. Future enhancements, such as real-time word highlighting and voice customization, will further refine the user experience, making the application even more inclusive.

4.6 General Accessibility Improvements

Beyond the core accessibility features discussed earlier, several structural and usability improvements were implemented to create a more inclusive user experience. These enhancements were aimed at improving navigation, readability, and interaction, particularly for users relying on assistive technologies such as screen readers and keyboard navigation.

4.6.1 Semantic HTML and ARIA Enhancements

A strong emphasis was placed on improving the semantic structure of the application. Assistive technologies such as screen readers rely heavily on semantic HTML elements to interpret and navigate content efficiently. To enhance compatibility and ensure proper document structure, several key improvements were implemented. The pages were structured with a consistent heading hierarchy (h1, h2, h3), ensuring that content flows meaningfully for screen reader users. Additionally, ARIA attributes were added to buttons, links, and form elements, providing clearer descriptions for screen reader users. Decorative elements, such as icons and purely aesthetic images, were marked with aria-hidden="true" to prevent unnecessary screen reader announcements.

These improvements facilitate a smoother browsing experience for visually impaired users, ensuring that the application provides a clear, structured, and meaningful interaction model.

4.6.2 Keyboard Navigation Enhancements

Ensuring full keyboard accessibility was a major focus, as users with motor impairments or those preferring keyboard navigation need to be able to interact with all application elements without requiring a mouse. The following adjustments were made to enhance the experience. A logical tab order was implemented by strategically applying the tabindex attribute to maintain a natural focus order, ensuring intuitive navigation. Interactive elements, such as buttons and modal dialogs, were configured to respond to both Enter and Space keypresses. Focus management was improved by standardizing styles for focus rings to provide clear visual indicators when elements are selected via keyboard input. Additionally, dialog boxes and popups were updated to trap focus within the modal until the user explicitly closes it, preventing unintended focus shifts.

These refinements significantly enhance the experience for users navigating without a mouse, ensuring they can seamlessly interact with all essential components.

4.6.3 Text Alternatives for Non-Text Content

To improve accessibility for users relying on screen readers, alternative text descriptions were provided for meaningful images. The `alt` attribute was systematically added to all relevant images to ensure that visually impaired users receive a clear description of their purpose or content. This enables screen reader users to understand the function of images without relying on visual cues.

While additional improvements, such as captions for data visualizations and marking decorative images appropriately, were considered, they were not implemented due to time constraints. Future work could focus on further refining these aspects to enhance accessibility for users engaging with complex graphical content.

4.6.4 Compliance with WCAG 2.1 Guidelines

All accessibility improvements were designed in alignment with WCAG 2.1 standards. Specifically:

- WCAG 1.3.1 (Info and Relationships): Ensuring a logical document structure with semantic HTML.
- WCAG 2.1.1 (Keyboard Accessibility): Guaranteeing full functionality via keyboard input.
- WCAG 1.4.5 (Text Alternatives): Providing meaningful descriptions for non-text content.
- WCAG 4.1.2 (Name, Role, Value): Implementing ARIA attributes to enhance screen reader compatibility.

By adhering to these standards, the platform not only meets technical compliance but also provides a more intuitive and accessible experience for all users.

4.7 Conclusion

The implementation phase of the DiGaBo platform introduced a comprehensive set of features aimed at improving accessibility and inclusivity. By systematically addressing language accessibility, dynamic theming, speech-to-text and text-to-speech functionalities, and broader usability enhancements, the platform now provides an adaptable and user-friendly experience.

Each feature was developed with careful consideration of user needs, technical feasibility, and WCAG 2.1 compliance. The combination of manual and automated testing ensured that these enhancements effectively addressed accessibility barriers.

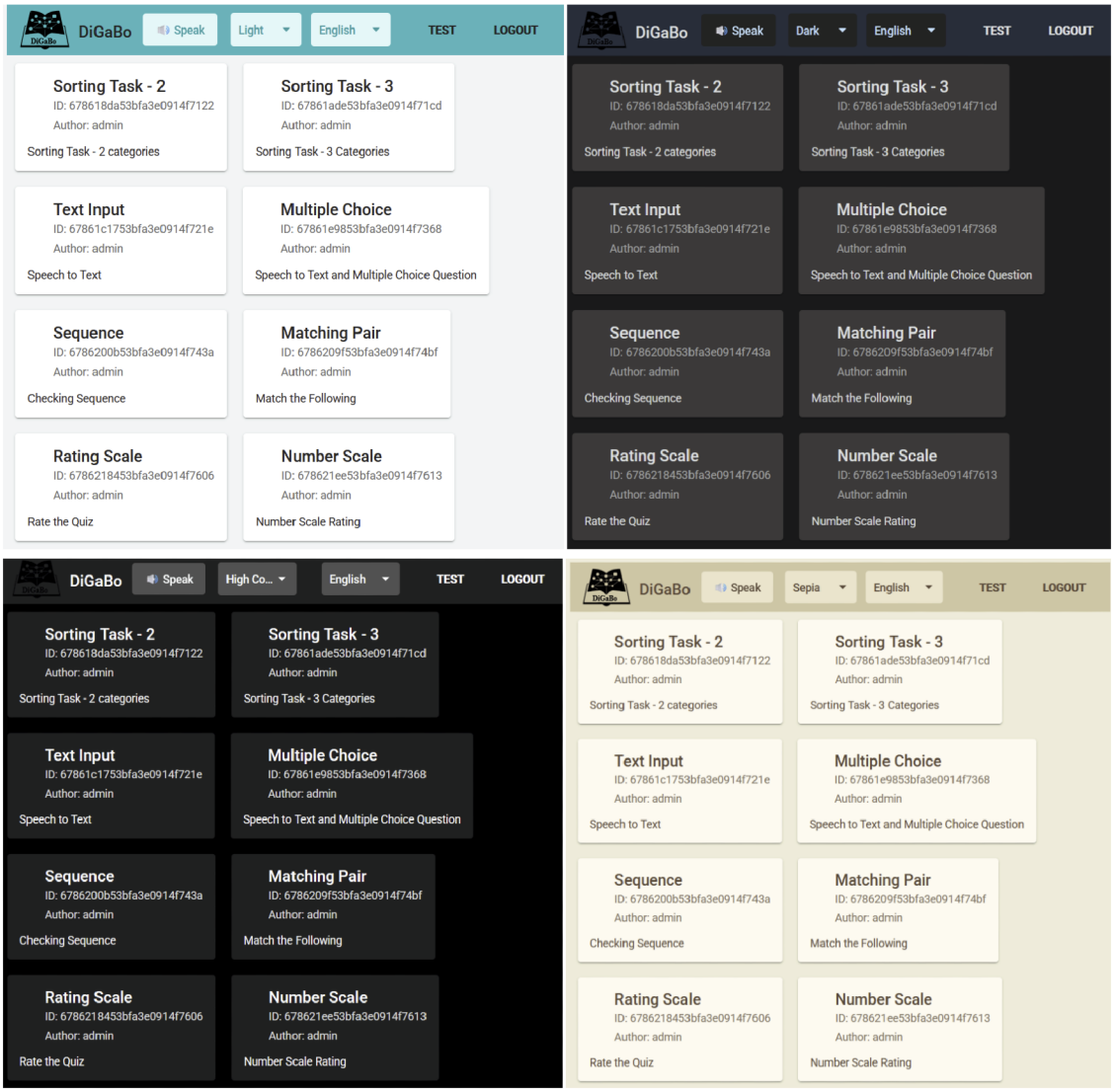

Figure 4.8: Homepage in all different Themes

4.7.1 Key Achievements

The following key improvements were successfully implemented:

- **Multilanguage Support:** Allowing users to interact with the platform in their preferred language while ensuring compliance with accessibility guidelines.

- **Dynamic Theming:** Providing visually accessible themes, including high contrast and reduced animation options.
- **Speech-to-Text Integration:** Offering an alternative text input method for users with motor impairments.
- **Text-to-Speech Implementation:** Enabling users to listen to game content with customizable language settings.
- **General Accessibility Enhancements:** Improving keyboard navigation, semantic structure, and screen reader compatibility.

These changes significantly enhance the platform's usability, making it accessible to a wider audience, including users with disabilities.

4.7.2 Final Thoughts

The implementation of accessibility features in DiGaBo reflects a commitment to inclusivity and usability. By focusing on multilingual accessibility, adaptable theming, alternative input/output methods, and structural enhancements, the platform has become more accessible to users with diverse needs. The iterative nature of development ensures that future updates can refine and expand on these efforts.
In the next chapter, we will present the evaluation results, highlighting how these implementations have improved the overall user experience and discussing potential future enhancements.

Chapter 5 Evaluation and Results

This chapter presents a comprehensive evaluation of the accessibility and usability improvements implemented in the educational gamebook platform. The evaluation was conducted using a combination of automated tools (WAVE and Lighthouse), manual testing, screen reader testing, cross-browser testing, and automated unit tests. The results indicate significant enhancements in accessibility compliance, usability, and overall user experience.

5.1 Automated Evaluation

To ensure a systematic evaluation, the website was analyzed before and after the implementation of accessibility improvements using two industry-standard tools: **WAVE** and **Lighthouse**. The results from these tools provide an objective measure of the enhancements made.

5.1.1 Initial Evaluation

The initial evaluation revealed several accessibility issues that impacted the usability of the platform for users with disabilities. The WAVE accessibility report, shown in Figure 5.1, identified multiple critical issues:

- **Missing alternative text:** One or more images lacked `alt` attributes, making them inaccessible to screen readers.
- **Contrast errors:** 17 instances of low contrast between text and background colours, reducing readability for visually impaired users.
- **Structural Issues:** Improper use of semantic HTML elements, including missing headings and unordered lists.
- **ARIA Issues:** Incorrect or missing ARIA attributes, affecting assistive technologies.
- **Presence of a `<noscript>` element:** Flagged as a potential issue, though not a critical error.

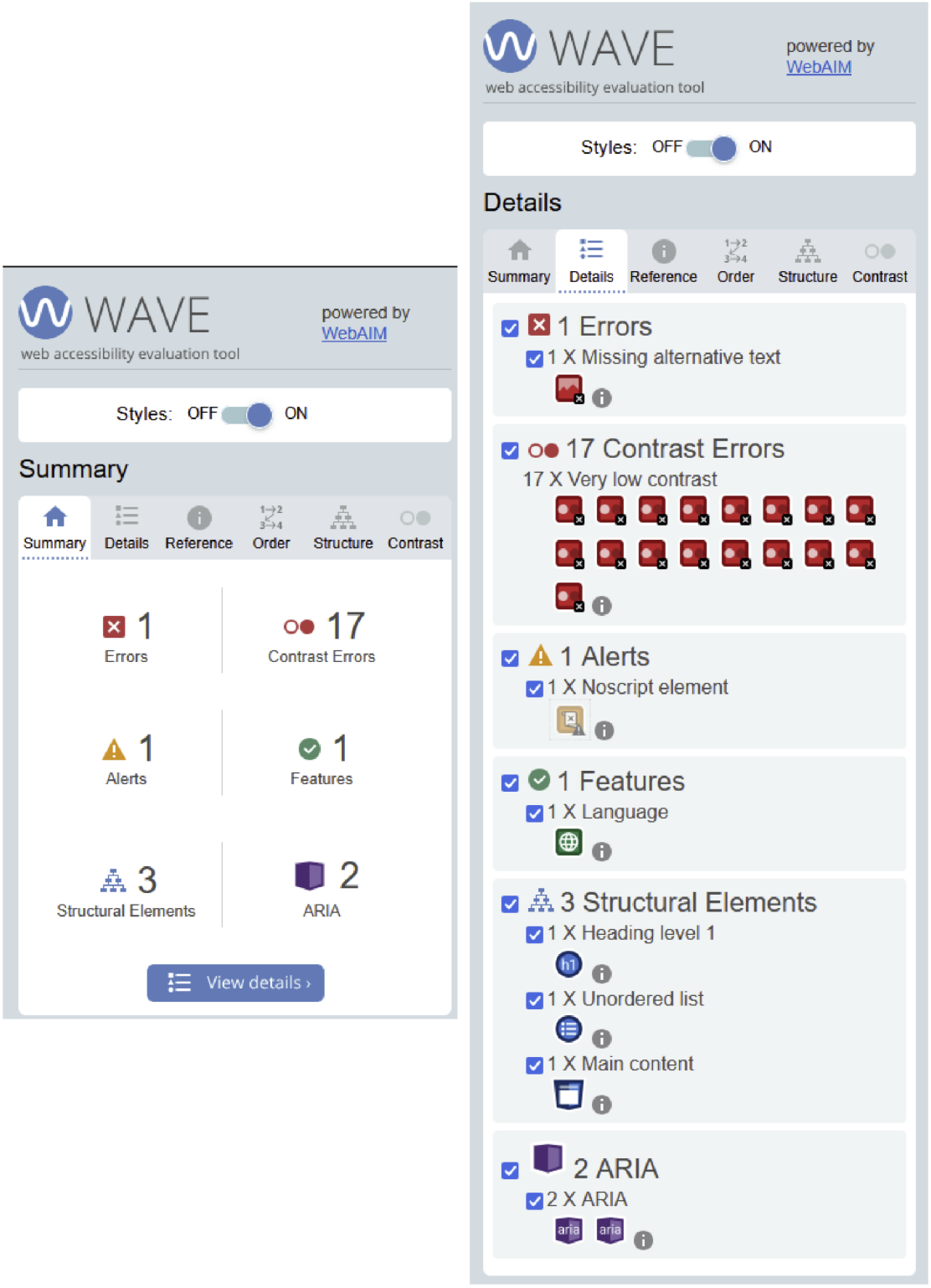

Figure 5.1: Initial WAVE Accessibility Report

Similarly, Lighthouse initially rated the platform's accessibility at **59/100**, as shown in Figure 5.2, highlighting key areas requiring improvement:

- Incorrect or misspelled ARIA attributes.
- Missing `alt` attributes for images.
- Form elements lacking associated labels.
- Links without discernible names, making navigation difficult for screen readers.
- Low contrast ratios affecting readability.
- Absence of a `lang` attribute in the HTML element.

These issues underscored the necessity for comprehensive accessibility enhancements to comply with Web Content Accessibility Guidelines (WCAG) 2.1 standards.

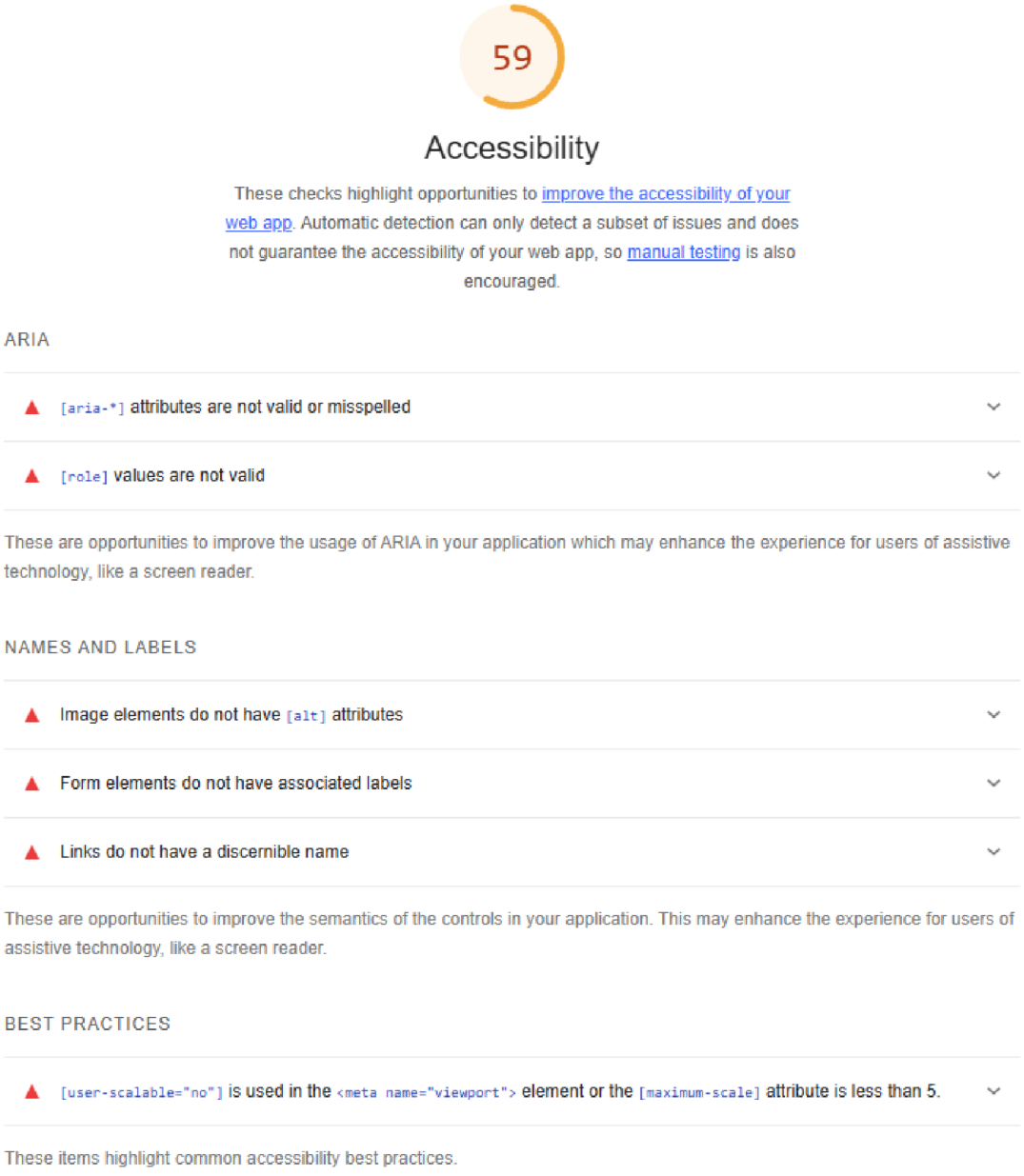

Figure 5.2: Initial Lighthouse Accessibility Report (Score: 59/100)

5.1.2 Post-Implementation Evaluation

Following the implementation of accessibility enhancements, a second round of evaluations was conducted. The WAVE report, as shown in Figure 5.3, confirmed the resolution of all critical errors:

- All missing `alt` attributes were correctly assigned.
- Contrast issues were fixed by adjusting colour schemes and text styling.
- Structural improvements, including proper use of headings and landmarks, were implemented.
- ARIA attributes were corrected and properly applied to interactive elements.

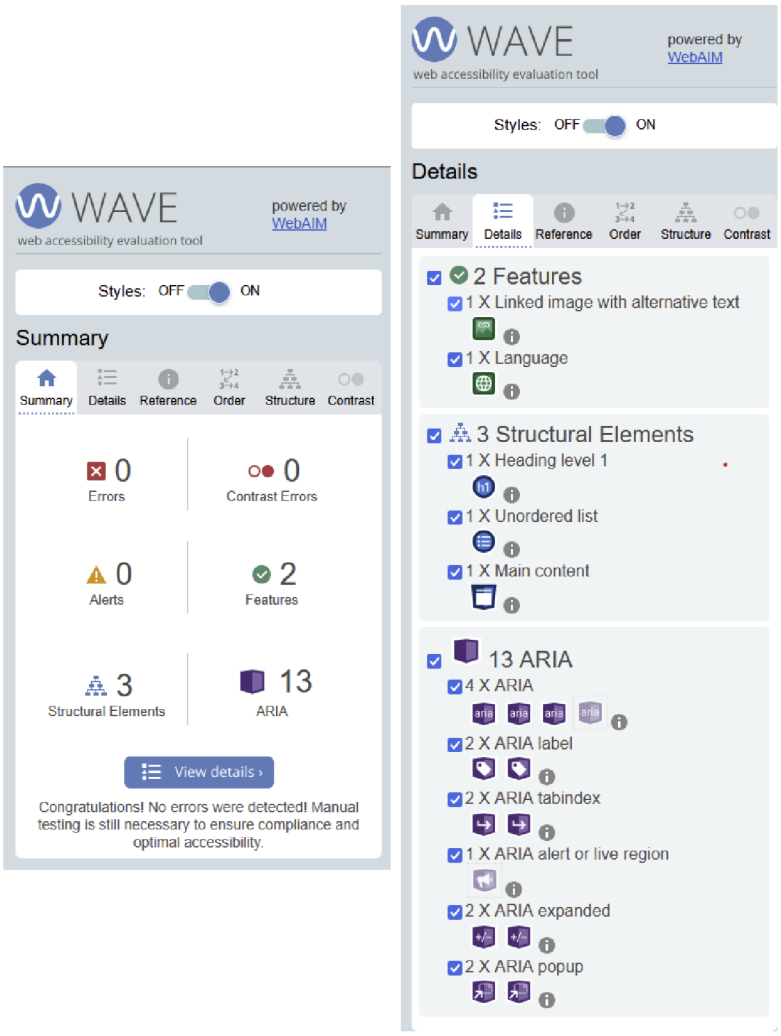

Figure 5.3: Post-Implementation WAVE Accessibility Report

The Lighthouse audit demonstrated a remarkable improvement, with the accessibility score increasing from **59 to 100/100**, as depicted in Figure 5.4. All previously flagged issues were successfully addressed, and additional manual checks were performed to ensure compliance.
The dramatic improvement in accessibility scores demonstrates the effectiveness of the enhancements and their compliance with industry standards.

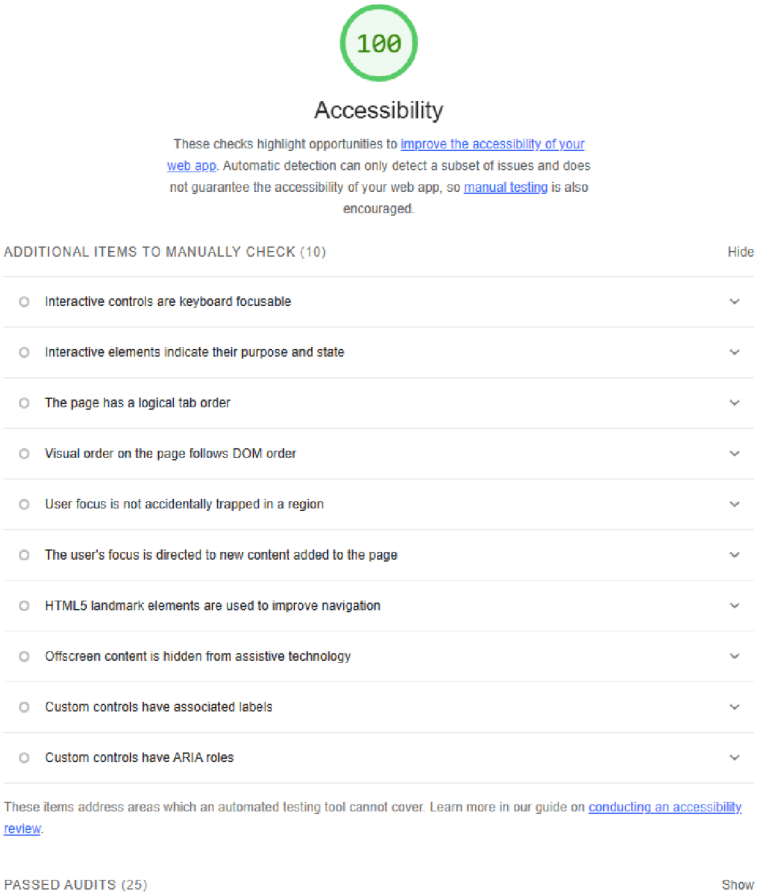

Figure 5.4: Post-Implementation Lighthouse Accessibility Report (Score: 100/100)

Also, to ensure that the accessibility enhancements did not introduce functional regressions, the pre-existing automated unit tests were executed. These tests validated the correctness of various UI components and interactions. All tests passed successfully, indicating that the new features did not introduce regressions.

5.2 Manual Testing

While automated tools provide valuable insights, they may not capture all accessibility barriers. Therefore, a detailed manual testing process was conducted to identify additional usability issues.

5.2.1 Screen Reader Testing

To ensure accessibility for visually impaired users, the platform was tested using the NVDA screen reader. The testing confirmed several key aspects of accessibility. All interactive elements were successfully navigable using NVDA, and proper ARIA roles and labels were read aloud correctly by the screen reader. During navigation between elements, keyboard focus was announced correctly as expected. Throughout the testing process, no unexpected or redundant announcements were encountered.

The successful NVDA test results confirm that the platform is accessible for screen reader users, significantly improving its usability.

5.2.2 Cross-Browser Compatibility

To ensure a consistent experience across different web browsers, the platform was tested on Chromium and Mozilla Firefox. The evaluation revealed important insights regarding browser compatibility. On Chromium, all accessibility features worked as expected, demonstrating full functionality. However, in Mozilla Firefox, an issue was identified where the speech-to-text functionality did not function as intended. This challenge was addressed by implementing a fallback mechanism for browsers that do not fully support speech recognition APIs, wherein a popup alert is displayed to inform users about the limitation and suggest using a compatible browser such as Chrome or Edge. Beyond this single issue, no other major discrepancies were found between browsers. These tests ensured that accessibility improvements were effective across multiple browsers, allowing a seamless experience for users regardless of their browser choice.

5.2.3 Findings from Manual Testing

During manual testing, several accessibility barriers were observed:

- **Inconsistent keyboard navigation:** Some interactive elements were not accessible using the keyboard alone.
- **Improper focus management:** Modal dialogs did not retain keyboard focus, leading to a disorienting experience for users relying on keyboard navigation.
- **Missing descriptive labels:** Certain custom UI components lacked descriptive labels, making them inaccessible to screen readers.

5.2.4 Fixes and Enhancements

To address these issues, the following improvements were implemented:

- Ensured all interactive elements are focusable using the `tab` key.
- Implemented proper focus trapping within modal dialogs.
- Added ARIA labels and roles to custom elements to enhance screen reader compatibility.

These refinements were verified by retesting the platform with screen readers and keyboard navigation, ensuring a seamless experience for users with disabilities.

5.3 Results Summary

The evaluation process demonstrated a significant enhancement in accessibility and usability. The key improvements can be summarized as follows:

- The WAVE tool no longer reports any critical accessibility errors.
- Lighthouse accessibility scores improved from **59/100 to 100/100**, signifying full compliance with WCAG standards.

- Manual testing confirmed seamless keyboard navigation, proper focus management, and improved screen reader support.
- Screen reader testing with NVDA validated full accessibility compliance.
- Cross-browser testing ensured compatibility with major browsers, with mitigations for identified issues.
- All automated unit tests passed successfully, verifying the stability of the implemented changes.

These results highlight the success of the accessibility improvements in making the educational gamebook platform more inclusive and user-friendly for diverse audiences.

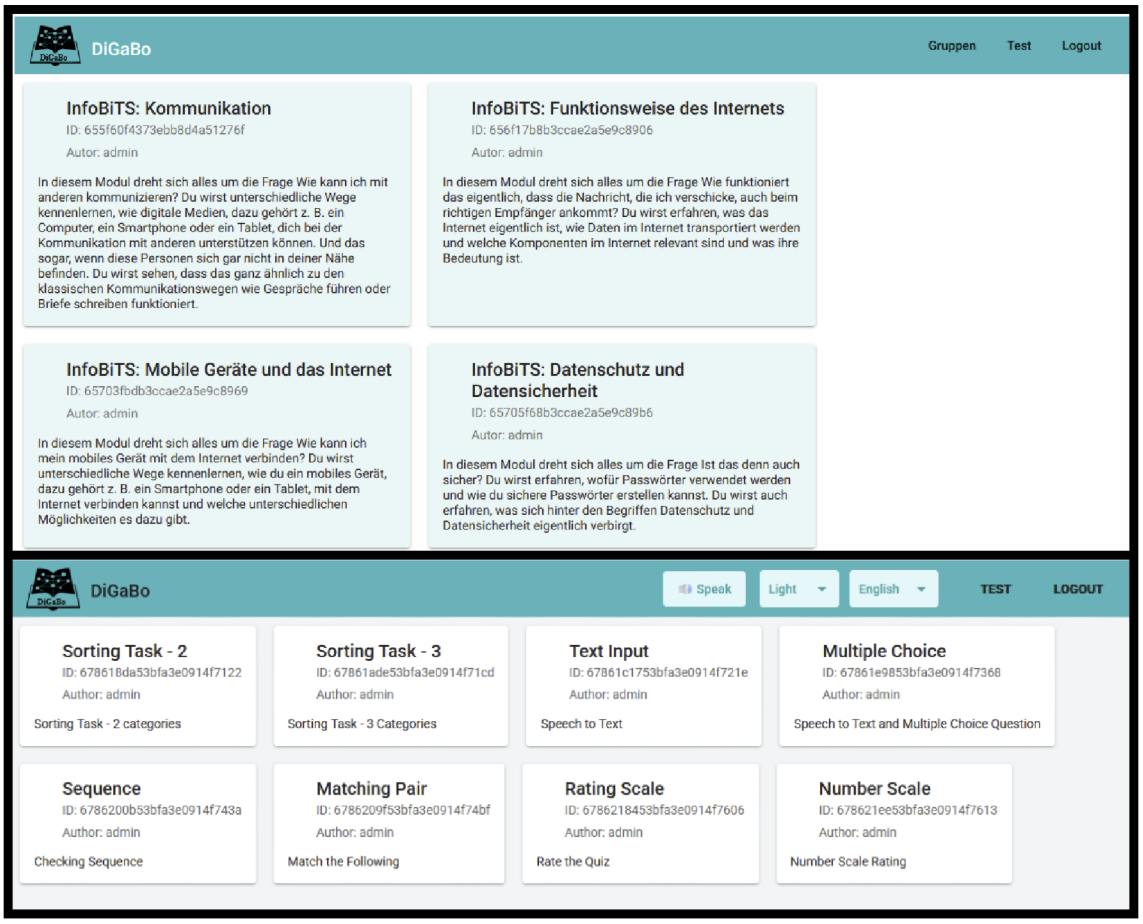

Figure 5.5: Comparison of Old UI (Top) vs. New UI (Bottom) - Light Theme

Chapter 6 Conclusion

This thesis has explored the critical intersection of gamification, accessibility, and usability in educational technologies, with a specific focus on enhancing educational gamebooks. Through a comprehensive literature review, methodological implementation, and rigorous evaluation, we have addressed the pressing need for more inclusive and user-friendly digital learning tools. The work conducted on the DiGaBo platform serves as a practical demonstration of how these enhancements can be realized in an existing educational gamebook system.

6.1 Key Findings and Contributions

Enhanced Accessibility Features

The implementation of multilanguage support, flexible theming (including Light, Dark, High Contrast, Sepia, and Low Animation modes), and robust speech-to-text/text-to-speech functionalities have significantly broadened the platform's accessibility. These enhancements not only exceed WCAG 2.1 guidelines [W3C18] but also cater to diverse needs including visual, auditory, and motor impairments. The multilanguage support particularly breaks down linguistic barriers, enabling global accessibility while maintaining runtime language switching through the ngx-translate library.

User-Centered Usability Improvements

Through iterative design cycles, we implemented semantic HTML, ARIA enhancements, and improved keyboard navigation. This dual approach of automated evaluations (WAVE, Lighthouse) and manual testing with assistive technologies resulted in an interface that is both intuitive for general users and functional for those with physical limitations. The custom theming system and dynamic speech feature adjustments demonstrate how accessibility can be seamlessly integrated without compromising technical performance.

Technical Innovation in Gamified Accessibility

This research makes significant strides in proving that gamification elements can coexist with rigorous accessibility standards. The successful integration of game mechanics with features like high-contrast modes and reduced animation options establishes a new paradigm for educational technologies—showing that engagement and inclusivity can be mutually reinforcing rather than competing priorities.

6.2 Evaluation Results

The evaluation phase of this project yielded promising results. The automated accessibility tests showed a marked improvement, with the Lighthouse accessibility score increasing from 59 to 100. Manual testing further confirmed the effectiveness of the implemented features, particularly in screen reader compatibility and cross-browser

functionality. Post-implementation analysis confirms the framework's effectiveness in creating inclusive digital learning environments. The results particularly highlight how automated testing tools combined with manual assistive technology verification can ensure comprehensive accessibility compliance.

6.3 Implications for Educational Technology

The findings of this thesis have several important implications for the field of educational technology. First and foremost, this work underscores the importance of considering accessibility and usability from the outset of educational technology development. Our research demonstrates that these aspects can be seamlessly integrated into gamified learning platforms, setting a new standard for inclusive design in educational tools. By improving accessibility and usability, we have potentially enhanced the learning experience for a diverse range of users, which could lead to improved engagement, knowledge retention, and overall educational outcomes.

The methodologies and technologies used in this project have proven to be both scalable and adaptable, allowing other educational platforms to adopt similar approaches to enhance their accessibility and usability features. Furthermore, by making educational gamebooks more accessible, we contribute to bridging the digital divide in education, ensuring that learners with diverse needs and backgrounds can benefit from these innovative learning tools.

6.4 Limitations

While our research yielded valuable insights, it is important to acknowledge certain limitations. The evaluation phase was primarily conducted through automated tools and limited manual testing, suggesting that a more comprehensive user study with diverse participants could provide deeper insights into the platform's effectiveness. Additionally, since our implementation was specific to the DiGaBo platform, the technical solutions may need to be adapted for other platforms, even though the underlying principles remain broadly applicable. The speech-to-text functionality showed varying support across different browsers, with optimal performance limited to Chromium-based browsers, which restricts universal accessibility. Furthermore, balancing aesthetic appeal with strict accessibility guidelines required compromises in certain areas, potentially impacting the overall user experience for some segments of our target audience.

6.5 Future Work

- **User Studies:** Conducting longitudinal studies in real-world educational settings would provide valuable insights into the long-term impacts of the implemented features on learning outcomes. This data could help refine the design principles and lead to the development of best practices for accessible educational technologies.
- **AI Integration:** Investigating AI potential for personalizing learning experiences while maintaining accessibility, such as real-time captioning for videos.

- **Mobile Optimization:** Enhancing the mobile experience of educational gamebooks, ensuring that accessibility features are fully functional across all devices.
- **Enhanced Personalization:** Future work could focus on developing more adaptive interfaces that learn from individual user interactions and automatically adjust settings such as speech rate, theme intensity, and language preferences. Such personalization would further enhance the inclusivity of digital educational tools.
- **Collaboration with Educational Institutions:** Working closely with schools and universities to pilot the enhanced platform in actual classrooms would facilitate the collection of extensive user feedback and drive iterative improvements.
- **Offline and Cross-Browser Solutions:** Develop offline-compatible STT/TTS using on-device models (e.g., Mozilla DeepSpeech) to overcome browser limitations.
- **Global Expansion:** Extend multilingual support to underrepresented languages and integrate cultural considerations into gamification design.

6.6 Final Thoughts

This thesis has demonstrated that it is not only possible but essential to create educational gamebooks that are both engaging and accessible. By enhancing accessibility and usability, we open up new possibilities for inclusive education, allowing a wider range of learners to benefit from innovative digital learning tools.

The work conducted here serves as a stepping stone towards more inclusive educational technologies. It challenges developers, educators, and researchers to consider accessibility and usability not as afterthoughts, but as fundamental aspects of educational technology design.

As we move forward in the digital age, the principles and methodologies outlined in this thesis can guide the development of future educational technologies. By continuing to prioritize accessibility and usability alongside engaging gamification elements, we can create a more inclusive digital learning landscape that truly serves all learners.

In conclusion, this thesis contributes to the ongoing evolution of educational technology, pushing it towards a future where engaging, gamified learning experiences are accessible to everyone, regardless of their abilities or background. It is a call to action for the educational technology community to embrace inclusive design principles and create learning tools that leave no learner behind.

Appendix A Bibliography

[ACDM+06] Carmelo Ardito, Maria Francesca Costabile, Maria De Marsico, Rosa Lanzilotti, Stefano Levialdi, Teresa Roselli, and Veronica Rossano. Usability of e-learning tools. *Proceedings of the working conference on Advanced visual interfaces*, pages 80–84, 2006.

[AMKP04] Chadia Abras, Diane Maloney-Krichmar, and Jenny Preece. User-centered design. *Bainbridge, W. Encyclopedia of Human-Computer Interaction. Thousand Oaks: Sage Publications*, 37(4):445–456, 2004.

[BD17] Patrick Buckley and Elaine Doyle. An empirical study of gamification frameworks. *Journal of Organizational and End User Computing (JOEUC)*, 29(1):1–22, 2017.

[BM07] Peter Brusilovsky and Eva Millán. Adaptive and intelligent web-based educational systems. *International Journal of Artificial Intelligence in Education*, 17(2):159–172, 2007.

[Bro96] John Brooke. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[Bur15] Sheryl E Burgstahler. *Universal design in higher education: From principles to practice*. Harvard Education Press, 2015.

[CAS18] CAST. Universal design for learning guidelines version 2.2, 2018.

[CCJ12] Martyn Cooper, Chetz Colwell, and Anne Jelfs. Web accessibility guidelines for the classroom. *Journal of Applied Research in Higher Education*, 2012.

[CDN88] John P Chin, Virginia A Diehl, and Kent L Norman. Development of a tool for measuring user satisfaction of the human-computer interface. pages 213–218, 1988.

[Csi90] Mihaly Csikszentmihalyi. *Flow: The psychology of optimal experience*. Harper & Row, 1990.

[DDKN11] Sebastian Deterding, Dan Dixon, Rilla Khaled, and Lennart Nacke. From game design elements to gamefulness: defining gamification. *Proceedings of the 15th international academic MindTrek conference: Envisioning future media environments*, pages 9–15, 2011.

[Ded09] Chris Dede. Immersive interfaces for engagement and learning. *science*, 323(5910):66–69, 2009.

[DSdNDM+13] Adrián Domínguez, Joseba Saenz-de Navarrete, Luis De-Marcos, Luis Fernández-Sanz, Carmen Pagés, and José-Javier Martínez-Herráiz. Gamifying learning experiences: Practical implications and outcomes. *Computers & education*, 63:380–392, 2013.

[GCdMGL17] Antonio Garcia-Cabot, Luis de Marcos, and Eva Garcia-Lopez. Accessible gamification in education: A systematic literature review. *Proceedings of the 7th International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Info-exclusion*, pages 1–8, 2017.

[Hal04] Edward T Hall. Culture, context, and behavior. *Journal of personality and social psychology*, 86(1):57, 2004.

[HF15] Michael D Hanus and Jesse Fox. Assessing the effects of gamification in the classroom: A longitudinal study on intrinsic motivation, social comparison, satisfaction, effort, and academic performance. *Computers & education*, 80:152–161, 2015.

[HK14] Juho Hamari and Jonna Koivisto. Measuring flow in gamification: Dispositional flow scale-2. *Computers in Human Behavior*, 40:133–143, 2014.

[Int18] International Organization for Standardization. Ergonomics of human-system interaction – part 11: Usability: Definitions and concepts. (ISO 9241-11:2018), 2018.

[JSV] JSVerse. Transloco - angular internationalization library. `https://github.com/jsverse/transloco/`. Accessed: 2025-02-21.

[Kap12] Karl M Kapp. *The gamification of learning and instruction: game-based methods and strategies for training and education*. John Wiley & Sons, 2012.

[Kel10] John M Keller. *Motivational design for learning and performance: The ARCS model approach*. Springer Science & Business Media, 2010.

[KH19] Jonna Koivisto and Juho Hamari. The rise of motivational information systems: A review of gamification research. *International Journal of Information Management*, 45:191–210, 2019.

[Kir98] Jurek Kirakowski. The use of questionnaire methods for usability assessment. *Unpublished manuscript*, 1998.

[KSLB18] Sangkyun Kim, Kibong Song, Barbara Lockee, and John Burton. Gamification in learning and education. In *Gamification in learning and education*, pages 25–38. Springer, 2018.

[Lan14] Richard N Landers. Developing a theory of gamified learning: Linking serious games and gamification of learning. *Simulation & gaming*, 45(6):752–768, 2014.

[LFH17] Jonathan Lazar, Jinjuan Heidi Feng, and Harry Hochheiser. Research methods in human-computer interaction. 2017.

[LL14] Richard N Landers and Amy K Landers. An empirical test of the theory of gamified learning: The effect of leaderboards on time-on-task and academic performance. *Simulation & Gaming*, 45(6):769–785, 2014.

[moz] Speech api - speech recognition - mozillawiki.

[Nie94] Jakob Nielsen. Enhancing the explanatory power of usability heuristics. pages 152–158, 1994.

[Nie12] Jakob Nielsen. *Usability 101: Introduction to usability*. Nielsen Norman Group, 2012.

[ntt] ngx-translate team. ngx-translate. `https://github.com/ngx-translate/core`. Accessed: 2025-02-21.

[PHK09] Helen Petrie, Fiona Hamilton, and Neil King. An investigation into the accessibility of web-based information for people with print disabilities. *Proceedings of the 11th international ACM SIGACCESS conference on Computers and accessibility*, pages 145–152, 2009.

[QC16] Meihua Qian and Karen R Clark. Game-based learning and 21st century skills: A review of recent research. *Computers in Human Behavior*, 63:50–58, 2016.

[Ram20] Aldo González Ramírez. Answer to “google speech to text available offline?”, September 2020.

[RD00] Richard M Ryan and Edward L Deci. Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. *American psychologist*, 55(1):68, 2000.

[Sea13] Jane Seale. *E-learning and disability in higher education: accessibility research and practice*. Routledge, 2013.

[SH20] Michael Sailer and Lutz Homner. Gamification of in-class activities in flipped classroom lectures. *British Journal of Educational Technology*, 51(4):1290–1303, 2020.

[SHMM17] Michael Sailer, Jan Ulrich Hense, Sarah Katharina Mayr, and Heinz Mandl. How gamification motivates: An experimental study of the effects of specific game design elements on psychological need satisfaction. *Computers in Human Behavior*, 69:371–380, 2017.

[SP00] David Squires and Jenny Preece. Usability and learning: evaluating the potential of educational software. *Computers & Education*, 34(1):15–22, 2000.

[Sta] Statista. Online education - worldwide. `https://www.statista.com/outlook/emo/online-education/worldwide?currency=EUR`. Accessed: 2025-2-6.

[SVMP98] John Sweller, Jeroen JG Van Merrienboer, and Fred GWC Paas. Cognitive architecture and instructional design. *Educational psychology review*, 10(3):251–296, 1998.

[TA08] Tom Tullis and Bill Albert. *Measuring the user experience: collecting, analyzing, and presenting usability metrics*. Morgan Kaufmann, 2008.

[tea] Angular team. Angular internationalization (i18n). `https://angular.io/guide/i18n`. Accessed: 2025-02-21.

[TKL18] Crystal Han-Huei Tsay, Alexander Kofinas, and Jing Luo. The impact of gamification on learning and instruction: A systematic review of empirical evidence. *Educational Research Review*, 25:1–17, 2018.

[TVI18] Armando M Toda, Pedro HT Valle, and Seiji Isotani. Dark patterns in the design of games. *arXiv preprint arXiv:1810.08581*, 2018.

[W3C18] W3C. Web content accessibility guidelines (wcag) 2.1. *W3C Recommendation*, 2018.

[YFHJ11] Bei Yuan, Eelke Folmer, and Frederick C Harris Jr. Game accessibility: a survey. *Universal Access in the Information Society*, 10(1):81–100, 2011.

[ZP09] Panagiotis Zaharias and Angeliki Poylymenakou. Developing a usability evaluation method for e-learning applications: Beyond functional usability. *International Journal of Human–Computer Interaction*, 25(1):75–98, 2009.

Declaration:

1. Enhancing the structure and flow of the Literature Review section, particularly in synthesizing information from multiple sources [tool: ChatGPT]
2. Assisting in the analysis and interpretation of automated accessibility testing results (WAVE and Lighthouse) in Chapter 5, providing deeper insights into the improvements made [tool: ChatGPT]
3. Generating code snippets and suggesting optimizations for the implementation of accessibility features, including multilanguage support, theming, and speech-to-text functionality [tool: GitHub Copilot]
4. Proofreading and suggesting minor language improvements throughout the thesis to ensure clarity and consistency [tool: Grammarly]

Eidesstattliche Versicherung

Shanker, Ritwik	441957
Name, Vorname	Matrikelnummer (freiwillige Angabe)

Ich versichere hiermit an Eides Statt, dass ich die vorliegende Arbeit/Bachelorarbeit/ Masterarbeit* mit dem Titel

Accessibility and Usability Enhancements for Educational Gamebooks

ohne unzulässige fremde Hilfe (insbes. akademisches Ghostwriting) erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt. Für den Fall, dass die Arbeit zusätzlich auf einem Datenträger eingereicht wird, erkläre ich, dass die schriftliche und die elektronische Form vollständig übereinstimmen. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Aachen , February 26, 2025	
Ort, Datum	Unterschrift

*Nichtzutreffendes bitte streichen

Belehrung:

§ 156 StGB: Falsche Versicherung an Eides Statt
Wer vor einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.

§ 161 StGB: Fahrlässiger Falscheid; fahrlässige falsche Versicherung an Eides Statt
(1) Wenn eine der in den §§ 154 bis 156 bezeichneten Handlungen aus Fahrlässigkeit begangen worden ist, so tritt Freiheitsstrafe bis zu einem Jahr oder Geldstrafe ein.
(2) Straflosigkeit tritt ein, wenn der Täter die falsche Angabe rechtzeitig berichtigt. Die Vorschriften des § 158 Abs. 2 und 3 gelten entsprechend.

Die vorstehende Belehrung habe ich zur Kenntnis genommen:

Aachen , February 26, 2025	
Ort, Datum	Unterschrift

Acknowledgement

Thank God, this is over :)