Shift-Left Accessibility for User Experience and User Interface Design: A Figma Preflight versus Code-Time Checks (Pilot)

Noelynn Faith Batalingaya

October 12, 2025

1 Introduction

Accessibility in technology is vital because digital tools shape nearly every part of modern life. People use websites and applications to study, work, receive healthcare, shop, and connect with others. When these systems are not accessible, individuals with disabilities are excluded from opportunities others enjoy. Accessibility is not just an additional feature; it is essential for fairness and inclusion.

The lack of accessibility has real-world consequences. A student who relies on a screen reader may be unable to complete an online exam. A worker who cannot use a mouse may struggle to submit a job application. A person with low vision may be unable to read critical health information if the text contrast is too low. These examples show that inaccessible design blocks access to education, employment, and essential services. This is why international standards, such as the Web Content Accessibility Guidelines, were created.

Even with the Web Content Accessibility Guidelines, many digital products remain inaccessible. Research shows that many accessibility issues start with design decisions. Designers may use

colors with low contrast, forget image labels, or create confusing navigation structures. Once these choices are implemented in code, fixing them becomes costly and frustrating. This project uses a different approach: it applies a software engineering method called shift-left testing. Shift-left testing means testing earlier, when changes are easier and cheaper. Instead of waiting for code-time tools like Lighthouse or Accessibility Engine Core (axe-core) to find issues, this project develops a Figma plugin that performs accessibility preflight checks during design. The plugin highlights issues such as poor contrast or missing alternative text and provides suggestions before coding begins. The main research question is: can early design checks reduce accessibility barriers later and support the creation of more inclusive technology?

2 Background

Accessibility means making digital products usable by people with diverse abilities. Some use screen readers, others need captions or rely on large buttons and voice commands. Accessibility ensures that systems adapt to different ways of interacting. The Web Content Accessibility Guidelines define accessibility through four main principles: perceivable, operable, understandable, and robust.

- **Perceivable:** Content should be presented in ways that users can see or hear. Images need alternative text, videos should have captions, and text must have adequate color contrast.
- **Operable:** Users should be able to interact with the system using multiple input methods, such as keyboards, voice, or assistive technologies.
- **Understandable:** The design should be predictable and clear. Forms should explain errors, and navigation should remain consistent.
- **Robust:** Content must work across various devices and assistive technologies, from desktop screen readers to mobile zoom features.

The Web Content Accessibility Guidelines also define three compliance levels: Level A, Level AA, and Level AAA. Level A covers basic accessibility, such as adding text alternatives and keyboard navigation. Level AA, the most common target, requires higher contrast and captions for videos. Level AAA adds stricter standards, like sign language interpretation and even higher contrast ratios. Although Level AAA represents best practice, achieving it universally is difficult, so Level AA is often considered a balanced goal.

Timing is crucial. In many workflows, accessibility is checked late, when products are almost complete. Fixing problems at that point is expensive. Shift-left testing addresses this issue by detecting problems early. Accessibility should follow the same approach already used in security and performance testing.

Figma is an ideal platform for this project. It is widely used by design teams, supports collaboration, and allows custom plugins. Tools such as Stark (for contrast) and Able (for alternative text) exist but handle only one issue at a time. What is missing is a single plugin that performs multiple checks and integrates accessibility thinking into every design stage.

2.1 Planned Software Deliverable

The main deliverable of this project is a Figma plugin that runs accessibility checks on design files. The plugin scans text, shapes, and components using accessibility rules and shows the results directly on the canvas.

The plugin checks for:

- **Color contrast:** Ensures text and background colors meet Web Content Accessibility Guidelines standards.
- **Alternative text:** Flags images and icons without descriptions.
- **Focus order:** Confirms that interactive elements follow a logical keyboard navigation order.
- **Target size:** Highlights buttons or touch areas that are too small.

- **Use of color:** Warns when color alone conveys meaning.

When the plugin runs, it lists all issues and links each warning to the exact element. Designers can click, adjust, and rerun checks. For instance, if text fails a contrast test, the plugin suggests accessible color options. If an icon lacks alternative text, it prompts the designer to add one. This creates a feedback loop that makes accessibility both interactive and educational.

3 Related Work

Most accessibility tools today focus on code-time checks. Accessibility Engine Core (axe-core), Lighthouse, and Web Accessibility Evaluation Tool (WAVE) are popular for scanning finished websites and applications. They detect problems like missing Accessible Rich Internet Applications (ARIA) labels or poor heading structure but do so too late in the process. Design-stage tools are fewer and narrower. Stark addresses contrast, and Able helps with alternative text. Academic work on design-phase evaluation exists but is not widely adopted.

Cultural and workflow barriers also play a role. Many designers assume accessibility is the developer's job. If tools are external to their workflow, they are rarely used. Moreover, accessibility is sometimes treated as a legal requirement rather than a design value. Studies show tools work best when they explain why an issue matters, helping designers learn as they correct mistakes.

This project combines those insights. It builds a tool that integrates multiple checks, explains their importance, and works directly inside Figma. It promotes a culture where accessibility is a shared responsibility.

3.1 Future Work

This pilot can expand in several directions. Future versions could include more Web Content Accessibility Guidelines checks, such as motion sensitivity and device orientation. User studies could evaluate how the plugin affects workflows, reduces defects, and increases awareness.

The plugin could also integrate with code-time tools like Accessibility Engine Core, creating an end-to-end accessibility pipeline. In education, it could train students to recognize accessibility issues early. These expansions would make the tool more comprehensive and provide stronger evidence for shift-left accessibility.

4 Conclusion

Accessibility should not be an afterthought. When ignored early, problems multiply and exclude users. This project shifts accessibility left, embedding it into the design stage through a Figma plugin. The plugin identifies issues like poor contrast, missing labels, and confusing focus order while designs are still being created.

This work contributes both practically and conceptually. It offers a working tool and demonstrates how shift-left testing applies to accessibility. It also reinforces the idea that inclusive design is good design. By normalizing accessibility early, digital products can serve more people fairly.

This junior project lays the groundwork for senior research. Future studies can test the plugin in real teams and measure its impact. Doing so would not only improve the tool but also prove that early checks lead to more inclusive technology. In this way, the project strengthens both computer science research and equity in technology.