Accessiblü
Managed Digital Accessibility Ops

Accessibility Evaluation Report:

COVE Studio

September 30, 2025

Conducted by: Accessiblü, LLC

For: Library Accessibility Alliance (LAA)

Prepared by:
Accessiblü, LLC
882 Pompton Ave, STE A-2
Cedar Grove, NJ 07009
hey@accessiblu.com

Primary Contact:
Jeff Rodgers, MS Ed
Director of Digital Accessibility
Accessiblü
jeff@accessiblu.com

Table of Contents

Summary of Accessibility Findings

Accessiblü conducted a **high-level accessibility evaluation** of the COVE Studio platform from COVE Collective to assess its usability for individuals with disabilities. The review was conducted using automated testing tools, keyboard-only navigation, and attempted manual inspection for conformance to select WCAG 2.2 AA success criteria.

Important Note: During initial testing, a critical modal accessibility barrier prevented screen reader access to the platform. The COVE development team promptly resolved this issue, allowing our auditors to complete comprehensive testing of the platform's core functionality.

Key Findings

The COVE Studio platform represents an ambitious effort to create a collaborative digital humanities workspace. While the platform demonstrates thoughtful design in certain areas, our testing revealed **some** accessibility barriers that **may** create substantial challenges for users who rely on assistive technology.

The evaluation identified multiple critical issues, including extensive keyboard traps throughout the interface, improper semantic structure, missing state announcements for interactive elements, and inadequate alternative text for visual content. These barriers collectively prevent screen reader users and keyboard-only users from effectively navigating, understanding, and interacting with the platform's collaborative features.

Positively, the team's responsive approach to addressing the initial modal barrier demonstrates a commitment to accessibility that, if sustained through systematic remediation efforts, could transform COVE Studio into a truly inclusive digital humanities tool.

Top 3 Issues Identified

1. **Pervasive Keyboard Traps in Menus and Dialogs**

   - Multiple expandable menus, filters, and interface controls trap keyboard focus, requiring users to repeatedly press the Escape key (sometimes 4-5 times) to exit. Some elements create complete keyboard traps with no escape mechanism.
   - Impact: Keyboard-only users and screen reader users become repeatedly stuck in interface elements, creating significant frustration and preventing efficient workflow. The unpredictable behavior makes the platform nearly unusable for these user groups.
   - WCAG Success Criteria: 2.1.2 No Keyboard Trap (A), 2.1.1 Keyboard (A).

2. **Missing State Announcements and ARIA Attributes**

- Interactive elements including menus, tab panels, and collapsible sections fail to announce their expanded or collapsed states. Content changes occur without programmatic notification to assistive technology.
- Impact: Screen reader users cannot determine whether activating controls produced the expected results, leading to confusion about interface state and content availability. Users may miss critical content updates or attempt to interact with collapsed elements.
- WCAG Success Criteria: 4.1.2 Name, Role, Value (A), 4.1.3 Status Messages (AA).

3. **Improper Semantic Structure and Roles**

- Critical interface components are coded incorrectly, including tabs implemented as static text, buttons mislabeled as links, improper heading hierarchy with multiple H1s and out-of-sequence headings, and multiple banner regions.
- Impact: Screen reader users rely on semantic structure for navigation and comprehension. Incorrect implementation forces users to navigate linearly through content, missing the efficient navigation patterns that proper semantic markup provides. This significantly increases cognitive load and time required to complete tasks.
- WCAG Success Criteria: 1.3.1 Info and Relationships (A), 2.4.1 Bypass Blocks (A), 2.4.6 Headings and Labels (AA).

Disabilities Impacted

Blind and Low-Vision Users

- **Issues:** Keyboard traps in menus and dialogs, missing alternative text for images and visual content, improper heading structure preventing efficient navigation, missing state announcements when interface elements expand or collapse, no programmatic association between annotations and referenced text, inaccessible tooltip content, tabs and interactive elements coded as static text.
- **Impact:** Screen reader users experience difficulty navigating the platform due to repeated keyboard traps and missing semantic information. The inability to understand interface state changes or navigate efficiently through content makes collaborative work extremely challenging. Critical visual information in images and annotations remains inaccessible, preventing full participation in digital humanities projects.

Users with Motor Disabilities

- **Issues:** Extensive keyboard traps requiring repeated Escape key presses, drag-and-drop functionality with no keyboard alternative, focus management issues causing unexpected jumps to top of page, radio button controls creating navigation loops, inaccessible close buttons on dialogs.
- **Impact:** Keyboard-only users face constant interruptions from keyboard traps, making simple navigation exhausting and frustrating. The need to repeatedly press keys to escape menus creates physical strain for users with limited dexterity. Inaccessible drag-and-drop features prevent document organization, while unpredictable focus management disrupts workflow.

Users with Cognitive Disabilities

- **Issues**: Inconsistent interface behavior, unpredictable focus management, color-only indicators for annotations, confusing navigation patterns where tabs function as links, unclear button labels, and missing feedback when actions are performed.
- **Impact**: Users with cognitive disabilities struggle to develop reliable mental models of interface behavior due to inconsistent patterns. The lack of clear feedback and unpredictable responses increases cognitive load and creates confusion about whether actions succeeded. Missing visual and programmatic associations between related content fragments for understanding.

Page-Specific Findings and Impact Analysis

The following section lists the accessibility findings by page and WCAG violations and describes their impact on users.

COVE Studio Landing Page

Issue	WCAG Success Criteria	Description	Example
Improper Heading Hierarchy	**1.3.1 Info and Relationships (A), 2.4.6 Headings and Labels (AA)**	Heading structure begins with H2 elements before H1, with the H1 appearing as the last heading on the page. Multiple H1 headings exist where only one should be present.	Page starts with "My Projects" and "Groups" as H2, while "LAA Testing" H1 appears later in the document flow.
Multiple Banner Regions	**1.3.1 Info and Relationships (A)**	Page contains multiple elements identified as banner regions, violating the requirement for a single banner per page.	Screen reader announces "Banner region" twice when navigating page landmarks.
Missing Alternative Text	**1.1.1 Non-text Content (A)**	Project thumbnails and interface graphics lack descriptive alternative text or use generic placeholder descriptions.	Unlabeled graphics announced without context, providing no information about project content or purpose.
Keyboard Traps in Menus	**2.1.2 No Keyboard Trap (A)**	Profile menu, notification dialog, view filters, and sort controls trap keyboard focus, requiring multiple Escape key presses to exit.	Activating "Sort: Name" menu requires pressing Escape 3-4 times to close and return to normal navigation.
Missing State Announcements	**4.1.2 Name, Role, Value (A)**	Expandable menus and buttons do not announce collapsed or expanded states to screen readers.	"View: Active Projects" menu announces as collapsed but does not announce when expanded upon activation.

Impact Summary:

The landing page presents navigation challenges for screen reader users due to improper heading structure and repeated keyboard traps. Users cannot efficiently navigate between projects using heading navigation, and attempting to use filtering or sorting controls results in becoming trapped in menus. The missing state announcements leave users uncertain whether interface elements have responded to their actions, requiring repeated experimentation to understand interface behavior.

COVE Studio Landing Page Screenshot

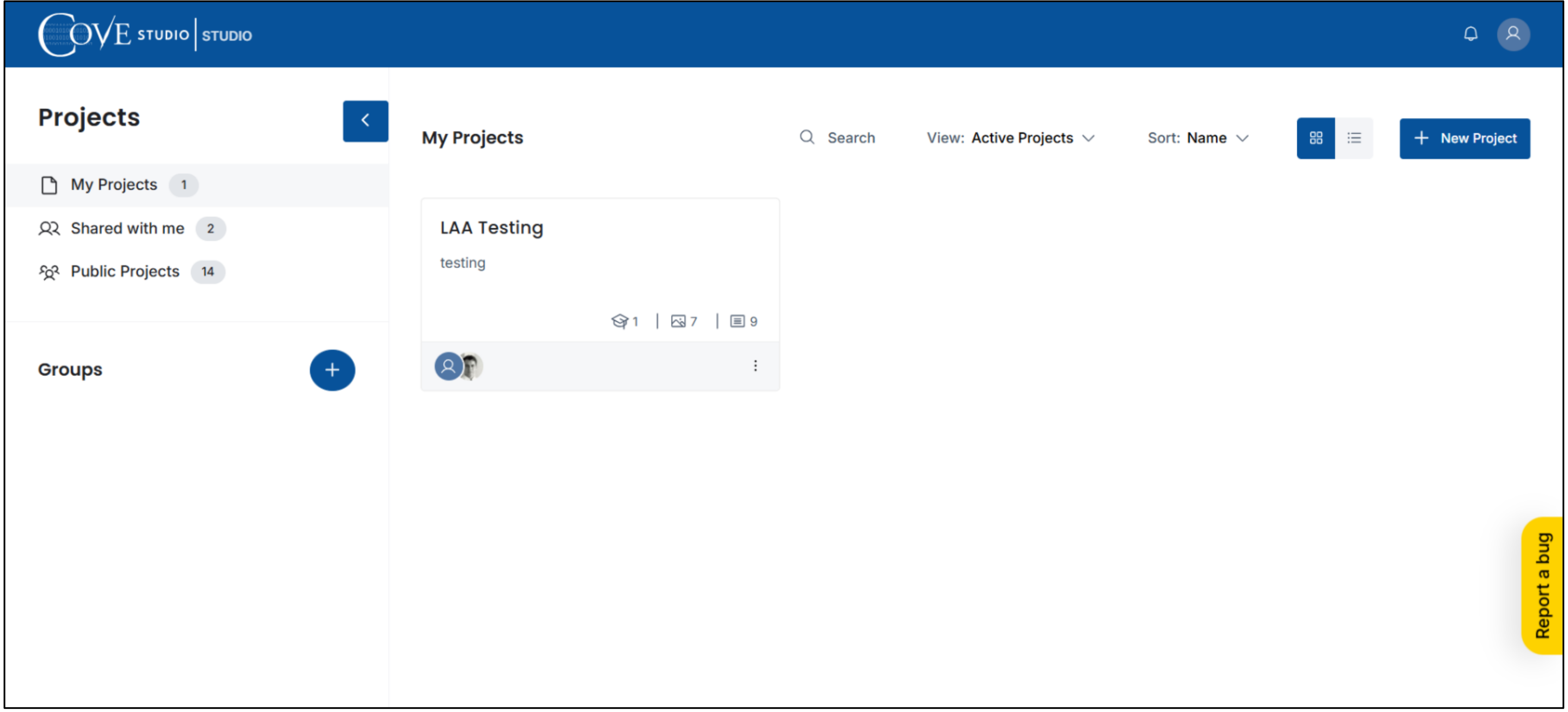


Figure 1. The COVE Studio main landing page displays project organization and collaboration features.

LAA Testing Project Page

Issue	WCAG Success Criteria	Description	Example
Tabs Coded as Static Text	**1.3.1 Info and Relationships (A), 4.1.2 Name, Role, Value (A)**	"My Projects," "Shared with me," and "Public Projects" tabs are announced as static text with no indication of interactivity or tab role.	Screen reader announces "Blank. My projects 1. Blank. Shared with me 2" as plain text rather than as interactive tab controls.
Missing Live Regions	**4.1.3 Status Messages (AA)**	When tabs are activated, content changes without programmatic announcement to screen readers.	Activating "Shared with me" tab updates content but provides no notification that content has changed or loaded.
Buttons Mislabeled as Links	**1.3.1 Info and Relationships (A)**	"Users" and "Settings" controls are announced as buttons but function as links to new pages.	"Users button" navigates to new page instead of performing button action.
Inaccessible Radio Buttons	**2.1.1 Keyboard (A), 4.1.2 Name, Role, Value (A)**	"Arrange projects as cards" and "Arrange projects as list" radio buttons are not in tab order and create keyboard navigation loops when accessed.	Arrowing through radio buttons creates loop, preventing navigation to subsequent content.
Document Links Lack Context	**2.4.4 Link Purpose (A), 4.1.2 Name, Role, Value (A)**	Document thumbnail links provide inadequate context, with associated icon information not available to screen readers.	Links announced as generic numbers with no descriptive information about document type or content.

Impact Summary:

The project page's fundamental navigation structure is inaccessible to screen reader users. Tabs coded as static text prevent users from understanding available navigation options or efficiently switching between project views. The lack of live region announcements means users receive no feedback when content changes, forcing them to manually search for updated content. Radio button controls that should allow view switching are completely inaccessible, while document links provide insufficient context for informed selection.

LAA Testing Project Page Screenshot

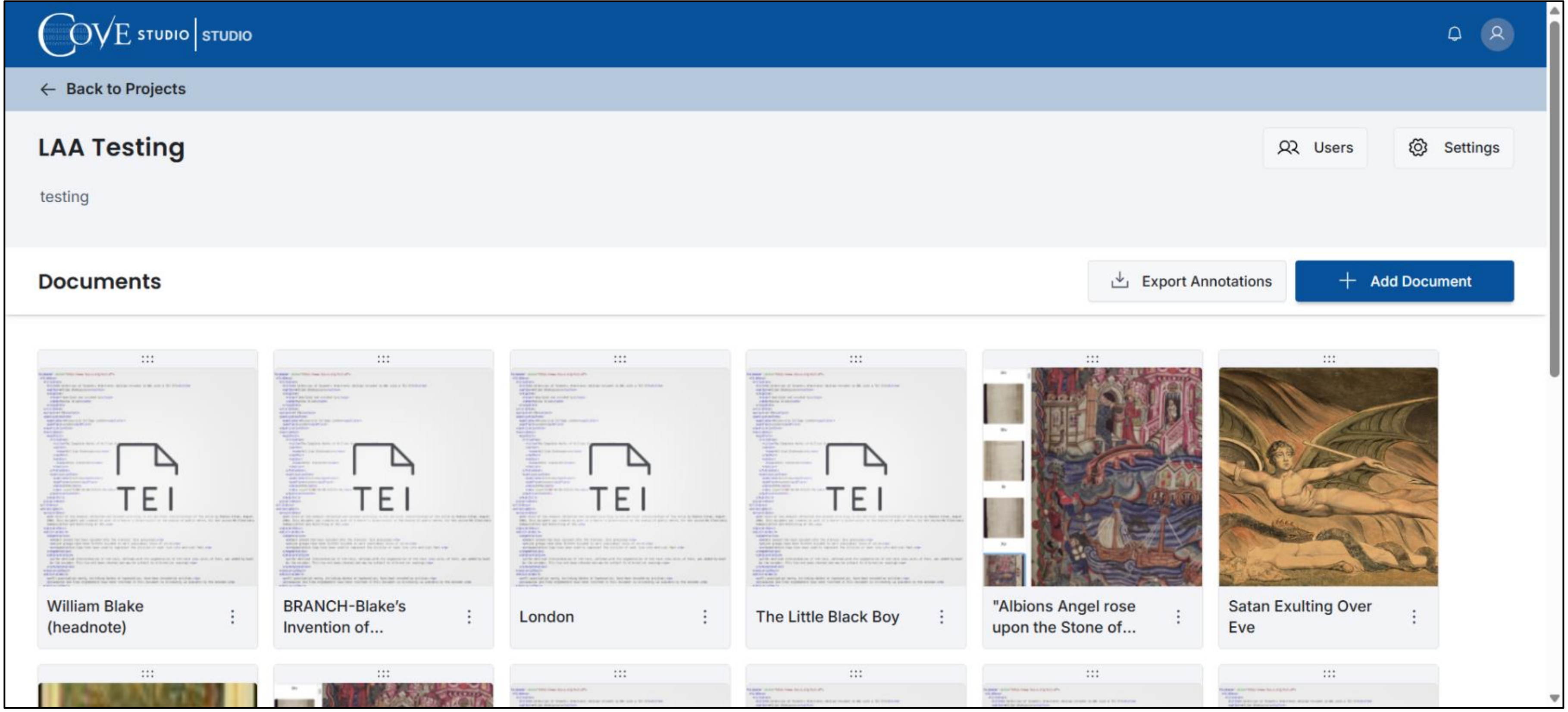


Figure 1. The LAA Testing project page displays various historical documents and images in a collaborative workspace format.

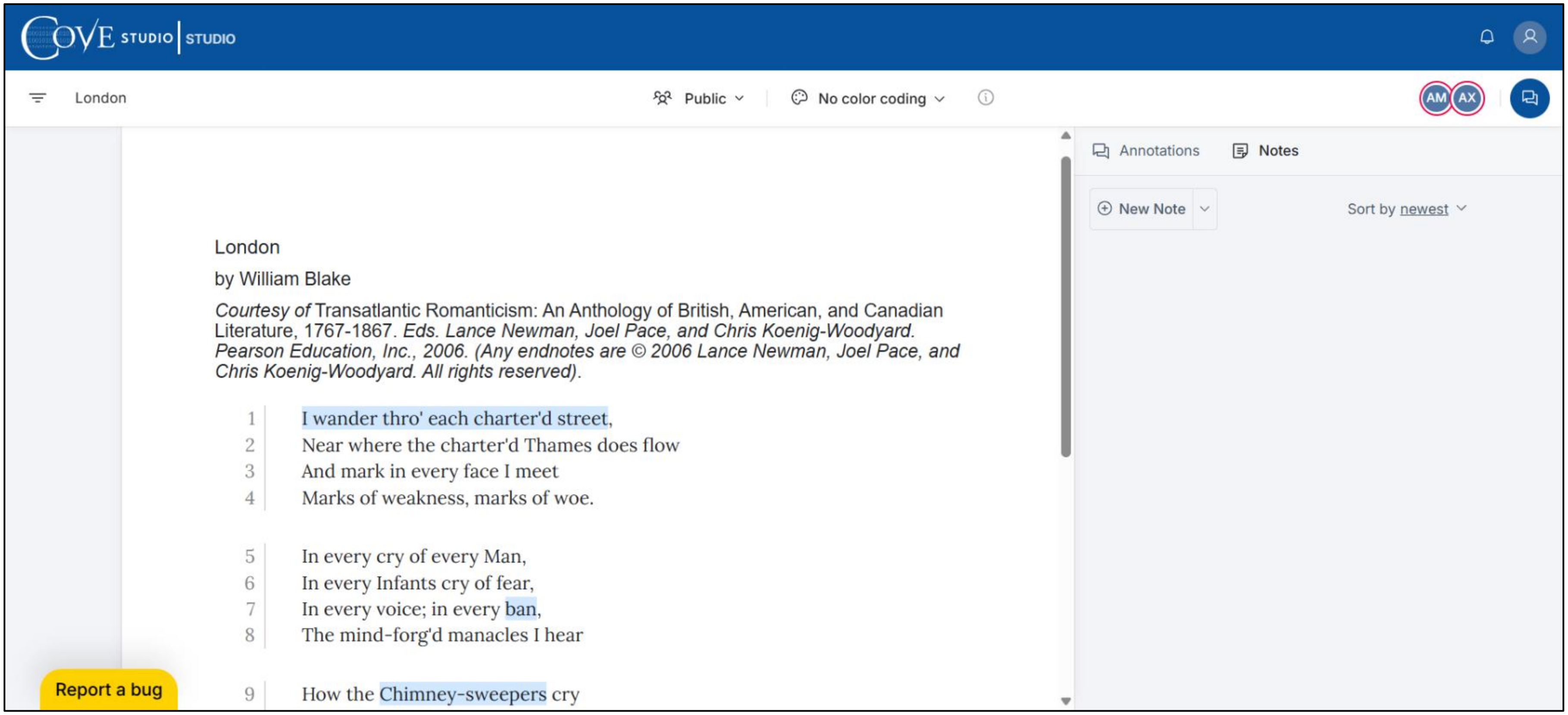
London Document Page

Issue	WCAG Success Criteria	Description	Example
Partial Keyboard Trap in Menus	**2.1.2 No Keyboard Trap (A)**	"Public" annotation visibility menu and "Annotation color by" dropdown trap keyboard focus, requiring multiple Escape key presses to exit.	After activating "Public" menu, pressing Escape 3-4 times is required to close and return focus to page.
Buttons Without State	**4.1.2 Name, Role, Value (A)**	Expandable buttons and menus do not announce expanded state after activation.	"Show Annotation list button" activates and displays content but does not announce expansion.
No Association Between Annotations and Text	**1.3.1 Info and Relationships (A)**	Annotations list provides no programmatic or descriptive connection to the lines of text they reference.	Annotation "This is a test annotation number one" provides no indication it references "I wander through each chartered street" line 1.
Color-Only Indicators	**1.4.1 Use of Color (A)**	Annotations use only red highlighting to indicate associated text, with no text or pattern alternative.	Line highlighting uses red color as sole indicator of annotation presence.
Unlabeled Interactive Elements	**4.1.2 Name, Role, Value (A)**	User presence indicators and submit buttons lack accessible labels.	"AM button" and "Unlabeled 8 button" provide no context about their purpose.

Impact Summary:

The document viewing experience is severely compromised for screen reader users. While the platform attempts to provide collaborative annotation features, the complete lack of association between annotations and referenced text makes these features unusable for screen reader users who cannot perceive the visual highlighting. Keyboard traps in annotation controls frustrate attempts to interact with the content, while missing state announcements leave users uncertain about interface changes.

London Document Page Screenshot



London

by William Blake

Courtesy of Transatlantic Romanticism: An Anthology of British, American, and Canadian Literature, 1767-1867. *Eds. Lance Newman, Joel Pace, and Chris Koenig-Woodyard. Pearson Education, Inc., 2006. (Any endnotes are © 2006 Lance Newman, Joel Pace, and Chris Koenig-Woodyard. All rights reserved).*

I wander thro' each charter'd street,
Near where the charter'd Thames does flow
And mark in every face I meet
Marks of weakness, marks of woe.

In every cry of every Man,
In every Infants cry of fear,
In every voice; in every ban,
The mind-forg'd manacles I hear

How the Chimney-sweepers cry

Figure 2. The London document viewing page shows William Blake's poem with annotation features and collaborative tools.

Moses Receiving the Law Page

Images Not Recognized	**1.1.1 Non-text Content (A), 4.1.2 Name, Role, Value (A)**	Primary image content is announced only as static text labels with no indication that images are present.	Screen reader announces "Recto. Crop to Image" as plain text; image itself is not accessible or identified.
Inaccessible Image Controls	**2.1.1 Keyboard (A), 4.1.2 Name, Role, Value (A)**	Image viewing controls and labels cannot be selected or interacted with via keyboard.	"Recto" and "Crop to Image" labels are not interactive; underlying images cannot be accessed.
Missing Alternative Text	**1.1.1 Non-text Content (A)**	Historical artwork images lack any descriptive alternative text or metadata.	Images display visual content with no textual description available to screen readers.

Impact Summary:

The image viewing functionality is completely inaccessible to screen reader users. Primary visual content is not properly identified as images, and no alternative text is provided to convey the content or context of the historical artwork. This creates a complete barrier to accessing visual materials, fundamentally undermining the platform's purpose for users with visual disabilities.

Moses Receiving the Law Page Screenshot

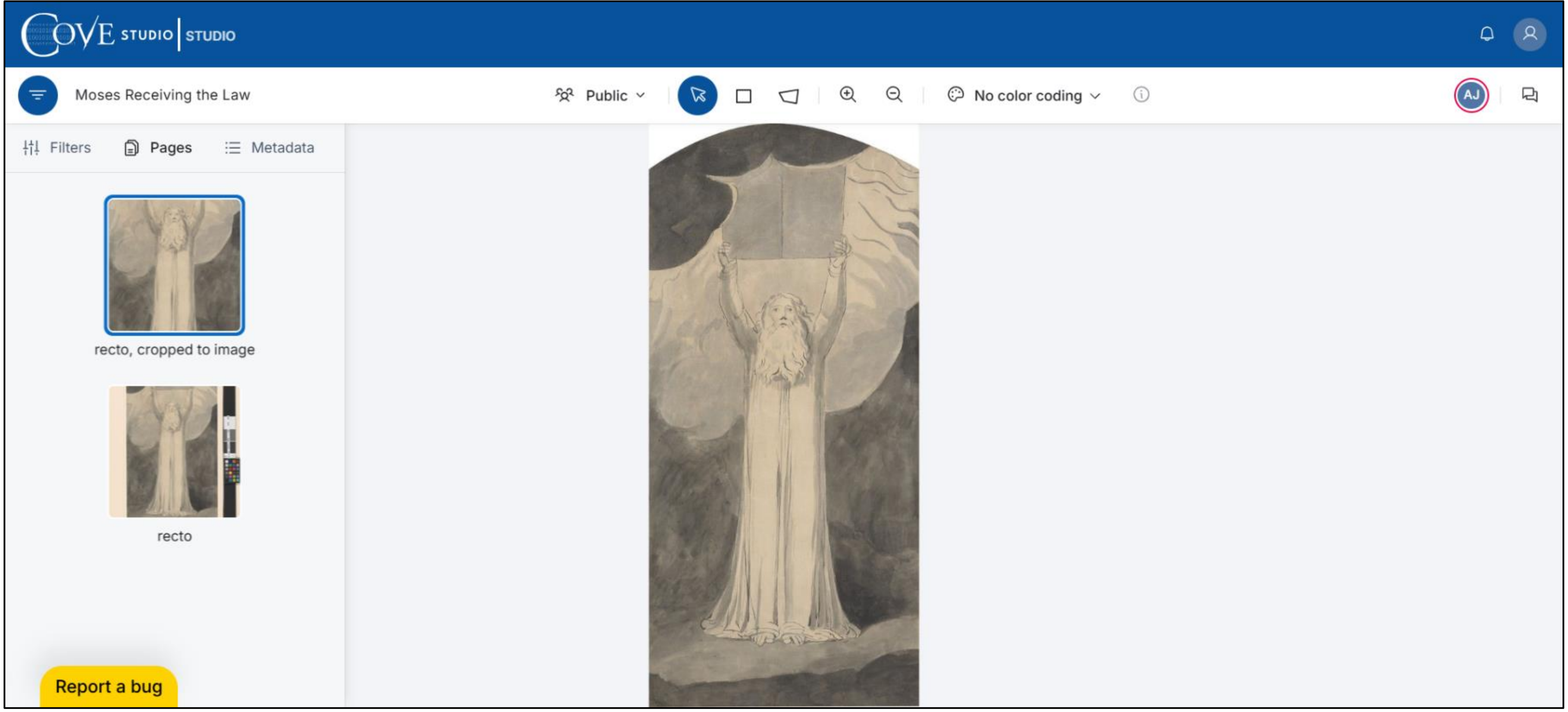


Figure 3. The Moses Receiving the Law image viewer shows historical religious artwork with page navigation thumbnails on the left.

Additional Critical Issues Identified

Focus Management Problems

- Closing dialogs with cancel button sends focus to top of page rather than returning to trigger element (WCAG 2.4.3 Focus Order).
- Unpredictable focus behavior creates disorientation and requires users to relocate their position repeatedly.

Inaccessible Drag and Drop

- Document reordering functionality uses drag and drop with no keyboard alternative (WCAG 2.1.1 Keyboard).
- Attempting to interact with "Rearrange this document by dragging" creates confusion and potential keyboard traps.

Tooltip Content Inaccessibility

- User presence indicators show information only on hover with no keyboard access (WCAG 4.1.2 Name, Role, Value).
- Critical context about collaborators is hidden from screen reader users.

Report a Bug Feature

- Activating "Report a bug" button opens chat interface without announcement or focus management (WCAG 4.1.3 Status Messages).
- Users navigate behind the dialog, unaware it has opened.

Keyboard Trap in Menus (Critical Priority)

html

```
<!-- Current problematic implementation -->
<button id="sortMenu" onclick="toggleMenu()">Sort: Name</button>
<div class="menu-content">
  <a href="#name">Name</a>
  <a href="#newest">Newest</a>
  <a href="#oldest">Oldest</a>
</div>

<!-- Recommended fix -->
<button
  id="sortMenu"
  aria-expanded="false"
  aria-haspopup="true"
  aria-controls="sortMenuContent"
  onclick="toggleMenu(this)">
  Sort: Name
</button>
<div
  id="sortMenuContent"
  class="menu-content"
  role="menu"
  hidden>
  <a href="#name" role="menuitem">Name</a>
  <a href="#newest" role="menuitem">Newest</a>
  <a href="#oldest" role="menuitem">Oldest</a>
</div>

<script>
function toggleMenu(button) {
  const menu = document.getElementById(button.getAttribute('aria-controls'));
  const isExpanded = button.getAttribute('aria-expanded') === 'true';

  button.setAttribute('aria-expanded', !isExpanded);
  menu.hidden = isExpanded;

  if (!isExpanded) {
    // Move focus to first menu item
    menu.querySelector('[role="menuitem"]').focus();
  }
}

// Handle Escape key to close menu
document.addEventListener('keydown', (e) => {
  if (e.key === 'Escape' && !document.querySelector('.menu-content').hidden) {
    const button = document.querySelector('[aria-expanded="true"]');
    closeMenu(button);
    button.focus(); // Return focus to trigger
  }
});
</script>
```

Tabs Coded as Static Text

html

```
<!-- Current problematic implementation -->
<div class="sidebar-nav">
  <div class="nav-item">My Projects 1</div>
  <div class="nav-item">Shared with me 2</div>
  <div class="nav-item">Public Projects 14</div>
</div>

<!-- Recommended fix -->
<div class="sidebar-nav" role="tablist" aria-label="Project views">
  <button
    role="tab"
    aria-selected="true"
    aria-controls="myProjectsPanel"
    id="myProjectsTab">
    My Projects <span class="count">1</span>
  </button>
  <button
    role="tab"
    aria-selected="false"
    aria-controls="sharedPanel"
    id="sharedTab">
    Shared with me <span class="count">2</span>
  </button>
  <button
```

```
    role="tab"
    aria-selected="false"
    aria-controls="publicPanel"
    id="publicTab">
    Public Projects <span class="count">14</span>
  </button>
</div>

<div role="tabpanel" id="myProjectsPanel" aria-labelledby="myProjectsTab">
  <!-- My Projects content -->
</div>
<div role="tabpanel" id="sharedPanel" aria-labelledby="sharedTab" hidden>
  <!-- Shared with me content -->
</div>
<div role="tabpanel" id="publicPanel" aria-labelledby="publicTab" hidden>
  <!-- Public projects content -->
</div>
```

Missing Alternative Text for Images

html

```
<!-- Current problematic implementation -->
<div class="document-card">
  <div class="thumbnail">
    <!-- Image not properly marked up -->
  </div>
  <div class="label">Recto. Crop to Image</div>
```

```
</div>

<!-- Recommended fix -->
<div class="document-card">
  <img
    src="moses-receiving-law.jpg"
    alt="Moses Receiving the Law, historical religious artwork depicting Moses on Mount Sinai holding stone tablets inscribed with the Ten Commandments, surrounded by divine light and clouds"
    class="thumbnail">
  <div class="controls">
    <button aria-label="View recto (front) side">Recto</button>
    <button aria-label="Crop image to fit window">Crop to Image</button>
  </div>
</div>
```

Missing Live Regions for Dynamic Content

html

```
<!-- Current problematic implementation -->
<div id="projectView">
  <!-- Content updates here via JavaScript with no announcement -->
</div>

<!-- Recommended fix -->
<div id="projectView" role="region" aria-live="polite" aria-relevant="additions removals">
  <!-- Content updates are now announced -->
</div>
```

```
<!-- Status message for actions -->
<div class="sr-only" role="status" aria-live="polite" aria-atomic="true" id="statusMessage"></div>

<script>
function updateProjectView(newContent) {
  document.getElementById('projectView').innerHTML = newContent;
  document.getElementById('statusMessage').textContent = 'Project view updated to show ' + viewType;
}
</script>
```

Proper Heading Hierarchy

html

```
<!-- Current problematic implementation -->
<h2>My Projects</h2>
<h2>Groups</h2>
<h1>LAA Testing</h1> <!-- Wrong: H1 appears after H2s -->

<!-- Recommended fix -->
<h1>COVE Studio Dashboard</h1> <!-- Single H1 for page -->
<h2>My Projects</h2>
<h3>LAA Testing</h3> <!-- Proper hierarchy -->
<h2>Groups</h2>
```

Final Thoughts and Recommendations

COVE Studio represents an ambitious vision for collaborative digital humanities research, with innovative features designed to facilitate annotation, analysis, and scholarly discussion of historical texts and images. The platform's collaborative capabilities and document management features demonstrate thoughtful consideration of academic workflows.

However, the current implementation presents significant accessibility barriers that prevent users who rely on assistive technology from participating in these collaborative activities. The pervasive keyboard traps create frustration and inefficiency for keyboard-only users, while missing semantic markup and state announcements leave screen reader users without critical information about interface behavior and content organization.

The development team's rapid response to the initial modal accessibility issue demonstrates a willingness to address accessibility concerns. This same commitment, applied systematically across the platform, would unlock COVE Studio's potential as an inclusive research tool.

The issues identified are not insurmountable. Most can be addressed through proper implementation of ARIA attributes, semantic HTML, and keyboard interaction patterns. The platform's underlying functionality is sound; the accessibility improvements needed are primarily in the presentation and interaction layers.

Recommended Fixes

Immediate Priority (Blocking Issues)

- **Eliminate keyboard traps** by implementing proper escape mechanisms for all menus and dialogs, ensuring single Escape key press closes elements and returns focus appropriately.
- **Fix semantic markup** by converting pseudo-tabs to proper tab controls with correct ARIA attributes, ensuring single H1 per page, and fixing heading hierarchy throughout.
- **Implement proper focus management** so closing dialogs returns focus to triggering element rather than page top, maintaining user orientation and workflow.

High Priority (Critical Functionality)

- **Add comprehensive ARIA state announcements** for all expandable elements, ensuring "aria-expanded" attribute updates correctly and is announced by screen readers.
- **Implement live regions** for dynamic content updates, particularly tab panel changes and search results, so screen reader users receive notifications of content changes.
- **Provide alternative text for all images** with meaningful descriptions of historical artwork, documents, and interface graphics.
- **Create keyboard alternatives** for drag-and-drop functionality, allowing document reordering through keyboard commands.

Important Priority (Significant Impact)

- **Fix button/link semantics** throughout the platform, ensuring elements that navigate use link markup and elements that perform actions use button markup.
- **Associate annotations with referenced text** programmatically, providing context in annotation lists about which text lines are being annotated.
- **Remove color-only indicators** by adding text labels, patterns, or icons to annotation highlighting to convey information beyond color.
- **Label all interactive elements** descriptively, particularly user presence indicators, action buttons, and menu triggers.
- **Make tooltip content accessible** by providing alternative access method for information currently only available on hover.

Ongoing Improvements

- **Implement consistent interaction patterns** across all menus and controls to reduce cognitive load and create predictable user experience.
- **Conduct user testing with assistive technology users** to validate fixes and identify additional barriers that automated and expert testing may miss.
- **Establish accessibility testing protocols** for development workflow to prevent regression and catch new issues early.
- **Provide accessibility documentation** for users explaining available keyboard shortcuts and assistive technology support.

The investment required to address these accessibility issues would be substantial but achievable. The result would be a platform that serves as a model for inclusive digital humanities tools, opening collaborative scholarly work to researchers and academics across the full spectrum of abilities. Given the academic mission of supporting diverse scholarly communities, this investment aligns strongly with the platform's core values and intended impact.

Disclaimer

Accessiblü prepared this report as a high-level accessibility evaluation of the COVE Studio platform. The evaluation utilized industry-standard testing methodologies, including screen reader testing (JAWS 2025, NVDA), keyboard-only navigation, and manual inspection for select WCAG 2.2 AA success criteria.

This report does not represent a comprehensive WCAG compliance audit and should not be seen as a certification of accessibility compliance. While we have identified significant accessibility concerns and usability barriers, this evaluation was limited in scope and may not encompass all accessibility issues on the platform.

No Legal Liability:

Accessiblü offers this report for informational purposes only. It assumes no legal responsibility for accessibility violations or compliance failures resulting from its use. Organizations seeking formal certification should conduct a comprehensive audit and user testing with individuals with disabilities.

Limitations of Testing:

This evaluation was conducted at a specific time, and platform updates may have occurred after testing was completed. Additionally, while expert review and assistive technology testing were utilized, real-world users with disabilities determine the true measure of accessibility. We recommend conducting user testing with individuals who have disabilities as part of a comprehensive accessibility strategy.