

Todo.md - Collaborative Editor Issues Analysis

DOCUMENTATION Issues

Missing/Incomplete Documentation

1. **API Documentation** - No comprehensive API docs for WASM functions
2. **Development Setup Guide** - While Makefile exists, missing detailed setup instructions
3. **Architecture Overview** - No high-level system architecture documentation
4. **Deployment Guide** - No production deployment instructions
5. **Contributing Guidelines** - No CONTRIBUTING.md file
6. **License Information** - No LICENSE file found

Documentation Inconsistencies

1. **Content Duplication** - README.md has duplicated sections (PromiseGrid features appear twice, folder structure has duplicates)
2. **Version Mismatches** - Documentation references different versions/features than implemented
3. **Outdated Examples** - Some code examples in docs don't match current implementation
4. **Broken Documentation Links** - README.md references `docs/guid-rooms.md` but file is actually `docs/uuid-rooms.md`
5. **Inconsistent Feature Count** - README mentions "23+ shortcuts" while shortcutManager.js implements 51+ shortcuts
6. **Menu Documentation Mismatch** - README claims "Over 30 menu features" while docs/editor-menu.md shows 47 features with different implementation status

CODE BUGS (Definitive Issues)

Critical Bugs

1. **Memory Leak in Remote Cursor Plugin** (`src/ui/remoteCursorPlugin.js:76-80`)

javascript

```
// Unsafe setTimeout in updateDecorations - can cause memory leaks
setTimeout(() => {
  this.view.dispatch({
    effects: setRemoteCursors.of(this.decorations)
  });
}, 0);
```

Issue: No cleanup mechanism for pending timeouts when component is destroyed.

2. XSS Vulnerability in User Names (`src/setup/userSetup.js:43`)

javascript

```
display.textContent = `Current User: ${name}`;
```

Issue: Direct interpolation without sanitization. While `textContent` is safer than `innerHTML`, the template literal pattern is risky.

3. Race Condition in Document Copy (`src/setup/documentCopy.js:28-35`)

javascript

```
setTimeout(() => {
  if (copyData.content && view && ytext) {
    // Operations here
  }
}, 1000);
```

Issue: Fixed 1-second delay is unreliable and can cause race conditions.

4. Improper Error Handling in WASM Init (`src/wasm/initWasm.js:35-41`)

javascript

```
try {
  const testMessage = create_promisegrid_edit_message(/*... */);
  console.log(" PromiseGrid CBOR message created:", testMessage.length, "bytes");
  log_promisegrid_message(testMessage);
} catch (error) {
  console.log(" PromiseGrid test failed:", error);
}
```

Issue: Swallowing errors silently can hide critical initialization failures.

5. Uncaught Promise Rejection (`src/ui/githubCommitDialog.js:380-385`)

javascript

```
const coAuthors = this.getCoAuthors();  
const result = await githubService.commitFile(/*...*/);
```

Issue: No error handling for potential async failures in `getCoAuthors()`.

Functional Bugs

6. Clipboard API Fallback Broken (`src/export/handlers.js:730-740`)

javascript

```
const ta = document.createElement('textarea');  
ta.value = url;  
ta.setAttribute('readonly', '');  
ta.style.position = 'fixed';  
ta.style.left = '-9999px';  
document.body.appendChild(ta);  
ta.select();  
const ok = document.execCommand && document.execCommand('copy');  
document.body.removeChild(ta);
```

Issue: `execCommand('copy')` is deprecated and may fail in newer browsers without proper error handling.

7. Infinite Loop Potential (`src/ui/shortcutManager.js:114-120`)

javascript

```
this.shortcuts.forEach((config, action) => {  
  if (!this.keyToAction.has(config.key)) {  
    this.keyToAction.set(config.key, action);  
  }  
});
```

Issue: If customizations create circular references, this could cause infinite loops.

8. Menu State Corruption (`index.html:485-495`)

javascript

```
toggleMenu(menuName, button) {  
  const menu = document.getElementById(`${menuName}-menu`);  
  const isCurrentlyActive = this.activeMenu === menuName;  
  this.closeAllMenus();  
  if (!isCurrentlyActive) {  
    menu.classList.add('show');  
    // ...  
  }  
}
```

Issue: If `menu` is null, calling `classList.add` will throw an error.

9. File Download Security Issue (`src/export/handlers.js:295-310`)

javascript

```
function sanitizeFilename(raw, fallback = 'document.txt') {  
  // ... sanitization logic  
  if (!/^[a-z0-9]{1,5}$/i.test(cleaned)) return cleaned + '.txt';  
  return cleaned;  
}
```

Issue: Regex allows dangerous extensions like `.exe.txt` to pass through.

10. User Activity Log Memory Leak (`src/ui/logging.js:25-40`)

javascript

```
function logEntry(message, color = '#000') {  
  const entry = document.createElement('div');  
  // ... create entry  
  logContainer.appendChild(entry);  
}
```

****Issue**:** No cleanup mechanism - log entries accumulate indefinitely.

Performance Issues

11. Inefficient Search Implementation (`src/export/handlers.js:430-450`)

javascript

```
setTimeout(() => {  
  try {  
    const content = view.state.doc.toString();  
    const results = search_document(content, query, false);  
    // ...  
  } catch (e) {  
    // ...  
  } finally {  
    searchInput.disabled = false;  
  }  
}, 10);
```

****Issue**:** Converting entire document to string on every search is inefficient for large documents.

12. Redundant Event Listeners ([src/export/handlers.js:700-720](#))

javascript

```
((() => {  
  if (window.__shortcutsInstalled) return;  
  window.__shortcutsInstalled = true;  
  // ... setup listeners  
})();
```

****Issue**:** Global flag pattern is fragile and can lead to duplicate listeners if flag is cleared elsewhere.

Configuration/Setup Issues

13. Hardcoded WebSocket URL ([src/setup/yjsSetup.js:23](#))

javascript

```
const provider = new WebSocketProvider('ws://localhost:1234', room, ydoc);
```

****Issue**:** Hardcoded localhost URL won't work in production environments.

14. Missing WASM Error Handling ([rust-wasm/src/lib.rs:380-390](#))

rust

```
fn encode_with_grid_tag(message: &PromiseGridMessage) -> Result<Vec<u8>, serde_cbor::Error> {  
    let untagged_bytes = serde_cbor::to_vec(message)?;  
    // ... manual tag addition  
    Ok(tagged_bytes)  
}
```

****Issue**:** Manual CBOR tag construction is error-prone and may produce invalid CBOR.

15. Inconsistent Error Handling in GitHub Service (`src/github/githubService.js:160-170`)

javascript

```
if (!response.ok) {  
    const errorData = await response.json();  
    throw new Error(`GitHub API error: ${errorData.message || 'Unknown error'}`);  
}
```

****Issue**:** Assumes error response is always JSON - will throw if response is not valid JSON.

TECHNICAL DEBT

Architecture Issues

1. **Circular Dependencies** - Several modules have circular import dependencies
2. **Global State Pollution** - Extensive use of `window` object for state management
3. **Mixed Async Patterns** - Inconsistent use of async/await vs Promises vs callbacks
4. **Tight Coupling** - UI components tightly coupled to specific DOM structure
5. **No State Management** - Lack of centralized state management system

Code Quality Issues

1. **Inconsistent Error Handling** - Mix of throw, console.error, and silent failures
2. **Missing Type Safety** - No TypeScript usage despite complex data structures
3. **Large Files** - Several files exceed 500+ lines (handlers.js, index.html)
4. **Hardcoded Values** - Magic numbers and strings throughout codebase
5. **No Unit Tests** - Only E2E tests, missing unit test coverage

Security Concerns

1. **Input Validation** - Insufficient validation of user inputs in multiple places
2. **CORS Configuration** - No explicit CORS configuration for production
3. **Content Security Policy** - Missing CSP headers
4. **Dependency Security** - No security audit of npm dependencies

MAINTENANCE ISSUES

Build System

1. **Complex Makefile** - Makefile has become overly complex with many targets
2. **Missing CI/CD** - No automated testing/deployment pipeline
3. **Version Management** - No proper versioning strategy for WASM module
4. **Asset Optimization** - No optimization for production builds

Monitoring/Logging

1. **Excessive Console Logging** - Debug logs left in production code
2. **No Error Tracking** - No integration with error tracking services
3. **Performance Monitoring** - No performance metrics collection
4. **User Analytics** - No usage analytics or monitoring

COMPATIBILITY ISSUES

Browser Support

1. **WebAssembly Requirements** - No fallback for browsers without WASM support
2. **Clipboard API** - Fallback implementation is deprecated
3. **WebSocket Support** - No graceful degradation for WebSocket failures
4. **Modern JavaScript Features** - Uses features not supported in older browsers





Mobile Support

1. **Touch Events** - Limited touch event handling for mobile devices
2. **Virtual Keyboard** - No handling of virtual keyboard on mobile
3. **Responsive Design** - Limited mobile optimization in CSS
4. **PWA Features** - No Progressive Web App capabilities

QUESTIONS FOR CLARIFICATION

1. **PromiseGrid Protocol:** Is the current CBOR tag implementation (0x67726964) officially sanctioned by the PromiseGrid protocol specification?
 2. **GitHub Integration:** Should the GitHub integration support organization repositories or only personal repositories?
 3. **WASM Module:** Are there plans to replace the current WASM build system with a more standardized approach?
 4. **Real-time Sync:** Should the editor work without the WebSocket server for local-only editing?
 5. **Data Persistence:** Is IndexedDB the intended long-term storage solution, or should we implement server-side persistence?
 6. **User Authentication:** Are there plans to implement proper user authentication beyond local usernames?
 7. **Collaboration Limits:** Is there an intended maximum number of concurrent users per document?
 8. **Export Formats:** Are there plans to support additional export formats beyond the current set?
-

Priority Levels:

-  **Critical:** Security vulnerabilities, data loss risks, system crashes
-  **High:** Functional bugs, performance issues, user experience problems
-  **Medium:** Technical debt, maintenance issues, compatibility problems
-  **Low:** Documentation, minor improvements, nice-to-have features