Plaquita.com Design Editor

Improvements & Missing Features Report

Document Version: 1.0
Date: 2024
Project: Custom T-Shirt Design Editor

Executive Summary

This document outlines the key improvements needed and missing features for the Plaquita.com Custom T-Shirt Design Editor. The recommendations are organized by priority and impact to help guide development efforts and resource allocation.

Table of Contents

Critical Improvements

Security Issues

Priority: CRITICAL | Impact: CRITICAL | Effort: LOW

- **API Keys Exposed in Source Code**
  - WooCommerce API keys are currently hardcoded in the source code
  - This is a critical security vulnerability
  - Keys should be moved to environment variables immediately
  - Add environment variable validation
  - Create secure configuration management
- **Input Validation Missing**
  - No validation for file uploads (type, size)
  - User input not sanitized properly
  - API responses not validated
  - Need rate limiting for API calls
- **XSS Prevention**
  - SVG content not sanitized before rendering
  - Need Content Security Policy headers
  - Validate all external content sources

Architecture Issues

Priority: CRITICAL | Impact: HIGH | Effort: MEDIUM

- **Monolithic Component Structure**
  - Main DesignEditor component is approximately 4,700 lines
  - Contains UI rendering, canvas logic, business logic, and state management
  - Violates Single Responsibility Principle
  - Makes code difficult to test, maintain, and extend
  - Needs to be broken down into smaller, focused components
- **Excessive Prop Drilling**
  - SideMenu component receives 30+ props
  - State management scattered across multiple useState hooks
  - Makes components tightly coupled and fragile
  - Difficult to add new features without breaking existing code

High Priority Improvements

State Management

Priority: HIGH | Impact: HIGH | Effort: MEDIUM

- **Implement Context API**
  - Replace prop drilling with React Context
  - Centralize design-related state management

  - Improve component reusability
  - Reduce coupling between components

- **Use Reducer Pattern**
  - Implement useReducer for complex state logic
  - Better state management for undo/redo functionality
  - More predictable state updates
  - Easier to debug state changes

Performance Issues

Priority: HIGH | Impact: HIGH | Effort: MEDIUM

- **Canvas Rendering Performance**
  - Heavy re-renders on every state change
  - Canvas redraws on every keystroke
  - No debouncing for text input updates
  - Multiple canvas instances initialized unnecessarily
  - Need to optimize rendering cycles

- **Memory Leaks**
  - Event listeners not properly cleaned up
  - Multiple canvas instances may cause memory issues
  - Large images kept in memory unnecessarily
  - Need proper cleanup on component unmount

Error Handling

Priority: HIGH | Impact: MEDIUM | Effort: LOW

- **Missing Error Boundaries**
  - No error boundaries to catch React errors
  - Application crashes affect entire user experience
  - Need graceful error handling and recovery
  - User-friendly error messages required

- **API Error Handling**
  - No comprehensive error handling for API calls
  - Network errors not handled gracefully
  - Server errors not communicated to users
  - Need retry mechanisms for failed requests

Medium Priority Improvements

Code Organization

Priority: MEDIUM | Impact: MEDIUM | Effort: MEDIUM

- **File Structure**
  - Flat component structure
  - No clear separation of concerns
  - Utilities mixed with components
  - Need organized folder structure with clear responsibilities

- **Code Duplication**
  - Similar functions with different names (handleObjectScaling vs handleObjectScaling1)
  - Repeated canvas setup code for different views
  - Similar event handlers across multiple components
  - Need to extract common logic into reusable utilities

Styling Consistency

Priority: MEDIUM | Impact: LOW | Effort: MEDIUM

- **Mixed Styling Approaches**
  - Combination of inline styles, styled-components, and Tailwind CSS
  - Inconsistent spacing and colors throughout application
  - Need to standardize on one primary styling approach
  - Create design system with consistent tokens

Component Structure

Priority: MEDIUM | Impact: MEDIUM | Effort: MEDIUM

- **Extract Custom Hooks**
  - Canvas initialization and lifecycle management
  - Object manipulation logic
  - Text editing functionality
  - Image processing operations
  - Clip art management

- **Split UI Components**
  - Separate canvas wrapper from logic
  - Extract toolbar components
  - Create dedicated property panels
  - Organize dialog components

Missing Features

Core Functionality

Priority: HIGH | Impact: HIGH | Effort: MEDIUM

- **Undo/Redo System**
  - Currently not implemented
  - Essential for user experience
  - Users expect this in any design tool
  - Need command pattern implementation
  - History management required
- **Keyboard Shortcuts**
  - No keyboard shortcuts available
  - Standard shortcuts expected (Ctrl+Z, Delete, Copy, Paste)
  - Improve workflow efficiency
  - Need comprehensive shortcut system

Design Features

Priority: MEDIUM | Impact: HIGH | Effort: MEDIUM

- **Design Templates Library**
  - No pre-designed templates available
  - Users must start from scratch
  - Templates would improve user onboarding
  - Need template categories (sports, business, casual)
  - Allow saving custom templates
- **Advanced Text Features**
  - Limited text effects available
  - No text shadows or glows
  - Missing 3D text effects
  - No text alignment options beyond basic
  - Limited text shapes
  - Need text along path feature
- **Image Editing Tools**
  - No crop functionality
  - No resize tools
  - Missing filters and effects
  - No image adjustments (brightness, contrast, saturation)
  - Need comprehensive image editing suite

Export & Save Features

Priority: MEDIUM | Impact: MEDIUM | Effort: MEDIUM

- **Export Options**
  - Limited export formats
  - No high-resolution export option
  - Missing print-ready formats
  - No batch export capability
  - Need multiple format support (PNG, SVG, PDF, JPG)
- **Design History & Saving**
  - No local save functionality
  - No cloud save option
  - Missing design library/gallery
  - No recent designs feature
  - Need persistent storage solution

Collaboration Features

Priority: LOW | Impact: LOW | Effort: HIGH

- **Sharing Capabilities**
  - Cannot share designs
  - No collaboration features
  - Missing comments and annotations
  - No version history
  - Need sharing via link functionality

Advanced Tools

Priority: LOW | Impact: LOW | Effort: MEDIUM

- **Advanced Color Tools**
  - Basic color picker only
  - No eyedropper tool
  - Missing color palette generator
  - No color harmony tools
  - Limited custom color swatches
- **Alignment & Distribution Tools**
  - No snap-to-grid option
  - Missing alignment guides
  - No object distribution tools
  - Need smart guides when dragging

User Experience Enhancements

Loading & Feedback

Priority: MEDIUM | Impact: MEDIUM | Effort: LOW

- **Loading States**
  - No loading indicators during image upload
  - Missing progress bars for bulk operations
  - No feedback during async operations
  - Need skeleton loaders for content
  - Disable buttons during processing
- **User Feedback**
  - No success notifications
  - Missing error messages
  - No toast notifications
  - Need clear action confirmations

Mobile Experience

Priority: HIGH | Impact: HIGH | Effort: MEDIUM

- **Responsive Design**
  - Mobile experience needs significant improvement
  - Sidebar behavior problematic on mobile
  - Canvas scaling issues on small screens
  - Need touch gesture support (pinch-to-zoom, pan)
  - Require mobile-optimized toolbar
- **Touch Interactions**
  - Limited touch support
  - No gesture recognition
  - Need touch-friendly controls
  - Require adaptive UI for mobile devices

Drag & Drop

Priority: MEDIUM | Impact: MEDIUM | Effort: LOW

- **Visual Feedback**
  - No visual feedback during drag operations
  - Missing snap indicators
  - Need alignment guides
  - Require duplicate on drag option

Performance Improvements

Optimization Needs

Priority: HIGH | Impact: HIGH | Effort: MEDIUM

- **Canvas Performance**
  - Debounce text input updates
  - Optimize canvas rendering cycles
  - Use requestAnimationFrame for smooth updates
  - Implement object caching for static elements
  - Lazy load canvas instances
- **Component Optimization**
  - Memoize expensive components
  - Optimize re-renders
  - Use React.memo for pure components
  - Memoize callbacks and computed values

- **Asset Optimization**
  - Compress SVG cliparts
  - Optimize PNG assets
  - Implement lazy loading for cliparts
  - Code splitting for routes
  - Optimize font loading

Memory Management

Priority: HIGH | Impact: MEDIUM | Effort: MEDIUM

- **Cleanup Required**
  - Properly dispose canvas on unmount
  - Clean up event listeners
  - Release image memory
  - Manage multiple canvas instances efficiently

Security Improvements

Immediate Actions Required

Priority: CRITICAL | Impact: CRITICAL | Effort: LOW

- **API Key Security**
  - Move all API keys to environment variables
  - Never commit keys to version control
  - Implement key rotation policy
  - Add key validation
- **Input Sanitization**
  - Validate all file uploads
  - Sanitize user input
  - Validate API responses
  - Implement rate limiting
- **Content Security**
  - Sanitize SVG content
  - Implement CSP headers
  - Validate external content
  - Secure image processing

Quality & Testing

Testing Infrastructure

Priority: MEDIUM | Impact: MEDIUM | Effort: MEDIUM

- **Unit Tests**
  - No unit tests for utility functions
  - Missing tests for custom hooks
  - Need comprehensive test coverage
  - Test color extraction functions
  - Test boundary constraint logic
- **Component Tests**
  - No component tests
  - Missing integration tests
  - Need React Testing Library setup
  - Test user interactions
  - Verify component rendering
- **Visual Testing**
  - No visual regression tests
  - Missing screenshot testing
  - Need canvas output verification
  - Compare design exports

Code Quality Tools

Priority: MEDIUM | Impact: MEDIUM | Effort: LOW

- **Linting & Formatting**
  - No ESLint configuration
  - Missing Prettier setup
  - Need code style enforcement
  - Require React hooks linting
  - Add pre-commit hooks
- **Type Safety**

  - No TypeScript implementation
  - Missing type definitions
  - Need gradual migration plan
  - Add PropTypes as interim solution

Accessibility Improvements

WCAG Compliance

Priority: MEDIUM | Impact: MEDIUM | Effort: MEDIUM

- **Keyboard Navigation**
  - Not all elements keyboard accessible
  - Missing focus management
  - Need skip links
  - Require logical tab order
- **Screen Reader Support**
  - Missing ARIA labels
  - No canvas announcements
  - Need semantic HTML
  - Require alternative text for images
- **Visual Accessibility**
  - Color contrast issues
  - Need WCAG AA compliance
  - Require high contrast mode
  - Missing focus indicators

Documentation Needs

Technical Documentation

Priority: MEDIUM | Impact: LOW | Effort: MEDIUM

- **Code Documentation**
  - Missing JSDoc comments
  - No function documentation
  - Need component prop documentation
  - Require usage examples
- **API Documentation**
  - No API endpoint documentation
  - Missing request/response examples
  - Need error code documentation
  - Require integration guides

User Documentation

Priority: LOW | Impact: LOW | Effort: MEDIUM

- **User Guides**
  - No user manual
  - Missing tutorials
  - Need FAQ section
  - Require video tutorials
- **Help System**
  - No in-app help
  - Missing tooltips
  - Need contextual help
  - Require keyboard shortcuts reference

Implementation Priority

Phase 1: Critical Fixes (Immediate)

1. Move API keys to environment variables
2. Add error boundaries
3. Fix memory leaks
4. Implement basic error handling
5. Add input validation

Phase 2: Core Features (High Priority)

1. Implement undo/redo system
2. Add keyboard shortcuts
3. Improve mobile responsiveness
4. Add loading states and feedback

5. Implement Context API for state management

Phase 3: Architecture (Medium Priority)

1. Refactor monolithic component
2. Extract custom hooks
3. Reorganize file structure
4. Reduce code duplication
5. Standardize styling approach

Phase 4: Enhancements (Lower Priority)

1. Add design templates
2. Implement advanced text features
3. Add image editing tools
4. Improve export options
5. Add collaboration features

Success Metrics

Performance Metrics

- Time to Interactive (TTI)
- First Contentful Paint (FCP)
- Largest Contentful Paint (LCP)
- Canvas render time
- Bundle size

User Experience Metrics

- Task completion rate
- Time to create first design
- Error rate
- User satisfaction score
- Feature adoption rate

Code Quality Metrics

- Test coverage percentage
- Code duplication percentage
- Technical debt ratio
- Bug resolution time

Conclusion

This document outlines the comprehensive improvements and missing features needed for the Plaquita.com Design Editor. The recommendations are prioritized to help guide development efforts and ensure critical issues are addressed first.

Key focus areas:

- **Security**: Immediate action required for API key management
- **Architecture**: Refactoring needed for maintainability
- **Features**: Core functionality like undo/redo is essential
- **Performance**: Optimization needed for better user experience
- **Quality**: Testing and documentation improvements required

Regular reviews and updates to this document will help track progress and ensure all improvements are addressed systematically.

Document Prepared For: Development Team
Review Frequency: Quarterly
Next Review Date: [To be determined]