Software Requirements Specification (SRS)

Premium Zero-Track QR Code Generator

Version: 1.0
Date: February 15, 2026
Author: Portfolio Project
Status: Draft

---

1. Introduction

1.1 Purpose

This Software Requirements Specification (SRS) document describes the functional and non-functional requirements for a premium, privacy-focused QR code generator web application. The system is designed as a portfolio showcase that demonstrates advanced frontend engineering while solving real-world privacy concerns in existing QR code generators[1].

The intended audience includes:

- Development team (implementation reference)
- Stakeholders and portfolio reviewers
- UX/UI designers
- QA testers

1.2 Scope

Product Name: Premium Zero-Track QR Generator

Product Overview:
A client-side web application that generates QR codes without server-side data collection or user tracking. The system emphasizes

premium user experience, real-time customization, and transparent privacy practices.

Key Benefits:

- Complete client-side processing (zero server tracking)[2]
- Premium UI/UX with advanced animations and interactions
- Real-world simulation for QR code testing
- Professional-grade customization options
- High-performance, accessible, and responsive design

Out of Scope:

- Dynamic QR codes with URL redirection services
- User authentication or account management
- Server-side analytics or tracking
- Mobile native applications (web-only)
- Batch QR code generation for enterprise

1.3 Definitions, Acronyms, and Abbreviations

Term	Definition
QR Code	Quick Response Code - 2D barcode readable by cameras
Client-Side	Processing that occurs in the user's browser
ECL	Error Correction Level (L, M, Q, H)
SVG	Scalable Vector Graphics
PWA	Progressive Web Application
DPI	Dots Per Inch
Glassmorphism	UI design trend using translucent elements

1.4 References

[1] LinkedIn. (2025). Free QR Code Generator Risks: How a $500 Mistake Exposed Hidden Costs. https://www.linkedin.com/pulse/free-qr-code-generator-risks-how-500-mistake-exposed-hidden-garg-aj7mc

[2] Lucid Privacy. (2025). QR Codes: Useful Tool, or Privacy Disaster? https://blog.lucidprivacy.io/qr-codes-useful-tool-or-privacy-disaster/

[3] Perforce. (2025). How to Write a Software Requirements Specification. https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document

1.5 Overview

This SRS is organized into the following sections:

- **Section 2:** Overall product description, context, and constraints
- **Section 3:** Detailed functional requirements
- **Section 4:** Non-functional requirements (performance, security, usability)
- **Section 5:** System interfaces and dependencies
- **Section 6:** Verification and validation approach

---

2. Overall Description

2.1 Product Perspective

The Premium Zero-Track QR Generator is a standalone web application that operates entirely within the user's browser. Unlike traditional QR generators that proxy scan data through tracking servers[2], this system performs all operations client-side using JavaScript libraries.

System Context:

- Runs in modern web browsers (Chrome, Firefox, Safari, Edge)
- No backend server required for core functionality
- Deployed as static site on edge CDN (Vercel/Netlify)
- Optional PWA installation for offline usage

2.2 Product Functions

Core Functions:

1. **QR Code Generation**
    - Generate QR codes from URLs, text, WiFi credentials, contact cards
    - Real-time preview with live updates
    - Multiple error correction levels
2. **Visual Customization**
    - Color picker with gradient support
    - Shape morphing (rounded corners, module styles)
    - Logo/image embedding with automatic optimization
    - Custom eye frame designs
3. **Privacy Dashboard**
    - Visual audit showing zero data collection
    - Comparison with tracking-enabled generators
    - Educational tooltips about privacy risks
4. **Real-World Simulator**
    - Mockup templates (business cards, posters, packaging)
    - Distance/lighting/angle simulation
    - Scan success rate estimation
    - Camera shake effect
5. **Export Suite**
    - PNG export (multiple DPI: 72, 150, 300, 600)
    - SVG export (vector, print-ready)
    - Color variant batch export
    - Dark mode optimized versions

2.3 User Classes and Characteristics

User Class	Characteristics	Technical Expertise
Designers	Need high-quality exports, branding control	Medium
Small Business	Menu/poster QR codes, ease of use	Low
Marketers	Campaign codes, mockup previews	Medium
Developers	API integration, SVG manipulation	High
Privacy-Conscious	Zero-tracking requirement	Medium-High

Table 1: User class characteristics

2.4 Operating Environment

Client Requirements:

- Modern browser with ES6+ JavaScript support
- Minimum screen resolution: 375px width (mobile)
- Canvas API support
- Local storage support (5MB minimum)

Recommended Environment:

- Desktop: 1920x1080 or higher
- Mobile: iOS 14+ / Android 10+
- Browser: Latest 2 versions of Chrome, Firefox, Safari, Edge
- Network: Optional (offline-capable after first load)

2.5 Design and Implementation Constraints

Technical Constraints:

- Must operate entirely client-side (no backend server for QR generation)
- QR encoding limited by browser memory (~10KB data per code)
- Logo size limited to prevent error correction failure
- Canvas rendering performance on low-end mobile devices

Design Constraints:

- Maintain WCAG 2.1 AA accessibility standards
- Support high-contrast mode
- Animations must respect `prefers-reduced-motion`
- File sizes: Initial bundle <500KB, lazy-loaded assets <200KB each

Regulatory Constraints:

- GDPR compliance (no data collection, but must document)
- Accessibility laws (WCAG 2.1 AA minimum)

2.6 Assumptions and Dependencies

Assumptions:

- Users have JavaScript enabled
- Users grant camera permission for simulator feature
- Users understand basic QR code concepts
- Modern browser with HTML5 Canvas support

Dependencies:

- `qrcode` or `qrcode.react` npm library for encoding
- React 18+ or Next.js 14+ framework
- Tailwind CSS or similar utility framework
- Framer Motion or GSAP for animations
- Vercel/Netlify for deployment

---

3. Functional Requirements

3.1 QR Code Generation Module

FR-1.1: URL Input and Validation

Priority: High
Description: The system shall accept URL input and validate format before encoding.

Acceptance Criteria:

- System validates URL format using regex or URL API

- System displays error message for invalid URLs within 500ms
- System supports http, https, ftp protocols
- System allows up to 2048 characters for URL input

FR-1.2: Text Input Support

Priority: High
Description: The system shall accept plain text input up to 4,296 alphanumeric characters (QR version 40, Level L).

Acceptance Criteria:

- System provides multi-line textarea for text input
- System displays character count in real-time
- System warns when approaching character limit
- System prevents encoding beyond QR capacity

FR-1.3: WiFi Configuration

Priority: Medium
Description: The system shall generate WiFi configuration QR codes with SSID, password, and security type.

Acceptance Criteria:

- System provides form fields for SSID, password, encryption type (WEP/WPA/WPA2/None)
- System generates proper WIFI: format string
- System validates required fields before generation
- System masks password input with toggle visibility option

FR-1.4: Contact Card (vCard)

Priority: Medium
Description: The system shall generate vCard QR codes with name, phone, email, and address fields.

Acceptance Criteria:

- System provides form for name, organization, phone, email, URL, address
- System validates email and phone formats
- System generates vCard 3.0 format string

- System allows optional profile photo upload (<100KB)

FR-1.5: Real-Time Preview

Priority: High
Description: The system shall display QR code preview that updates within 100ms of input changes.

Acceptance Criteria:

- QR preview renders on canvas element
- Preview updates debounced at 100ms intervals
- Preview maintains aspect ratio
- Preview displays "Empty" state when no input provided

FR-1.6: Error Correction Level Selection

Priority: Medium
Description: The system shall allow users to select error correction level (L, M, Q, H).

Acceptance Criteria:

- System provides radio buttons or dropdown for ECL selection
- System explains each level: L (7%), M (15%), Q (25%), H (30%)
- System defaults to M (15%) for balance
- System warns when logo size requires higher ECL

3.2 Customization Module

FR-2.1: Color Picker - Foreground

Priority: High
Description: The system shall allow users to customize QR module color with hex, RGB, or gradient.

Acceptance Criteria:

- System provides color picker UI (React Colorful or similar)
- System supports hex input (#000000 format)
- System supports RGB/RGBA input
- System provides gradient editor for linear gradients
- System applies color changes to preview within 50ms

FR-2.2: Color Picker - Background

Priority: High
Description: The system shall allow users to customize background color independently.

Acceptance Criteria:

- Separate color picker for background
- Background color supports transparency (alpha channel)
- System warns when contrast ratio falls below 3:1
- System provides "transparent" quick option

FR-2.3: Shape Morphing - Modules

Priority: Medium
Description: The system shall allow users to change QR module shape (square, rounded, dots, circles).

Acceptance Criteria:

- System provides preset shape options with visual examples
- System applies SVG filters or canvas rendering for shapes
- System maintains scanability (warns if too stylized)
- Shape changes animate smoothly (300ms transition)

FR-2.4: Shape Morphing - Eye Frames

Priority: Medium
Description: The system shall allow users to customize eye frame (corner markers) design.

Acceptance Criteria:

- System provides eye frame presets (square, rounded, leaf, circle)
- System allows independent customization of 3 eyes
- Eye frames maintain required size ratios
- Preview updates in real-time

FR-2.5: Logo Upload and Embedding

Priority: High
Description: The system shall allow users to upload and embed logo in QR center.

Acceptance Criteria:

- System accepts PNG, JPG, SVG formats (<5MB)
- System auto-scales logo to safe size (max 30% of QR area)
- System applies white background padding around logo
- System warns if logo coverage requires ECL-H
- System provides logo positioning controls (center, custom offset)

FR-2.6: Size and Margin Control

Priority: Medium
Description: The system shall allow users to adjust QR output size and quiet zone margin.

Acceptance Criteria:

- Slider for size: 128px to 2048px
- Slider for margin: 0 to 10 modules
- Default size: 512px, margin: 4 modules
- Real-time preview reflects size changes (scaled to fit canvas)

3.3 Privacy Dashboard Module

FR-3.1: Privacy Audit Visualization

Priority: High
Description: The system shall display visual flowchart showing client-side processing only.

Acceptance Criteria:

- Animated flowchart: Input → Browser → Download
- Green checkmarks for "No server upload", "No IP logging", "No tracking"
- Red X's for comparison: "Typical generator: Server upload, IP logged, Analytics enabled"
- Tooltips explain each privacy checkpoint

FR-3.2: Data Transparency Statement

Priority: High
Description: The system shall display clear privacy statement on main page.

Acceptance Criteria:

- Statement visible above-the-fold or in header
- Text: "100% Client-Side. Your data never leaves your browser."
- Link to detailed privacy explanation page
- Privacy badge icon (shield or lock)

FR-3.3: Educational Tooltips

Priority: Low
Description: The system shall provide educational content about QR privacy risks.

Acceptance Criteria:

- Info icons next to key privacy terms
- Tooltips appear on hover/tap
- Content explains: dynamic vs static codes, tracking mechanisms, link hijacking risks
- Links to external privacy resources

3.4 Real-World Simulator Module

FR-4.1: Mockup Templates

Priority: Medium
Description: The system shall provide realistic mockup templates for QR placement preview.

Acceptance Criteria:

- Templates: business card, poster, bottle label, menu, t-shirt
- QR overlay drag-and-drop positioning
- Scale slider for QR size relative to mockup
- Rotation control (-45° to +45°)

FR-4.2: Environmental Simulation

Priority: Medium
Description: The system shall simulate scan conditions (lighting, distance, angle).

Acceptance Criteria:

- Lighting slider: dim (30%) to bright (100%)
- Distance slider: close (simulated blur) to far (simulated pixelation)
- Angle slider: 0° (straight) to 60° (tilted perspective)
- Camera shake toggle (subtle motion blur effect)

FR-4.3: Scan Success Rate Estimation

Priority: Low
Description: The system shall estimate scan success rate based on design choices.

Acceptance Criteria:

- Algorithm considers: contrast ratio, logo coverage, ECL, module size
- Display percentage: 0-100% with color coding (red <70%, yellow 70-90%, green >90%)
- Explanation tooltip for low scores
- Recommendations to improve score

3.5 Export Module

FR-5.1: PNG Export - Multiple Resolutions

Priority: High
Description: The system shall export QR as PNG at standard DPI presets.

Acceptance Criteria:

- Export options: 72 DPI (web), 150 DPI (draft print), 300 DPI (print), 600 DPI (high-res print)
- File naming: qr-code-[type]-[size]-[dpi].png
- Download triggered via browser download API

- File size optimization without quality loss

FR-5.2: SVG Export

Priority: High
Description: The system shall export QR as clean SVG for vector editing.

Acceptance Criteria:

- SVG contains proper XML structure
- Paths optimized (minimal nodes)
- Embedded styles or inline attributes (user preference)
- Supports custom colors, shapes, logo embedding
- File naming: qr-code-[type].svg

FR-5.3: Batch Export - Color Variants

Priority: Low
Description: The system shall export multiple color variations in one action.

Acceptance Criteria:

- User defines 2-5 color schemes
- System generates and zips all variants
- Zip file download: qr-code-variants.zip
- Each variant named by color scheme

FR-5.4: Dark Mode Optimization

Priority: Medium
Description: The system shall provide dark-mode optimized exports.

Acceptance Criteria:

- Toggle for "Export for dark background"
- Inverts colors appropriately (light QR on dark background)
- Adds subtle glow/shadow for contrast
- Preview shows both light and dark versions

3.6 Progressive Web App (PWA) Features

FR-6.1: Offline Capability

Priority: Medium
Description: The system shall function offline after initial load.

Acceptance Criteria:

- Service worker caches app shell and assets
- QR generation works without network
- Export functions work offline
- Offline indicator displayed when network unavailable

FR-6.2: Install Prompt

Priority: Low
Description: The system shall prompt users to install as PWA.

Acceptance Criteria:

- Install banner appears after 2 visits or 3 minutes usage
- User can dismiss permanently
- Install adds app to home screen/desktop
- Installed app runs in standalone mode

---

4. Non-Functional Requirements

4.1 Performance Requirements

NFR-1.1: Initial Load Time

Requirement: The application shall load and become interactive within 2 seconds on 4G connection.

Metric: Lighthouse Performance score ≥90

Acceptance Criteria:

- First Contentful Paint (FCP) <1.2s
- Time to Interactive (TTI) <2.0s
- Total Blocking Time (TBT) <200ms
- Initial bundle size <500KB gzipped

NFR-1.2: QR Generation Speed

Requirement: QR code generation shall complete within 100ms for inputs up to 1KB.

Acceptance Criteria:

- Encoding + rendering combined <100ms (measured in browser DevTools)
- 95th percentile <150ms
- Progress indicator shown if >200ms

NFR-1.3: Animation Frame Rate

Requirement: All animations shall maintain 60 FPS on desktop and 30 FPS minimum on mobile.

Acceptance Criteria:

- Measured via Chrome DevTools Performance panel
- No dropped frames during color picker interactions
- Smooth transitions on shape morphing

NFR-1.4: Memory Usage

Requirement: The application shall consume less than 100MB memory during typical usage.

Acceptance Criteria:

- Measured via Chrome Task Manager
- No memory leaks over 10-minute session
- Garbage collection does not cause UI freezes

4.2 Security Requirements

NFR-2.1: Client-Side Only Processing

Requirement: All QR generation and customization shall occur in the browser without server transmission.

Acceptance Criteria:

- Network tab shows zero POST/PUT requests for QR operations

- No API calls to external services during generation
- Privacy audit passes (no third-party tracking scripts)

NFR-2.2: Input Sanitization

Requirement: User inputs shall be sanitized to prevent XSS attacks.

Acceptance Criteria:

- URL inputs validated with URL API
- Text inputs escaped before encoding
- SVG uploads scanned for malicious scripts
- Content Security Policy (CSP) headers enforced

NFR-2.3: Secure External Resources

Requirement: All external resources (fonts, libraries) shall load over HTTPS.

Acceptance Criteria:

- No mixed content warnings
- Subresource Integrity (SRI) hashes for CDN resources
- Fonts loaded from trusted CDNs or self-hosted

4.3 Usability Requirements

NFR-3.1: Accessibility - WCAG 2.1 AA

Requirement: The application shall meet WCAG 2.1 Level AA standards.

Acceptance Criteria:

- Color contrast ratio ≥4.5:1 for normal text, ≥3:1 for large text
- All interactive elements keyboard accessible (tab navigation)
- Screen reader compatible (ARIA labels, semantic HTML)
- Focus indicators visible and high-contrast
- Axe DevTools audit passes with zero violations

NFR-3.2: Responsive Design

Requirement: The application shall provide optimal experience on all device sizes.

Acceptance Criteria:

- Mobile (375px - 767px): Single column, touch-optimized controls
- Tablet (768px - 1023px): Two-column layout, balanced preview
- Desktop (1024px+): Multi-column layout, side-by-side preview
- No horizontal scroll on any breakpoint

NFR-3.3: Reduced Motion Support

Requirement: The application shall respect user's motion preferences.

Acceptance Criteria:

- Detects `prefers-reduced-motion` media query
- Disables/simplifies animations when enabled
- Instant transitions replace animated ones
- Core functionality unaffected

NFR-3.4: Error Handling and Feedback

Requirement: The system shall provide clear, helpful error messages.

Acceptance Criteria:

- Errors displayed inline near relevant field
- Error messages explain problem and suggest solution
- No generic "Error 500" messages
- Success confirmations for exports (toast notification)

4.4 Reliability and Availability

NFR-4.1: Uptime

Requirement: The application shall maintain 99.9% uptime (measured via CDN availability).

Acceptance Criteria:

- Hosted on Vercel/Netlify with SLA guarantee
- Edge CDN ensures global availability
- Downtime monitored via UptimeRobot or similar

NFR-4.2: Browser Compatibility

Requirement: The application shall function correctly on latest 2 versions of major browsers.

Acceptance Criteria:

- Chrome 100+
- Firefox 100+
- Safari 15+
- Edge 100+
- Tested via BrowserStack or similar

4.5 Maintainability and Scalability

NFR-5.1: Code Quality

Requirement: Codebase shall maintain high quality and consistency.

Acceptance Criteria:

- ESLint passing with zero errors (warnings acceptable)
- Prettier formatting enforced
- TypeScript strict mode enabled
- Unit test coverage ≥70% for utility functions

NFR-5.2: Component Architecture

Requirement: UI shall be built with reusable, modular components.

Acceptance Criteria:

- Component library (Shadcn UI or similar)
- Atomic design structure (atoms, molecules, organisms)
- Storybook documentation for components
- Props documented with TypeScript interfaces

NFR-5.3: Deployment Pipeline

Requirement: Automated CI/CD pipeline for testing and deployment.

Acceptance Criteria:

- GitHub Actions or similar CI tool
- Automated builds on pull requests
- Lighthouse CI checks performance regression
- One-click deployment to production

---

5. System Interfaces

5.1 User Interfaces

UI-1: Desktop Layout

- Header: Logo, navigation (Home, Privacy, Export)
- Left Panel: Input form and customization controls
- Center Panel: Live QR preview canvas (512x512px default)
- Right Panel: Privacy dashboard and simulator
- Footer: Credits, GitHub link, version

UI-2: Mobile Layout

- Stacked vertical layout
- Collapsible sections (Input, Customize, Preview, Export)
- Bottom navigation bar
- Preview canvas scales to screen width

5.2 Software Interfaces

SI-1: QR Encoding Library

Interface: `qrcode` npm package or equivalent
Functions:

- `QRCode.toCanvas(canvas, text, options)` - Render to canvas
- `QRCode.toString(text, options)` - Generate SVG string
- Options: errorCorrectionLevel, margin, color, width

SI-2: Browser APIs

Canvas API: Rendering and export
Download API: File downloads (`document.createElement('a').download`)
LocalStorage API: Settings persistence
Service Worker API: Offline capability

5.3 Communication Interfaces

CI-1: Static Asset Loading

Protocol: HTTPS
Purpose: Load fonts, images, icons from CDN
Data Format: WOFF2, PNG, SVG

---

6. Verification and Validation

6.1 Testing Approach

Unit Testing

- Test QR encoding functions with various inputs
- Test color manipulation utilities
- Test export functions (PNG, SVG generation)
- Framework: Jest + React Testing Library

Integration Testing

- Test end-to-end user flows (input → customize → export)
- Test PWA installation and offline mode
- Framework: Playwright or Cypress

Accessibility Testing

- Automated: Axe DevTools in CI pipeline
- Manual: Screen reader testing (NVDA, VoiceOver)
- Keyboard navigation testing

Performance Testing

- Lighthouse CI on every pull request
- WebPageTest for real-world metrics
- Memory profiling via Chrome DevTools

Cross-Browser Testing

- BrowserStack for automated testing
- Manual testing on physical devices

6.2 Acceptance Criteria

- All functional requirements (FR-*) implemented and tested
- Lighthouse Performance score ≥90
- Lighthouse Accessibility score 100
- Zero critical or high-severity bugs
- Documentation complete (README, inline comments)
- Portfolio showcase ready (screenshots, demo video)

---

7. Appendices

7.1 Glossary

Client-Side Processing: Computation performed in the user's web browser without sending data to a server.

Error Correction Level (ECL): QR code redundancy setting that determines how much damage the code can sustain while remaining scannable. Levels: L (7%), M (15%), Q (25%), H (30%).

Quiet Zone: The blank margin surrounding a QR code, required for reliable scanning (minimum 4 modules).

Module: The smallest square element in a QR code grid.

7.2 Technology Stack Summary

Layer	Technology
Framework	Next.js 14+ (React 18+)
Styling	Tailwind CSS + Shadcn UI
Animations	Framer Motion or GSAP
QR Library	qrcode npm package
State Management	React Context or Zustand
Deployment	Vercel or Netlify
Testing	Jest, React Testing Library, Playwright
Language	TypeScript (strict mode)

Table 2: Recommended technology stack

7.3 Revision History

Version	Date	Author	Changes
1.0	Feb 15, 2026	Portfolio Project	Initial draft

End of Document