Requirements Specification Document

Project: Secure URL Shortener Web Application

---

1. Introduction

1.1 Purpose

This Requirements Specification Document defines all functional and non-functional requirements for the Secure URL Shortener Application. It serves as a foundation for development, testing, and future enhancements. The document ensures a shared understanding between stakeholders, developers, and testers.

1.2 Scope

The application allows users to: - Shorten long URLs - Generate QR codes for URLs - Copy, download, and delete short URLs - Track analytics such as click count - Securely store user credentials in encrypted form - Access personalized dashboards - Maintain full security protections against malicious links, SQL injection, and XSS

1.3 Definitions and Acronyms

- **URL:** Uniform Resource Locator
- **Short URL:** A compressed redirection link
- **QR Code:** Quick Response code for URL sharing
- **Encryption:** Secured encoding algorithm
- **API:** Application Programming Interface
- **Authentication:** Login/Signup process
- **UI:** User Interface

---

2. Overall Description

2.1 Product Perspective

The system is a standalone responsive web application with: - A backend API service for URL operations - A frontend dashboard for user interaction - Integrated QR generation service - Secure authentication & user management

2.2 User Classes and Characteristics

User Type	Description
Guest User	Can only view home page, must sign up to use features
Registered User	Can shorten URLs, manage links, generate QR, view analytics
Admin (Optional)	Can view system analytics, user issues

2.3 Assumptions and Dependencies

- Users must have internet access
- QR generation library must be available
- Backend servers must support encryption (bcrypt/argon2)
- Redirect should be optimized for quick response

---

3. Functional Requirements

3.1 User Authentication Requirements

FR-01 Signup

- User must be able to create an account using email, password, and name.
- Duplicate emails must be rejected.

FR-02 Login

- Login must require valid email and password.
- Credentials must be encrypted during transmission.

FR-03 Password Encryption

- Passwords must be stored using hashed encryption (bcrypt/argon2).

FR-04 Token Authentication

- User session must be maintained using secure tokens.

3.2 URL Shortening Requirements

FR-05 Shorten URL

- User can input a long URL and receive a shortened version.
- System must validate URL structure.

FR-06 Malicious URL Detection

- System must identify and block phishing/malware URLs.

FR-07 Duplicate URL Handling

- Same URL shortened again should return existing short code.

FR-08 Expiry Management (Optional)

- URLs may have an expiration date.

3.3 QR Code Requirements

FR-09 Generate QR

- A QR code must be generated for any shortened URL.

FR-10 Download QR

- Users should be able to download QR in PNG format.

FR-11 QR Preview

- A preview must be displayed before download.

3.4 URL Management Requirements

FR-12 Copy URL

- User can copy the short URL to clipboard.

FR-13 Delete URL

- User can delete a previously created short URL.
- Deleting someone else's URL must not be allowed.

FR-14 Edit URL Alias (Optional)

- User may customize the short link code.

3.5 Analytics Requirements

FR-15 Click Tracking

- Each visit to a short URL must increase the click counter.

FR-16 Analytics Dashboard

- Users can view total clicks, top-performing links, and traffic pattern (basic).

FR-17 Real-time Refresh (Optional)

- Analytics should update automatically.

3.6 Dashboard Requirements

FR-18 Dashboard Overview

- Dashboard must show all user-created URLs in a clean UI.

FR-19 Sorting & Filtering

- Users should filter URLs by date, clicks, or name.

FR-20 Mobile Responsive UI

- Dashboard must work across devices.

---

4. Non-Functional Requirements

4.1 Performance Requirements

- System should generate short URLs in less than 1 second.
- Redirect must occur in under 300ms.
- System must handle at least 500 requests/minute.

4.2 Security Requirements

- All passwords must be hashed using bcrypt/argon2.
- SQL injection must be prevented.
- XSS attacks must be blocked on all inputs.
- Short URLs must never expose internal logic.
- Tokens must not appear in URL parameters.

4.3 Usability Requirements

- Application must be simple and intuitive.
- QR and copy buttons must provide feedback.
- Dashboard must maintain consistent layout.

4.4 Compatibility Requirements

- Support Chrome, Firefox, Edge, Safari.
- Mobile and tablet support mandatory.

4.5 Reliability Requirements

- System uptime must be 99% or above.
- Redirect should work even during heavy traffic.

4.6 Maintainability Requirements

- Code must follow modular structure.
- API should be version-controlled.

---

5. Future Enhancement Requirements

- Custom domain support
- Browser extensions
- Bulk URL upload
- Team analytics view
- Dark mode UI

---

6. Conclusion

This Requirement Specification Document provides a complete breakdown of functional and non-functional elements of the Secure URL Shortener Application. It ensures clarity for developers, QA testers, stakeholders, and future enhancements.