System Design Document

Project Name

URL Shortener Microservice

Type

Full-Stack Web Application

Tech Stack

Node.js, Express, React, Material UI, Winston

Version

1.0

Author

Srikanth Pandaraboina

Date

August 2025

Objective

To build a scalable, maintainable, and production-ready microservice for URL shortening with custom code support, analytics, expiry handling, and a modern frontend, following clean architectural and software engineering principles.

Architectural Overview

Frontend (React) <--> Backend (Express) <--> Data Store (In-memory/Redis)

Key Design Decisions

- Microservice architecture for clear separation of concerns
- React frontend for modern UX
- Node.js backend for async processing
- Winston for structured logging
- Stateless design (no authentication)

Data Modeling

Shortened URL Object:

{ shortcode, originalUrl, created, expiry, clicks, clickData: [timestamp, source, location] }

API Design

- POST /shorturls: Create short URL
- GET /shorturls/:shortcode: Redirect to original
- GET /shorturls/:shortcode/stats: Analytics
- GET /health: Health check

Technology Selections

- React + Material UI: Frontend
- Node.js + Express: Backend
- Winston: Logging
- In-memory storage (or Redis)

Scalability Considerations

- Use Redis or MongoDB for persistent storage
- Add Kafka or TimescaleDB for analytics
- Add containerization (Docker), deploy to cloud

Assumptions

- No auth required
- Alphanumeric shortcodes
- Single-node architecture

Future Enhancements

- Add user accounts
- Rate limiting
- Real-time dashboards