**Software Requirements Specification (SRS)**

**Title:** URL Shortener with MySQL and Node.js Backend

**Table of Contents**

1. **Introduction** 1.1 Purpose 1.2 Scope 1.3 Definitions, Acronyms, and Abbreviations 1.4 References
2. **Overall Description** 2.1 Product Perspective 2.2 Product Features 2.3 User Classes and Characteristics 2.4 Operating Environment 2.5 Design and Implementation Constraints 2.6 Assumptions and Dependencies
3. **System Features** 3.1 URL Shortening 3.2 URL Redirection 3.3 Database Storage
4. **External Interface Requirements** 4.1 User Interfaces 4.2 Hardware Interfaces 4.3 Software Interfaces 4.4 Communication Interfaces
5. **Non-Functional Requirements** 5.1 Performance Requirements 5.2 Security Requirements 5.3 Usability Requirements 5.4 Maintainability Requirements
6. **Other Requirements**
7. **Appendices**

**1. Introduction**

**1.1 Purpose**

This document specifies the requirements for a web-based URL shortener application that allows users to enter long URLs and generate short URLs. The backend is built using Node.js and MySQL for database storage.

**1.2 Scope**

The URL shortener enables users to create short links for long URLs. The system will store the mappings in a MySQL database and allow users to access the original URL by visiting the shortened URL. It aims to provide quick access, secure storage, and easy redirection.

**1.3 Definitions, Acronyms, and Abbreviations**

* **URL**: Uniform Resource Locator
* **API**: Application Programming Interface
* **Node.js**: JavaScript runtime for backend development
* **MySQL**: Relational database for storing URL mappings

**1.4 References**

* Node.js documentation
* MySQL official guides

**2. Overall Description**

**2.1 Product Perspective**

The URL Shortener is a self-contained system with three main components:

* **Frontend**: HTML, CSS, and JavaScript
* **Backend**: Node.js for API management
* **Database**: MySQL for storing URLs

**2.2 Product Features**

* URL shortening
* Secure database storage
* Quick URL redirection

**2.3 User Classes and Characteristics**

* **End Users**: Individuals needing to shorten and share URLs
* **Admin**: Responsible for maintaining database integrity

**2.4 Operating Environment**

* **Frontend**: Modern web browsers
* **Backend**: Node.js runtime environment
* **Database**: MySQL 8.0 or later

**2.5 Design and Implementation Constraints**

* Must use Node.js for backend processing
* Must store URLs securely

**2.6 Assumptions and Dependencies**

* Users have access to a web browser
* MySQL and Node.js are correctly installed

**3. System Features**

**3.1 URL Shortening**

**Description**: Users enter a long URL, and the system generates a unique short URL.

* **Inputs**: Long URL
* **Processing**: Generate a short identifier and store in MySQL
* **Outputs**: Shortened URL

**3.2 URL Redirection**

**Description**: Redirect users when they visit a short URL.

* **Inputs**: Short URL
* **Processing**: Retrieve original URL from MySQL
* **Outputs**: Redirect to the long URL

**3.3 Database Storage**

**Description**: Store long and short URLs in MySQL.

* **Inputs**: URL mappings
* **Processing**: Store, retrieve, and manage URL data
* **Outputs**: Persistent data storage

**4. External Interface Requirements**

**4.1 User Interfaces**

* **Form to enter long URLs**
* **Display of shortened URLs**

**4.2 Hardware Interfaces**

The system is platform-independent and runs on any device with a web browser.

**4.3 Software Interfaces**

* Frontend: HTML, CSS, JavaScript
* Backend: Node.js
* Database: MySQL

**4.4 Communication Interfaces**

The frontend communicates with the backend via HTTP APIs (e.g., REST).

**5. Non-Functional Requirements**

**5.1 Performance Requirements**

* The URL shortening must be completed within 1 second.

**5.2 Security Requirements**

* The database must be protected against SQL injection attacks.

**5.3 Usability Requirements**

* The UI must be responsive and user-friendly.

**5.4 Maintainability Requirements**

* The system should be easy to update and scale.

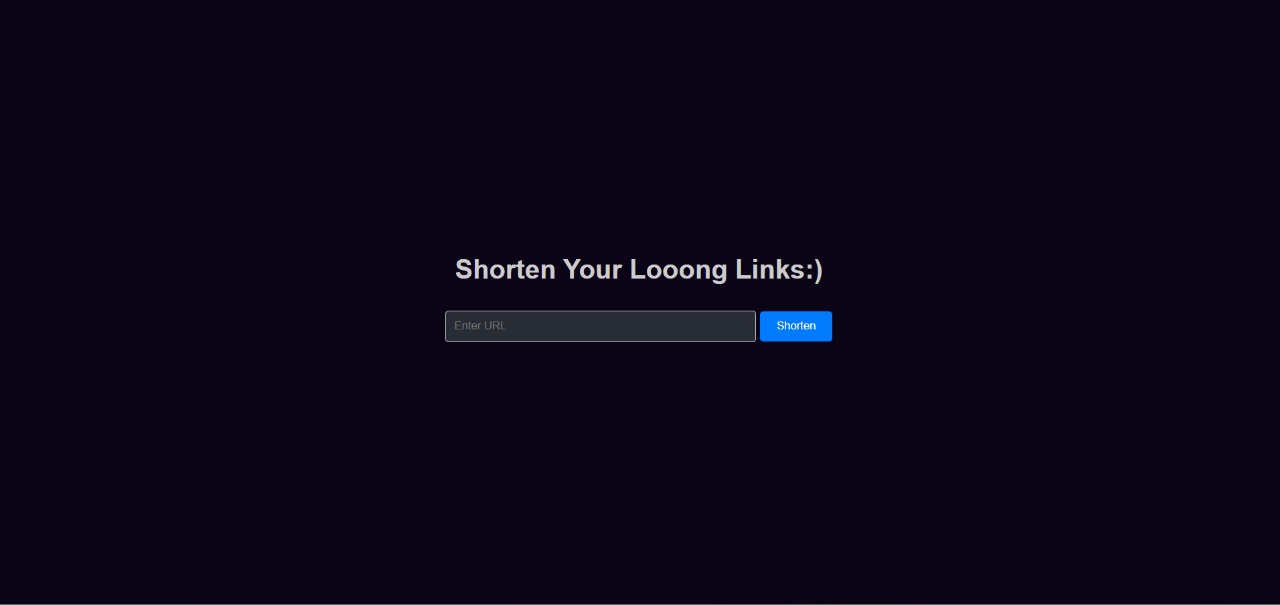
**6. Other Requirements**

* Regular database backups.
* Logging of URL creation events.

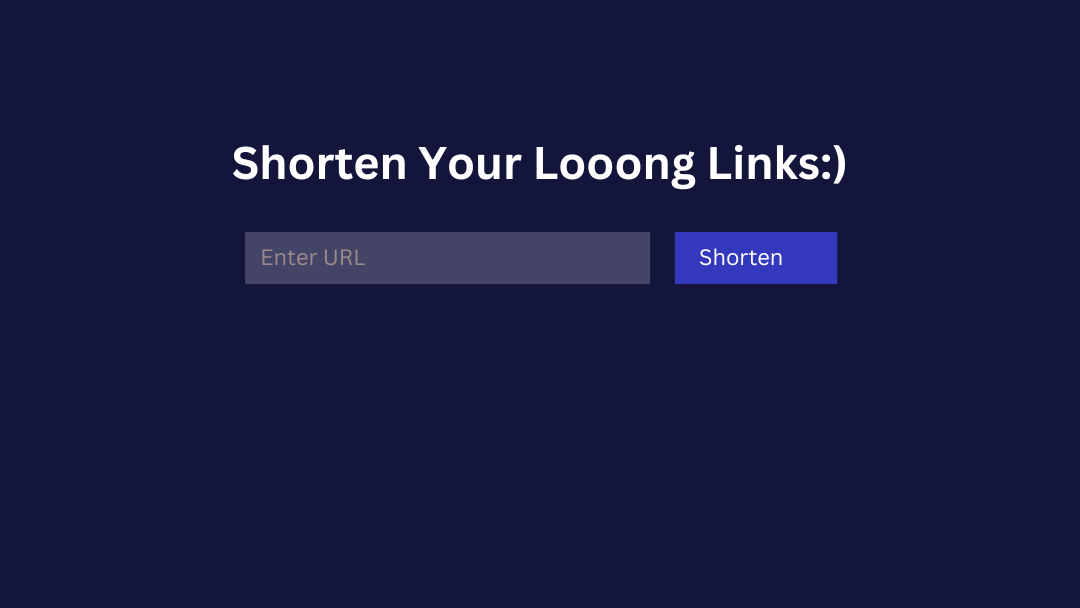
**7. Appendices**

* Tools used: Visual Studio Code
* Libraries: express, mysql, body-parser

**8.User Interface**

****

**9.Wireframe**

****