# Software Requirements Specification (SRS) for Blog Web Application

## 1. Introduction

### 1.1 Purpose

This SRS document outlines the requirements for a blog web application, designed to provide users with a platform to read, interact with, and subscribe to content. It includes functionality for users to log in, add articles, like, comment on articles, and subscribe to a newsletter.

### 1.2 Scope

The blog web application will be built using Django as the backend and HTML, CSS, and JavaScript for the frontend. Users will have interactive features, including article creation and editing. The application is designed to be modular and extendable, allowing for future enhancements.

### 1.3 Definitions, Acronyms, and Abbreviations

* **SRS:** Software Requirements Specification
* **Django:** Python-based web framework
* **Frontend:** Client-side components (HTML, CSS, JavaScript)
* **Backend:** Server-side components (Django)

## 2. Overall Description

### 2.1 Product Perspective

This application is a standalone web application that will run on a web server. It uses Django for backend processes, including data storage, authentication, and content management, with HTML, CSS, and JavaScript for frontend design and interactivity.

### 2.2 Product Functions

* **User functionalities:** Register, log in, view articles, like articles, comment on articles, subscribe to/unsubscribe from the newsletter, create and edit articles.

### 2.3 User Characteristics

* **End-users:** Individuals who view, interact with, and contribute content.

## 3. Functional Requirements

### 3.1 User Features

#### 3.1.1 Login and Registration

* Users can register by providing their name, email, and password.
* Users can log in using their registered credentials.

#### 3.1.2 Article Management

* **Add Article:** Users can create new articles by providing a title and content.
* **Edit Article:** Users can edit their existing articles.
* **Delete Article:** Users can delete their own articles.

#### 3.1.3 Article Browsing

* Users can browse a list of articles.
* Users can view individual articles on separate pages.

#### 3.1.4 Article Interactions

* **Like Article:** Users can like/unlike articles. The system will maintain a count of likes per article.
* **Comment on Article:** Users can add comments to articles.
* **View Comments:** Users can view other comments associated with an article.

#### 3.1.5 Newsletter Subscription

* Users can subscribe to receive newsletter updates.
* Users can unsubscribe from the newsletter at any time.

## 4. Non-Functional Requirements

### 4.1 Performance Requirements

* The system should handle up to 10,000 users with minimal latency.
* Article loading should take no longer than 2 seconds on average.

### 4.2 Security Requirements

* Only authenticated users can access features such as adding or editing articles.

### 4.3 Usability Requirements

* The interface should be intuitive and responsive across devices.
* Forms should include validation to help users avoid errors.

### 4.4 Reliability Requirements

* The system should be available 99.9% of the time.
* The system should include error-handling and logging for reviews.

### 4.5 Maintainability Requirements

* The codebase should be modular to support future enhancements.
* The system should have thorough documentation for developers.

## 5. Interface Requirements

### 5.1 User Interfaces

* **Login Page:** Allows users to log in to their account.
* **Home Page:** Lists recent articles with options to like and view.
* **Article Page:** Displays the article’s content, comments, like button, and a form to add comments.
* **Add/Edit Article Page:** Allows users to create or edit articles.

### 5.2 Software Interfaces

* **Database:** MySQL for storing data related to users, articles, comments, likes, and subscriptions.
* **Email Service:** For sending newsletters to subscribers.

## 6. Future Enhancements

* **User Analytics:** Track user engagement metrics such as views, likes, and comments.
* **Social Media Integration:** Allow users to share articles on social media.
* **Advanced Comment Moderation:** Implement automated tools to filter spam.