# Software Requirements Specification (SRS) Document for Parlor Booking Web Application

**Last Updated:** 28 July, 2025

## Table of Contents

1. Introduction
2. General Description
3. Functional Requirements
4. Interface Requirements
5. Performance Requirements
6. Design Constraints
7. Non-Functional Attributes
8. Preliminary Schedule and Budget
9. Appendices
10. Uses of SRS Document
11. Conclusion

## 1. Introduction

### 1.1 Purpose of this Document

This document outlines the software requirements for developing a web-based application for a beauty parlor. It aims to clearly define the functional and non-functional specifications to ensure successful development, delivery, and deployment of the system.

### 1.2 Scope of this Document

This document describes a parlor web application that enables users to browse services, book appointments, reschedule existing bookings, and have their appointments synced with Google Calendar. The application will feature a simple and intuitive UI to improve customer satisfaction. Development is expected to take approximately 6 weeks, with a moderate budget suitable for a small business.

### 1.3 Overview

The web application will consist of the following core modules: - Services Display - Appointment Booking - Rescheduling - Google Calendar Integration - Admin Panel (optional extension)

## 2. General Description

### 2.1 Product Perspective

The application will be developed as a standalone solution using modern web technologies, but may optionally integrate with existing systems (like Google Calendar) through APIs.

### 2.2 Product Functions

* Display services offered by the parlor
* Allow customers to book new appointments
* Enable rescheduling of appointments
* Integrate appointments with Google Calendar
* Responsive and user-friendly design

### 2.3 User Characteristics

* **Customers**: Looking to browse and book services. May have limited technical background.
* **Admin/Staff**: Responsible for managing service offerings and handling appointments.

### 2.4 Benefits

* Streamlined appointment system
* Reduced manual workload for staff
* Improved customer experience and retention

## 3. Functional Requirements

### 3.1 Services Display

* FR1.1: System shall display list of available services with name, description, price, and estimated time.

### 3.2 Appointment Booking

* FR2.1: Users shall be able to select a service, date, and time.
* FR2.2: Users shall input contact details (Name, Email, Phone).
* FR2.3: System shall store the booking in the database.
* FR2.4: System shall sync appointment with Google Calendar of parlor.

### 3.3 Rescheduling

* FR3.1: Users shall be able to view existing appointments using email or ID.
* FR3.2: Users shall be able to select a new date and time for rescheduling.
* FR3.3: System shall update both the database and Google Calendar.

### 3.4 Notifications (Optional)

* FR4.1: System shall optionally send confirmation and reminder emails/SMS.

## 4. Interface Requirements

### 4.1 User Interfaces

* Web-based UI accessible via browser.
* Mobile-responsive design.
* Form fields for service selection and booking.

### 4.2 External Interfaces

* **Google Calendar API**: For syncing and modifying appointments.
* **Database**: MongoDB or Firebase to store user and appointment data.
* **Optional APIs**: Email/SMS service for notifications.

## 5. Performance Requirements

### 5.1 Static Requirements

* PR1: Application should load on standard desktop/mobile devices within 2 seconds.

### 5.2 Dynamic Requirements

* PR2: Booking operation must complete within 3 seconds.
* PR3: Google Calendar sync must occur within 5 seconds post booking.
* PR4: Application must handle up to 50 simultaneous bookings without lag.

## 6. Design Constraints

* DC1: The frontend must be developed using React or Next.js.
* DC2: The application must support Google Calendar API v3.
* DC3: Hosting must be done on Vercel (frontend) and Render/Heroku (backend).
* DC4: Data must be stored securely with HTTPS and encrypted databases.

## 7. Non-Functional Attributes

* **Security**: Data encryption, Google OAuth2 for API usage.
* **Portability**: Web-based, accessible via all modern browsers.
* **Reliability**: System uptime of 99% expected.
* **Scalability**: Capable of scaling to 200+ bookings/day.
* **Maintainability**: Codebase to be modular and documented.
* **Usability**: Clean and intuitive UI with minimal training required.

## 8. Preliminary Schedule and Budget

### 8.1 Development Timeline

* Week 1: Requirements Gathering & Wireframing
* Week 2-3: Frontend Development
* Week 3-4: Backend Development & Google Calendar Integration
* Week 5: Testing & Bug Fixes
* Week 6: Deployment & Handover

### 8.2 Budget Estimate

* Design: $200
* Development: $800
* Hosting & APIs: $100
* **Total Estimated Cost: $1100**

## 9. Appendices

* A1: Google Calendar API Documentation: https://developers.google.com/calendar/api
* A2: TailwindCSS Docs: https://tailwindcss.com/docs
* A3: React Docs: https://reactjs.org/
* A4: Firebase Docs (if applicable)

## 10. Uses of SRS Document

* **Developers**: Understand scope and requirements clearly
* **Testers**: Build test cases from defined functionality
* **Clients**: Know exactly what will be delivered
* **Managers**: Estimate project costs and schedules
* **Support Team**: Know expected behaviors to troubleshoot effectively
* **Legal/Contract**: Serves as a mutual agreement between customer and developer

## 11. Conclusion

A clear and structured SRS is crucial for the success of the Parlor Booking Web Application. It ensures all stakeholders are aligned and sets a standard for implementation, testing, delivery, and post-launch support. By following the specifications in this document, the end product is expected to meet business objectives and provide a seamless experience to its users.