**Software Requirements Specification (SRS) for Online Lawyer Booking Web Application**

**1. Introduction**

**1.1 Purpose**

The Online Lawyer Booking Web App aims to provide a platform where users can find and book specialized lawyers for legal consultations and services. The platform also allows lawyers to register, showcase their expertise, and interact with potential clients. Admins manage the platform and ensure its smooth operation.

**1.2 Scope**

This web application serves three main stakeholders:

1. **Admin**: Manages users and lawyers, approves lawyer registrations, oversees subscriptions, and monitors chats.
2. **Lawyers**: Register as service providers, define their specialization, and interact with clients.
3. **Users**: Register to search for lawyers, book appointments, and communicate with lawyers.

The app includes functionalities like user authentication, lawyer subscriptions, chat systems, and appointment scheduling.

**1.3 References**

1. [Avvo](https://www.avvo.com/)
2. [Lawrato](https://lawrato.com/)

**1.4 Definitions, Acronyms, and Abbreviations**

* **SRS**: Software Requirements Specification
* **REST API**: Representational State Transfer Application Programming Interface
* **CRUD**: Create, Read, Update, Delete

**2. Functional Requirements**

**2.1 User Roles and Permissions**

1. **Admin**:
   * Approve or reject lawyer registrations.
   * Manage users and lawyers (CRUD operations).
   * Monitor chat sessions between users and lawyers.
   * Oversee subscription payments.
   * View reports on app activities.
2. **Lawyer**:
   * Register and submit details (license, CNIC, degree, specialization, etc.).
   * Update profile settings and upload documents.
   * Respond to user queries and chats.
   * Manage appointments.
   * Subscribe to premium plans for priority listing.
3. **User**:
   * Register and create an account.
   * Search for lawyers by specialization and city.
   * Book appointments with lawyers.
   * Interact with lawyers via chat after registration.
   * Post questions in specific categories.

**2.2 Core Features**

1. **User Registration and Authentication**:
   * Secure registration for both users and lawyers.
   * Password encryption and role-based login.
2. **Lawyer Specialization and Filtering**:
   * Categorize lawyers by specialization (e.g., divorce, child custody).
   * Filter lawyers by city and category.
3. **Appointment Booking**:
   * Calendar-based scheduling system for users.
   * Notification for lawyers and users on appointment booking, rescheduling, or cancellation.
4. **Chat System**:
   * Secure chat interface under admin supervision.
   * Open Q&A forum for users to post questions and get answers from lawyers.
5. **Lawyer Subscription**:
   * Subscription-based premium listing.
   * Payment gateway integration for lawyers to manage subscriptions.
6. **Admin Dashboard**:
   * Overview of all system activities.
   * Approval workflows for lawyer registrations.
   * Manage subscriptions, appointments, and chats.

**3. Non-Functional Requirements**

**3.1 Performance**

* The system should handle up to 10,000 concurrent users.
* Search results should load within 2 seconds.

**3.2 Security**

* Use HTTPS for secure data transmission.
* Implement role-based access control (RBAC).
* Encrypt sensitive data (e.g., passwords, CNIC).

**3.3 Scalability**

* Support additional features and users as the platform grows.

**3.4 Usability**

* Intuitive UI for all stakeholders.
* Mobile-friendly design (responsive layout).

**4. System Design**

**4.1 Technology Stack**

* **Frontend**: React.js
* **Backend**: Python (Django or Flask)
* **Database**: PostgreSQL or MySQL
* **Hosting**: AWS or DigitalOcean
* **Chat Functionality**: WebSocket or Firebase

**4.2 API Design**

1. **Authentication API**:
   * POST /register
   * POST /login
   * POST /logout
2. **Lawyer Management API**:
   * GET /lawyers (Search lawyers)
   * POST /lawyers/register (New registration)
   * PUT /lawyers/{id} (Update profile)
   * DELETE /lawyers/{id} (Admin action)
3. **Appointment Management API**:
   * POST /appointments
   * GET /appointments/{userId}
   * PUT /appointments/{id} (Reschedule)
4. **Chat API**:
   * POST /chats
   * GET /chats/{userId}
5. **Subscription API**:
   * POST /subscriptions
   * GET /subscriptions/{lawyerId}

**5. Database Design**

**Tables**

1. **Users**
   * UserID (Primary Key)
   * Username, Email, PhoneNumber, Password, Role
2. **Lawyers**
   * LawyerID (Primary Key)
   * Name, Email, LicenseID, CNIC, Degree, Specialization, City, Picture, ApprovalStatus, SubscriptionStatus
3. **Appointments**
   * AppointmentID (Primary Key)
   * UserID (Foreign Key)
   * LawyerID (Foreign Key)
   * Date, Time, Status
4. **Chats**
   * ChatID (Primary Key)
   * UserID (Foreign Key)
   * LawyerID (Foreign Key)
   * Message, Timestamp
5. **Questions**
   * QuestionID (Primary Key)
   * UserID (Foreign Key)
   * Category, QuestionText, Timestamp, AnsweredBy
6. **Subscriptions**
   * SubscriptionID (Primary Key)
   * LawyerID (Foreign Key)
   * StartDate, EndDate, PaymentStatus

**6. Assumptions and Constraints**

1. The platform will initially target users and lawyers within a single country.
2. Payments will support local payment gateways like Stripe or PayPal.
3. Lawyers must complete all mandatory fields in their profiles to be approved.

**7. Risks**

1. Delayed approval of lawyer registrations may impact user experience.
2. High server load during peak usage might slow down performance.

**8. Deliverables**

1. Functional web application (frontend and backend).
2. Complete API documentation.
3. Deployment instructions and server setup guides.