Great! Here's the **Software Requirement Specification (SRS)** for your freelancer-client project platform.

**📄 2. Software Requirement Specification (SRS)**

**Project Name:** *FreelancerConnect*  
**Prepared By:** [Your Name]  
**Date:** [Current Date]

**📘 1. Introduction**

**1.1 Purpose**

The purpose of this document is to detail the functional and non-functional requirements of *FreelancerConnect*, a web-based platform that connects freelancers with clients looking to outsource projects.

**1.2 Scope**

This system will allow freelancers to create profiles, bid for projects, manage ongoing jobs, and receive payments. Clients will be able to post projects, evaluate freelancer proposals, hire talent, and process payments. An admin will oversee the system.

**1.3 Intended Audience**

* Developers & Designers
* Project Stakeholders
* Testers & QA Engineers
* System Administrators

**1.4 Definitions**

* **Freelancer**: User who offers services.
* **Client**: User who posts jobs/projects.
* **Project**: A job/task posted by a client.
* **Proposal**: A bid/offer submitted by a freelancer.

**🧱 2. Overall Description**

**2.1 Product Perspective**

FreelancerConnect is a standalone web platform that follows a client-server architecture using REST APIs.

**2.2 Product Functions**

* User registration & role selection
* Login/logout functionality
* Profile management
* Project posting by clients
* Project browsing & bidding by freelancers
* Messaging between users
* Payment integration
* Admin dashboard (monitoring, reports, user management)

**2.3 User Classes and Characteristics**

| **Role** | **Description** |
| --- | --- |
| Freelancer | Can apply to projects, manage work |
| Client | Can post jobs, hire freelancers |
| Admin | Can manage users & resolve disputes |

**2.4 Assumptions and Dependencies**

* Users have internet access.
* Payments are processed through a third-party API.
* JWT authentication for security.

**⚙️ 3. Functional Requirements**

**3.1 User Management**

* FR-1: Users can register by choosing a role (freelancer/client).
* FR-2: Users can log in using email and password.
* FR-3: Users can update profile info, add skills (freelancer), or business details (client).

**3.2 Project Posting and Bidding**

* FR-4: Clients can post projects with title, description, budget, category, and deadline.
* FR-5: Freelancers can browse and filter projects.
* FR-6: Freelancers can send proposals with a message and bid amount.

**3.3 Hiring and Contracts**

* FR-7: Clients can view proposals and hire freelancers.
* FR-8: A contract is created once a freelancer is hired.
* FR-9: Both parties can mark milestones and submit deliverables.

**3.4 Messaging System**

* FR-10: Real-time messaging (WebSocket or Firebase).
* FR-11: Notifications for new messages, proposal updates, and hiring.

**3.5 Payments**

* FR-12: Secure payment flow using Razorpay/Stripe.
* FR-13: Admin holds payments in escrow until job completion.
* FR-14: Freelancer can withdraw earnings after project completion.

**3.6 Review & Rating**

* FR-15: Clients and freelancers can rate each other after job completion.

**3.7 Admin Panel**

* FR-16: Admin can view all users and projects.
* FR-17: Admin can deactivate or block accounts.
* FR-18: Admin resolves disputes and issues.

**📶 4. Non-Functional Requirements**

| **ID** | **Requirement** |
| --- | --- |
| NFR-1 | Application should be responsive and work on desktop/tablet/mobile. |
| NFR-2 | System uptime must be at least 99.5%. |
| NFR-3 | Data should be securely stored using encryption. |
| NFR-4 | APIs must respond within 2 seconds. |
| NFR-5 | Platform should support 10,000 concurrent users initially. |

**🧪 5. System Interfaces**

| **Interface** | **Description** |
| --- | --- |
| Frontend | React.js UI using REST APIs |
| Backend | Spring Boot APIs |
| Payment Gateway | Razorpay or Stripe |
| Database | MySQL or PostgreSQL |
| Messaging | WebSocket or Firebase |

**📂 6. Database Requirements**

Entities:

* **User**: id, name, email, password, role, profile info
* **Project**: id, client\_id, title, description, category, budget, deadline, status
* **Proposal**: id, project\_id, freelancer\_id, bid, message, status
* **Message**: id, sender\_id, receiver\_id, timestamp, content
* **Review**: id, project\_id, from\_user\_id, to\_user\_id, rating, comment

Let me know when you're ready for the next document:  
➡️ **Use Case Document**  
Just reply: **continue**