**Explanation of Each Service's Role**

**1. User Service**

* **Purpose:**
  + Manages user registration and assigns users to specific chat rooms.
  + Publishes user-related events (e.g., "User joined chat room") to Kafka topics for other services to consume.
* **Responsibilities:**
  + Exposes REST APIs for user registration.
  + Saves user data (e.g., username, assigned chat room) in the database.
  + Produces messages to a Kafka topic (user-events) whenever a user performs an action (like joining a chat room).

 Start with **User Registration API** (POST /users).

 Implement **Fetch All Users API** (GET /users).

 Add **Fetch Users by Chat Room API** (GET /users/chatroom/{chatRoom}).

 Complete **Delete User API** (DELETE /users/{id}).

 Implement **Update User Chat Room API** (PUT /users/{id}/chatroom).

 Add **Fetch User by ID API** (GET /users/{id}).

**2. Chat Service**

* **Purpose:**
  + Handles sending and receiving real-time chat messages between users in the same chat room.
  + Acts as a Kafka producer and consumer, connecting users via WebSocket for instant messaging.
* **Responsibilities:**
  + Uses Kafka topics dynamically for each chat room (e.g., chat-room-1, chat-room-2).
  + Sends messages to Kafka when a user sends a chat.
  + Listens to Kafka topics for new messages and broadcasts them to connected users via WebSocket.

**3. Message Storage Service**

* **Purpose:**
  + Persists all chat messages for historical data retrieval.
  + Provides APIs to fetch previous messages for a specific chat room.
* **Responsibilities:**
  + Consumes messages from all chat-room-{id} topics.
  + Stores messages in a database (e.g., MongoDB or PostgreSQL).
  + Exposes REST APIs for retrieving chat history by room or user.

**4. Kafka Config Module**

* **Purpose:**
  + Provides reusable Kafka producer and consumer configurations to avoid duplication.
  + Ensures a consistent setup across all services (e.g., bootstrap-servers, serializers).
* **Responsibilities:**
  + Centralizes Kafka configurations (e.g., retry policies, partitions).
  + Helps streamline integration with Kafka for each service.