□ TÀI LIỆU PHÂN TÍCH YÊU CẦU HỆ THỐNG BACKEND ỨNG DỤNG CHAT CƠ BẢN

1. Phân tích yêu cầu

🎯 Mục tiêu

Xây dựng hệ thống backend cho một ứng dụng chat cơ bản cho phép người dùng:

- Đăng nhập và trò chuyện **1-1** hoặc **theo nhóm (group)**.
- Gửi / nhận tin nhắn **văn bản** hoặc **hình ảnh**.
- Nhận **thông báo realtime** khi có tin nhắn mới.
- Theo dõi **trạng thái online / offline** của người dùng.
- Đảm bảo **tốc độ xử lý nhanh, dữ liệu an toàn**, và **khả năng mở rộng tốt**.

🔑 Các chức năng chính

Nhóm chức năng	Mô tả
Đăng nhập & xác thực	Đăng ký, đăng nhập, xác thực JWT token, quản lý phiên.
Quản lý người dùng	Cập nhật thông tin, avatar, trạng thái online/offline.
Trò chuyện 1-1	Gửi tin nhắn trực tiếp giữa 2 người dùng.
Trò chuyện nhóm	Tạo nhóm, gửi yêu cầu tham gia nhóm, admin duyệt.
Tin nhắn	Gửi, nhận, hiển thị tin nhắn text hoặc hình ảnh.
Realtime	Cập nhật tin nhắn và trạng thái người dùng theo thời gian thực.
Thông báo	Gửi thông báo khi có tin nhắn mới hoặc yêu cầu tham gia nhóm.

2. Yêu cầu đặc biệt

🔐 Xác thực & bảo mật

- Mỗi tin nhắn gửi lên server **phải kèm JWT token** trong header hoặc payload.
- Token chứa **id người dùng (kiểu int)** — dùng để xác thực người gửi.
- Backend sẽ:
    1. Giải mã JWT token để lấy user_id.
    2. Kiểm tra user_id có thuộc participants của conversation_id hay không.
    3. Nếu hợp lệ → xử lý tin nhắn; nếu không → trả lỗi **403 Forbidden**.

💬 **Trò chuyện 1-1**

- Khi người dùng A nhắn cho B, hệ thống kiểm tra:
    - Nếu conversation giữa 2 user đã tồn tại → dùng lại.
    - Nếu chưa → tạo mới conversation type = "direct".

👥 **Trò chuyện nhóm**

- Người dùng nhập **mã code (conversation code)** để gửi yêu cầu tham gia nhóm.
- Yêu cầu được thêm vào mảng requests của conversation tương ứng.
- Admin có thể **accept** hoặc **reject** yêu cầu:
    - Nếu **accept** → thêm user vào participants.
    - Nếu **reject** → xóa khỏi requests.

🖼️ **Gửi ảnh**

- Nếu tin nhắn chứa ảnh (attachments), file sẽ được:
    - Upload lên **Cloudinary**
    - Server lưu lại đường dẫn (URL) trong attachments.url.
- Frontend hiển thị trực tiếp ảnh dựa trên URL trả về.

⚡ **Realtime**

- Sử dụng **WebSocket (Socket.IO)** để:
    - Nhận/gửi tin nhắn tức thời.
    - Cập nhật trạng thái online/offline.
    - Gửi thông báo “new message” đến các client cùng conversation.

3. Cơ sở dữ liệu

💾 Công nghệ: MongoDB (NoSQL)

3.1. Collection users


```
{
  "id": 1,
  "username": "minhnguyen",
  "email": "minh@example.com",
  "avatar_url": "https://cdn.example.com/avatars/minh.jpg",
  "status": "online",              // hoặc "offline"
  "last_seen": "2025-10-13T07:42:00Z",
  "created_at": "2025-09-10T12:00:00Z",
  "role": ["ADMIN", "SHIPPER", "CLIENT", "STORE"]
}
```


Bổ sung gợi ý:

- Thêm trường password_hash (hoặc lưu ngoài, tuỳ hệ thống auth).
- Index nên có: id, username, email, status.

3.2. Collection conversations


```
{
  "_id": "conv_001",
  "type": "direct",                  // "direct" hoặc "group"
  "name": null,                      // chỉ có nếu type = "group"
  "created_by": "user_123",
  "code": "ABCD1234",                // mã để join nhóm
  "last_message": {
    "message_id": "msg_789",
```

```
    "sender_id": "user_456",
    "content_preview": "Hey, bạn rảnh không?",
    "sent_at": "2025-10-13T07:40:00Z"
  },
  "participants": [
    { "user_id": 1, "role": "ADMIN" },
    { "user_id": 2, "role": "MEMBER" }
  ],
  "requests": [1, 3],                    // danh sách user_id yêu cầu tham gia
  "created_at": "2025-09-12T10:00:00Z",
  "updated_at": "2025-10-13T07:40:00Z"
}
```

3.3. Collection messages

```
{
  "_id": "msg_789",
  "conversation_id": "conv_001",
  "sender_id": "user_456",
  "content": "Hey, bạn rảnh không?",
  "attachments": [
    {
      "type": "image",
      "url": "https://cdn.example.com/images/photo1.jpg"
    }
  ],
  "reply_to": "msg_788",
  "reactions": [
```

{ "user_id": "user_123", "emoji": "❤" }
],
"seen_by": ["user_123"],
"sent_at": "2025-10-13T07:40:00Z",
"updated_at": null,
"deleted": false
}

4. Công cụ hỗ trợ

Thành phần	Công nghệ sử dụng	Mục đích
Database	MongoDB	Lưu trữ người dùng, cuộc trò chuyện, tin nhắn
Backend	Node.js (Express hoặc NestJS)	Xử lý API, xác thực, logic chat
Realtime	Socket.IO / WebSocket	Gửi nhận tin nhắn và cập nhật realtime
Storage	Cloudinary	Lưu file ảnh
Frontend	ReactJS	Giao diện người dùng
Authentication	JWT (JSON Web Token)	Xác thực người dùng
Cache / PubSub (tuỳ chọn)	Redis	Quản lý session và realtime scaling

5. Quy trình gửi tin nhắn (flow tóm tắt)

1. Client gửi tin nhắn qua WebSocket / API → kèm JWT token.
2. Backend:
    - Giải mã JWT, lấy user_id.
    - Kiểm tra user_id có thuộc conversation.participants không.

- Lưu tin nhắn vào messages.
- Cập nhật conversation.updated_at và last_message.
- Gửi sự kiện realtime đến các client cùng conversation.

3. Nếu tin nhắn có attachments:
    - Upload ảnh lên Cloud Storage → lưu URL.
    - Trả về link ảnh hiển thị ngay.

6. Bảo mật và hiệu năng

- Tất cả API đều yêu cầu JWT hợp lệ.
- Chống truy cập trái phép vào conversation.
- Ảnh upload phải có **signed URL tạm thời**.
- Thiết kế **stateless backend** để dễ scale ngang.

✅ **Tóm tắt nhanh:**

Thành phần	**Mục tiêu**
Người dùng	Quản lý, xác thực bằng JWT
Cuộc trò chuyện	1-1 hoặc nhóm có quyền & yêu cầu join
Tin nhắn	Gửi text/image realtime
Realtime	Socket.IO để đồng bộ tức thời
Storage	Cloud Storage cho file
CSDL	MongoDB (NoSQL, linh hoạt, dễ mở rộng)

Bạn có muốn mình **vẽ sơ đồ kiến trúc hệ thống backend này (gồm API, WebSocket, DB, Storage, Realtime flow)** để minh hoạ cho tài liệu không?
Sơ đồ đó rất hữu ích nếu bạn chuẩn bị báo cáo, nộp đồ án, hoặc trình bày với nhóm phát triển.