🧩 COMPLETE REQUIREMENTS — SOCIAL MEDIA APP (MERN Stack)

🧠 1. Project Goal

Create a **fully functional social media platform** where users can:

- Sign up / log in securely
- Connect with other users (follow / unfollow)
- Post text, photos, and short videos (Reels)
- Like, comment, and share posts
- Chat in real-time
- Get notifications
- Manage profiles (bio, avatar, cover)
- Explore trending posts, users, or reels
- Use light/dark themes (optional)

⚙️ 2. Tech Stack (No Tailwind CSS)

Frontend

- **React.js**
- **React Router DOM** – for page navigation
- **Axios / Fetch API** – for backend requests
- **Redux Toolkit / Context API** – for global state management
- **Pure CSS or SCSS** – for styling
- **React Icons / Lucide React / Font Awesome** – for icons
- **React Player** – for video/reel playback
- **Socket.io-client** – for real-time chat and notifications

Backend

- **Node.js + Express.js**
- **JWT (jsonwebtoken)** – for user authentication
- **bcrypt.js** – for password hashing
- **Multer** – for handling image/video uploads

- **Cloudinary / AWS S3 / Firebase Storage** – for storing images & videos
- **Socket.io** – for real-time messaging and notifications
- **Nodemailer** – for email verification and password reset

Database

- **MongoDB (Atlas)** with **Mongoose ODM**

🗂️ 3. Database Structure (Detailed)

🧍 Users Collection

```
{
 _id,
 username,
 email,
 password,
 profilePic,
 coverPic,
 bio,
 followers: [userId],
 following: [userId],
 isPrivate: Boolean,
 createdAt,
 updatedAt
}
```

📸 Posts Collection

```
{
 _id,
 userId,
 caption,
 imageUrl,
 videoUrl,
 likes: [userId],
 comments: [
   {
     userId,
     text,
     createdAt
   }
 ],
 createdAt,
 updatedAt
}
```

🎥 Reels Collection

```
{
  _id,
  userId,
  videoUrl,
  caption,
  likes: [userId],
  comments: [
    {
      userId,
      text,
      createdAt
    }
  ],
  createdAt
}
```

💬 Chats Collection

```
{
  _id,
  members: [userId],
  messages: [
    {
      senderId,
      text,
      mediaUrl,
      createdAt
    }
  ]
}
```

🔔 Notifications Collection

```
{
  _id,
  recipientId,
  senderId,
  type, // like, comment, follow, message
  postId,
  isRead,
  createdAt
}
```

💡 4. Core Features (Phase-wise Breakdown)

Phase 1 – Authentication

✅ Signup, Login, Logout
✅ Password hashing (bcrypt)
✅ JWT authentication
✅ Email verification / password reset (optional)

Phase 2 – Profile System

☑ Update profile info (name, bio, profile pic, cover pic)
☑ Follow / unfollow users
☑ View followers/following list
☑ Private/public profiles

Phase 3 – Post & Feed System

☑ Create post (image or video)
☑ Like / unlike post
☑ Comment system
☑ Delete own post
☑ Feed of followed users
☑ Pagination / infinite scroll

Phase 4 – Reels (Short Videos)

☑ Upload short video (<60 seconds)
☑ Autoplay / mute toggle
☑ Like & comment on reels
☑ Explore section (trending reels)

Phase 5 – Chat System (Socket.io)

☑ One-to-one real-time chat
☑ Message delivery & seen status
☑ Image / file sharing
☑ Typing indicator

Phase 6 – Notifications (Socket.io)

☑ Real-time notifications (like, follow, comment, message)
☑ Mark as read/unread
☑ Notification dropdown

Phase 7 – Search & Explore

☑ Search users by name/username
☑ Search posts by caption/hashtags
☑ Trending users & reels

Phase 8 – Admin Panel (optional)

☑ Manage users
☑ Delete inappropriate content
☑ Handle user reports

🧱 5. Folder Structure

```
social-media-app/
 ├── client/              # React frontend
|    ├── public/
|    ├── src/
|   |    ├── components/
|   |    ├── pages/
|   |    ├── redux/ or context/
|   |    ├── assets/
|   |    ├── styles/       # your pure CSS or SCSS files
|   |    ├── utils/
|   |    ├── App.js
|   |   └── index.js
|
 ├── server/              # Express backend
|    ├── config/
|   |   └── db.js
|    ├── controllers/
|    ├── routes/
|    ├── models/
|    ├── middlewares/
|    ├── uploads/ (temp files)
|   └── server.js
|
└── .env
```

🔐 6. Backend API Endpoints (Example Set)

Auth Routes

Method	Endpoint	Description
POST	/api/auth/register	Register a new user
POST	/api/auth/login	Login
GET	/api/auth/user	Get logged-in user info

User Routes

Method	Endpoint	Description
GET	/api/users/:id	Get user profile
PUT	/api/users/:id	Update profile
PUT	/api/users/follow/:id	Follow user
PUT	/api/users/unfollow/:id	Unfollow user

Post Routes

Method	Endpoint	Description
POST	/api/posts	Create post
GET	/api/posts/feed	Get feed
GET	/api/posts/:id	Get single post
PUT	/api/posts/:id/like	Like post
PUT	/api/posts/:id/comment	Comment
DELETE	/api/posts/:id	Delete post

Chat Routes

Method	Endpoint	Description
POST	/api/chats	Create chat
GET	/api/chats/:userId	Get user’s chats
POST	/api/messages	Send message
GET	/api/messages/:chatId	Get messages

🧰 7. Tools and Services

Purpose	Tool
Database	MongoDB Atlas
Image/Video Storage	Cloudinary
Hosting Frontend	Vercel / Netlify
Hosting Backend	Render / Railway
Real-time Chat	Socket.io
Auth Tokens	JWT
CSS Styling	Pure CSS / SCSS
Email Service	Nodemailer
Deployment Environment	.env variables

🚀 8. Deployment Setup

- **Frontend:** Vercel / Netlify
- **Backend:** Render / Railway
- **Database:** MongoDB Atlas
- **Media:** Cloudinary (free tier supports images & videos)
- **Environment Variables (.env):**
- MONGO_URI=
- JWT_SECRET=
- CLOUDINARY_NAME=
- CLOUDINARY_API_KEY=
- CLOUDINARY_API_SECRET=
- PORT=5000

📅 9. Development Roadmap (Recommended Plan)

Phase	Module	Duration
1	Setup MERN project, DB connection	2 days
2	Auth (register/login/logout)	3 days
3	Profile system (CRUD + follow/unfollow)	4 days
4	Posts (CRUD, like/comment, feed)	5 days
5	Reels (video upload/playback)	5 days
6	Real-time chat (Socket.io)	5 days
7	Notifications (Socket.io)	3 days
8	Search + Explore	3 days
9	Testing + Deployment	3 days

Total: ~1 month (full working version)

mongodb+srv://ahmadtariq7799_db_user:GFs7KfLuoPtQoigH@cluster0.certgm7.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0