Great! Here's a solid **GitHub repository structure** for your **Super App** that includes **Reddit, Netflix, YouTube, WhatsApp, Zoom, Discord, and Flipkart-like features** in a modular way. 🚀

**📂 GitHub Repository Structure**

superapp/

│── backend/ # Backend (Node.js, Express, MongoDB/PostgreSQL)

│ ├── src/

│ │ ├── controllers/ # API logic (auth, posts, chat, video, etc.)

│ │ ├── models/ # Database models (User, Post, Video, Product, etc.)

│ │ ├── routes/ # Express routes for different features

│ │ ├── middleware/ # Auth middleware, rate limiting, logging, etc.

│ │ ├── config/ # Config files (DB, Firebase, JWT, etc.)

│ │ ├── utils/ # Helper functions

│ │ ├── server.js # Main server file

│ ├── .env # Environment variables

│ ├── package.json # Backend dependencies

│ ├── README.md # Backend setup guide

│

│── frontend/ # Frontend (Flutter)

│ ├── lib/

│ │ ├── screens/ # UI Screens (Home, Chat, Video, Profile, etc.)

│ │ ├── widgets/ # Reusable UI components

│ │ ├── services/ # API calls (Dio or http)

│ │ ├── providers/ # State management (Riverpod/Provider/Bloc)

│ │ ├── main.dart # Entry point of Flutter app

│ ├── pubspec.yaml # Flutter dependencies

│ ├── README.md # Frontend setup guide

│

│── database/ # Database schema & queries

│ ├── mongodb/ # MongoDB collections & indexes

│ ├── postgresql/ # PostgreSQL schemas

│ ├── redis/ # Redis cache setup

│ ├── firebase/ # Firebase Firestore rules & collections

│

│── docs/ # Documentation & API references

│ ├── API.md # API endpoints documentation

│ ├── ARCHITECTURE.md # Explanation of system design

│ ├── DEPLOYMENT.md # Steps to deploy backend/frontend

│

│── deployment/ # Scripts for deployment (Docker, Kubernetes, etc.)

│ ├── docker-compose.yml # Docker setup

│ ├── nginx.conf # Nginx configuration (for reverse proxy)

│

│── .gitignore # Ignore files for Git

│── README.md # Main project documentation

**🔧 Feature Breakdown in the Repo**

**1️⃣ Backend (Node.js, Express, MongoDB)**

| **Feature** | **Folder** | **Notes** |
| --- | --- | --- |
| **User Authentication** | routes/auth.js, models/User.js | JWT/Firebase Auth |
| **Posts & Comments (Reddit-style)** | routes/post.js, models/Post.js | REST API |
| **Real-Time Chat (WhatsApp-style)** | routes/chat.js, models/Message.js | WebSockets / Firebase |
| **Video Upload & Streaming (YouTube/Netflix-style)** | routes/video.js, models/Video.js | AWS S3 / Firebase Storage |
| **Live Streaming (YouTube Live/Twitch)** | routes/live.js | WebRTC / RTMP |
| **Video Calls (Zoom-style)** | routes/call.js | WebRTC / Agora SDK |
| **E-commerce (Flipkart-style)** | routes/shop.js, models/Product.js | Stripe/Razorpay |

**🚀 Deployment Strategy**

* **Frontend:** Deploy on **Firebase Hosting / Vercel**
* **Backend:** Deploy on **Render / AWS / DigitalOcean**
* **Database:** **MongoDB Atlas (NoSQL)** or **PostgreSQL (SQL)**
* **Storage:** **AWS S3 / Firebase Storage** for videos

**🛠 Example API Endpoints**

| **Feature** | **Method** | **Endpoint** | **Description** |
| --- | --- | --- | --- |
| **Auth** | POST | /api/auth/register | Register user |
| **Auth** | POST | /api/auth/login | Login user |
| **Posts** | POST | /api/posts | Create a post |
| **Posts** | GET | /api/posts | Fetch all posts |
| **Comments** | POST | /api/posts/:id/comment | Add comment |
| **Chat** | GET | /api/chat/:userId | Fetch messages |
| **Video Upload** | POST | /api/video/upload | Upload video |
| **Video Stream** | GET | /api/video/:id/stream | Stream video |
| **Live Stream** | POST | /api/live/start | Start live stream |
| **E-commerce** | POST | /api/shop/product | Add product |

**🔥 Next Steps**

✅ **Would you like me to generate the basic backend code structure?** (Express.js + MongoDB)  
✅ **Do you need help with API design or database schema?**