1. Executive Summary

This MVP defines an **internal mobile chat application** for a software agency:

- Real-time text messaging
- File sharing
- Voice notes
- No data stored locally on mobile
- Fetches all data from a central server
- Clears all data on disconnect
- Web-based **Admin Panel** for user and chat management

---

2. System Architecture

2.1 High-Level Architecture

```
Mobile App (React Native - Expo)
⬇ WebSocket + REST API
Backend API Server (Node.js + Express + Socket.IO)
⬇
MongoDB Database
⬇
File Storage (AWS S3 or Local)
```

2.2 Core Principles

- No local persistence on mobile
- Stateless client
- Real-time communication
- Role-based access control
- Admin-controlled user management

---

3. Technology Stack

3.1 Mobile Application

- React Native (Expo)
- Socket.IO Client (WebSocket)
- Expo AV (Voice recording)
- Expo Document Picker (File upload)
- React Context API (State management)

  No local storage (AsyncStorage, SQLite, SecureStore, MMKV)

3.2 Backend

- Node.js
- Express.js
- Socket.IO (Realtime messaging)
- MongoDB
- Redis (optional for presence & scaling)

3.3 Admin Panel

- Next.js
- Tailwind CSS
- JWT Authentication
- REST API Integration

---

4. Authentication Flow

1. User logs in via REST API
2. Server returns JWT (stored in memory only)

3. WebSocket connection initiated with token
4. On disconnect, app state cleared and redirected to login

---

5. Core Features

5.1 Text Messaging

- User sends message → emitted via WebSocket → stored in database → broadcast to participants → clients update in-memory state

5.2 File Sharing

- User selects file → uploaded to backend → backend stores in S3/local → file URL saved in database → message event emitted
- Files **not stored on device**

5.3 Voice Notes

- Record via Expo AV → upload to backend → stored on server → message event emitted
- Playback **streamed only**, no local save

---

6. Database Schema (MongoDB)

Users

- _id (ObjectId)
- name
- email
- password_hash
- role (admin | member)
- created_at

Threads

- _id
- name
- is_group (boolean)
- created_by (user_id)
- created_at

ThreadMembers

- _id
- thread_id
- user_id

Messages

- _id
- thread_id
- sender_id
- type (text | file | voice)
- content (text or file_url)
- created_at

---

7. Admin Panel Features

User Management

- Create / update / delete users
- Assign roles
- Activate / deactivate users

Chat Management

- View chat history
- Delete messages / threads
- Assign users to threads

---

8. Real-Time Events

Client → Server:

- connect
- join_thread
- leave_thread
- send_message
- typing

Server → Client:

- thread_messages
- new_message
- user_typing
- user_online
- user_offline

---

9. Security

- HTTPS only
- JWT authentication with short-lived tokens
- Role-based access control
- Input validation

- File type validation
- Rate limiting

---

10. Development Roadmap

Week 1

- Authentication
- User CRUD (admin only)
- Thread CRUD
- Basic text messaging

Week 2

- File upload
- Voice notes
- Admin panel development

Week 3

- Permissions & roles
- Presence system (online/offline)
- Testing and hardening

---

11. Future Enhancements

- Message pagination
- Push notifications
- Read receipts
- Message reactions

- Message editing
- End-to-end encryption

---

12. Folder Structure

Mobile App (Expo)

```
/src
  /api
  /context
  /screens
  /components
  /hooks
  /services
  /utils
```

Backend (Express + Socket.IO + MongoDB)

```
/src
  /controllers
  /services
  /sockets
  /middleware
  /models
  /routes
```

Admin Panel (Next.js)

```
/pages
/components
/context
/lib
```

---

13. Deployment

- **Backend:** Railway / Render / AWS
- **Database:** MongoDB Atlas / AWS
- **Admin Panel:** Vercel
- **File Storage:** AWS S3 or local server
