Full-Stack Internship Assignment

Project: Mini Team Chat Application (Slack-like)

Build and deploy a **mini team chat application** where users can communicate in real-time inside channels. The application should function as a basic, usable collaboration tool with multiple users interacting simultaneously.

1. Overview

The goal is to create a small but complete full-stack project with:

- Real-time messaging
- Channels
- User authentication
- Online/offline presence
- Message history with pagination
- A deployed, publicly accessible application

You may use **any tech stack** of your choice.

2. Core Requirements

2.1 User Accounts

- Users should be able to **sign up and log in**.
- Authentication can be implemented using sessions, JWT, or any method you prefer.
- Users should remain logged in on page refresh.

2.2 Channels

- Users should be able to:
  - View existing channels
  - Create new channels
  - Join or leave channels
- Show channel information:
  - Name
  - Members (or member count)

2.3 Real-Time Messaging

- Messages must appear instantly to all users in the same channel.

- Implement using WebSockets, SSE, or any real-time mechanism.
- Every message must be stored in a **database**.

Message structure should include:

- Sender user
- Channel
- Text content
- Timestamp

2.4 Online Status (Presence)

- Show which users are currently online.
- Presence tracking should work across multiple browser tabs and users.

2.5 Message History & Pagination

- When opening a channel, load recent messages.
- Implement pagination so that older messages can be loaded without fetching everything at once.

2.6 Frontend Interface

- Provide a clean and functional interface that allows:
  - Viewing channel list
  - Entering a channel
  - Viewing the chat history
  - Sending new messages
  - Seeing who is online

3. Optional Add-Ons (Bonus)

You may implement any of the following if you wish:

- Private channels
- Typing indicators
- Message editing or deletion
- Message search

If you add any optional features, list them briefly in your README.

4. Database & Hosting

You may use any database system (PostgreSQL, MySQL, MongoDB, etc.).

Some free database hosting options:

PostgreSQL:

- Neon
- Supabase
- Railway

MongoDB:

- MongoDB Atlas
- Railway

Backend and frontend can be deployed on any platform. Examples include:

- Vercel
- Netlify
- Render
- Fly.io
- Railway

You are free to use any other hosting service as well.

5. What to Submit

Send an email to **ajay@deeref.co** with subject:

Full-Stack Internship Assignment – <Your Full Name>

The email should include:

1. GitHub Repository Link(s)

- Public repo(s) with the complete project code.
- Include a README.md with:
  - Setup and run instructions
  - Tech stack used
  - Any assumptions or limitations
  - Optional features implemented (if any)

2. Deployed Application Link(s)

- Live frontend URL
- Backend base URL (if separate)

3. Screen Recording

Record a short video (**8–15 minutes**) that includes:

a) Demo of the Application

- Sign up and log in
- Create/join channels
- Real-time messaging shown between two browser windows
- Online/offline user indicators
- Pagination for older messages

b) Codebase Walkthrough

- Overview of folder structure
- Explanation of how:
    - Real-time messaging is implemented
    - Presence is tracked
    - Messages are stored and retrieved
    - Channels and membership work
- Mention any key design decisions or tradeoffs

You can share the recording as:

- Loom link
- YouTube unlisted link
- Google Drive link
- Or any other accessible format