Project Title: File Sharing App (Like WeTransfer)

Overview
Need to send large files quickly with a simple shareable link? This app works like **WeTransfer** — users can upload PDFs, ZIPs, images, or other files and instantly receive a unique download link. They can even send the link to someone via email.

Each upload link is temporary — it **expires automatically after a set time** (e.g., 24 hours) to keep the system clean and secure. Simple to use, and very effective.

You'll build a full-stack application with user-friendly upload/download functionality, secure file handling, and optional login for advanced features.

Tech Stack

Backend:

- **Node.js + Express.js** – handles API logic and file uploads
- **File Uploads**: **Multer** – to store files on the server securely
- **Unique Links**: **uuid** – generates random, hard-to-guess IDs for file sharing
- **Email**: **Nodemailer** – to send file download links via email
- **Time Handling**: **Moment.js** or native date libraries for expiry calculation
- **Authentication**:
  → **Passport.js** with either **JWT** (token-based) or **session** (cookie-based) login

---

Database (Pick One):

- **MongoDB** (with Mongoose) – for flexible file metadata storage
- **PostgreSQL** (with Prisma or Sequelize) – if you prefer structured tables

You'll store metadata like file name, path, upload time, expiry time, sender's email, download count, and more.

Frontend (React.js)

- Build a clean, modern UI using **React**
- Main screens include:
  → **Upload Page**: Select file, enter email (optional), and get the shareable link
  → **Success Page**: Shows link with copy/share options
  → **Download Page**: Displays download button with expiry info
  → **(Optional)** Login page for tracking uploaded/downloaded files
- Use Axios/Fetch to interact with the Express API
- Handle time display and countdown for expiry dynamically on the frontend

---

Key Features

Easy Uploads

- Users select a file and upload it via the React interface
- Optionally enter sender and recipient emails
- File is stored securely on the server using Multer
- Metadata saved in MongoDB or PostgreSQL

Shareable Download Links

- After upload, the app returns a **unique download link** using UUID
- Users can:
  → Copy the link
  → Or send it directly to someone via email (using Nodemailer)

Expiry and Auto-Cleanup

- Every file link has an **expiry time** (e.g., 24 hours)
- The backend checks and deletes expired files automatically
- Users see how much time is left via a countdown or relative time (`e.g., “expires in 5 hrs”`)

Track Downloads

- Record download counts for each file
- Optionally, log basic access info like timestamp, IP, or user (if logged in)

Security & Clean UX

- All files stored securely on the server
- Uploaded files are removed after expiration
- Unique download links are unguessable thanks to UUID
- Authentication (optional): Logged-in users can view/manage their own uploads

---

Database Schema Example

- `files (or file_uploads)`
- `- id (UUID)`
- `- filename`
- `- file_path`
- `- upload_time`
- `- expiry_time`
- `- sender_email (optional)`
- `- receiver_email (optional)`
- `- download_count`
- `- created_by (if using auth)`

(Optional) You can also have:

- `download_logs`
- `- file_id`
- `- timestamp`
- `- ip_address`
- `- user_id (if logged in)`