📌 Case Study Assignment
Full-Stack Developer
To-Do App

This case study will help us evaluate your proficiency across backend and frontend development, best practices, and system design. Please read the requirements carefully and deliver a complete, functional application based on the following specifications.

Deadline: Apr 24, 2025 at 4:00 PM EAT

Submit to: Link

📋 Project Overview

Goal: Build a full-stack To-Do App using modern technologies. This app should allow users to securely log in and manage their personal to-do items, with support for media upload, search, tagging, and more.

🔧 Technical Requirements

🌐 General Architecture

- Build two separate applications:
  - Frontend: Built with React.js (or any React-based framework).
  - Backend: Built with Node.js using either:
    - Express.js
    - Nest.js
- Frontend and backend must communicate via a REST API.

🔐 Authentication

- Implement a Login Page using one of the following authentication strategies:
  - JWT (JSON Web Tokens)
  - Firebase Authentication
- Access control:

- Each user must only view and manage their own to-dos.

✅ To-Do Management Page

- Allow users to:
  - Add to-dos
  - Edit to-dos
  - Delete to-dos
  - View a list of to-dos

🎯 Features & Requirements

🔄 Validation

- Use a validation library such as:
  - Joi
  - express-validator
  - zod

🗄️ Database

- Use a real database — no in-memory/fake DBs.
- Bonus: Use MongoDB

🎨 UI/UX

- Design the frontend using a component library:
  - Ant Design
  - Mantine

📦 Media and File Management

- Image Upload:
  - Upload and use images as thumbnails for to-dos.
  - Display them as small preview images.
  - Validate file type (only valid images).
- File Upload:
  - Support attaching downloadable files to each to-do.

🔍 Search & Filters

- Text Search: Implement backend support to search to-dos by keywords.

- Bonus:

  - Tagging: Add and filter to-dos using tags

  - Pagination: Backend-controlled logic

🚀 Bonus Features

- Use TypeScript across the stack.

- Use MongoDB as the database.

- Provide Dockerfiles and a Docker Compose setup for containerization.

- Deploy your application to a cloud service provider (e.g., Vercel, Netlify, Render, AWS, Heroku).

- Implement tagging and filtering of to-dos.

- Implement pagination logic in the backend.

📘 README.md Requirements

Your submission must include a comprehensive README.md with:

1. Running Instructions

   - How to start the backend and frontend apps.

   - Any required environment variables or configuration files.

2. Database Setup

   - Steps to set up and connect to the database.

   - Instructions to manually create a user for testing authentication.

✅ Submission Checklist

- Frontend with React or React-based framework

- Backend with Express.js or Nest.js

- REST API connection between both apps

- User authentication (JWT or Firebase)

- To-do CRUD functionality

- File/image upload

- Personal user access restrictions

- README with setup & usage

- Validation implemented

- Real database in use

- Bonus: TypeScript, MongoDB, Docker, Deployment, Pagination, Tags

📬 How to Submit

Please share a GitHub repository with:

- All source code (backend & frontend): Response Link

- Use your organizational email address

- The README.md

- Deployed links if applicable (for frontend/backend)

If you have any questions or run into blockers, feel free to reach out. Good luck and have fun building!