📌 Fullstack Web Project – Mini-CRM for Freelancers

Tech Stack: TypeScript | React | Express/NestJS | PostgreSQL

Duration: 4 days (Submited by 27.04.2025)

Mandatory: TypeScript, Responsive Design, Light/Dark Mode

Optional: Docker, Bonus Features

🧠 Project Overview

You are tasked with building a **Mini-CRM platform for freelancers**. This is a real-world, fullstack web application where you'll design and implement everything from authentication to database schema to responsive frontend UI.

The goal is to showcase your ability to design a working system from scratch using TypeScript, apply best practices, and make thoughtful technical decisions.

☑️ Core Requirements

🔐 Authentication

- Signup and Login functionality
- Use email and password (hashed securely)
- Protect all user routes so only the authenticated user can access their data
- Token/session-based auth (your choice, explain your decision)

👥 Clients

- Each user can create, update, delete, and view their own clients
- Required fields: name, email, phone
- Optional fields: company, notes

🗀 Projects

- A project belongs to a client
- Required fields: title, budget, deadline, status
- Projects should support full CRUD functionality

📝 Interaction Logs

- Log calls, meetings, or emails related to a client or project
- Fields: date, interaction type, notes

⏰ Reminders

- Add reminders tied to clients or projects
- Display a summary of reminders due this week

📊 Dashboard

- Show an overview:
    - Total clients
    - Total projects
    - Reminders due soon
    - Projects by status

You decide how to visualize and structure this part.

👩‍💻 Tech & Implementation Requirements

Frontend

- React + TypeScript
- No UI kits (e.g., no Material UI, AntD). Use TailwindCSS or custom CSS
- Fully **responsive design**
- Implement a **light/dark mode toggle** that remembers the user's preference
- Include client-side form validation

Backend

- Node.js with TypeScript (use Express or NestJS)
- Build a structured REST API with clear routing and modular layers (controllers, services, models)
- Secure endpoints with auth middleware
- Validate incoming data using Zod or similar

Database

- PostgreSQL with Prisma ORM (or an equivalent of your choice)
- Design the schema yourself and include a basic ERD diagram
- Each user must only have access to their own data

🔧 General Guidelines

- All code must be in **TypeScript**
- Structure your project using clean, modular, and scalable architecture
- Use **ESLint and Prettier** to maintain code quality
- Git commits should be meaningful and reflect your development process
- Store secrets in a .env file and use environment variables properly

🧪 Deliverables

Please submit the following in your GitHub repository:

- Working frontend and backend projects
- README with:
    - Setup instructions
    - Tech stack used
    - ERD (Entity Relationship Diagram)
    - Summary of your approach and decisions
- API documentation (Postman collection or Swagger if available)
- Sample test user (or signup flow)

💡 Optional but Appreciated

- Dockerized setup (docker-compose)
- Basic testing (unit or e2e)
- Search/filter for clients and projects
- Role-based permissions
- Notifications system (in-app or simulated)

🤯 Intentionally Open-Ended

We're leaving **some areas undefined** on purpose:

- How you persist theme preference
- How you visualize dashboard data
- How you organize your folders/modules
- How you define project statuses
- How logs and reminders are structured in your schema

This allows us to understand your thought process, problem-solving, and initiative. Feel free to explain your decisions in the README.

🧑‍⚖️ Evaluation Criteria

We’re evaluating the following:

- ✅ Code quality and structure
- ✅ Full TypeScript usage and typesafety
- ✅ UI responsiveness and dark mode functionality
- ✅ Auth, data protection, and validation
- ✅ Thoughtful schema and database design
- ✅ Git practices and documentation

Please reach out if you need clarification during the challenge.

We’re looking forward to seeing your approach!

Submission Rules: **Please submit your assignment by April 27, 2025.**
Ensure you follow the instructions below when submitting:

1. Full Name
2. Email Address
3. Mobile Number
4. Updated CV
5. GitHub link(s) to your completed task
6. Live site link (optional)
7. ERD Diagram (as Image or PDF)