

Hello!

Thank you for your interest in the **Full Stack Engineer** role at TurboVets.

***Please note: For U.S. applicants, TurboVets does not provide visa sponsorship. Applicants must be authorized to work in the U.S. without sponsorship now or in the future.***

## Overview

This take-home assessment simulates the kind of problems we tackle at TurboVets — with a focus on **security, scalability, and full stack architecture**.

You'll build a **secure task management system** using a modular **NX monorepo** with both frontend and backend components.

The goal is to design and implement a system where users with different roles can securely manage tasks within an organization.

## Estimated Time

Up to **8 hours**. We respect your time — please stop at 8 hours even if it is incomplete. We're evaluating depth and decision-making more than surface polish.

## What We're Really Evaluating

We care most about **security, correctness, and architectural discipline**. A great submission shows:

- **Real JWT authentication**
- **Service-layer RBAC enforcement**
- **Organization-level scoping** for data visibility
- Clean, maintainable **NestJS + Angular** code structure
- Working **Task CRUD** and **Add User** flows

UI polish is a bonus — **security and reliability** come first.

## The Assessment: Secure Task Management System

Design and implement a role-based task management app that supports multiple users and organizations.

The system should include authentication, user roles, and secure task operations. Beyond that, how you model access control, handle authentication, and structure the codebase is up to you — just be ready to explain your reasoning.

## Core Requirements

### Backend (NestJS + TypeORM + SQLite/Postgres)

- Implement authentication and authorization appropriate for a multi-user environment
- Define and enforce user roles.
- Support creating, updating, viewing, and deleting tasks.
- Include at least a minimal hierarchy or grouping structure between users and organizations.
- Log or track important system actions in some form.

### Frontend (Angular + TailwindCSS)

- Provide a simple, functional UI for authentication and task management.
- Persist session state appropriately.
- Design with responsiveness and usability in mind.
- How you approach state management is your choice — explain your reasoning in the README.

## Submission Package

Please submit the following:

1. **GitHub repository** or ZIP file containing your codebase
2. **README** that includes:
  - How to set up and run the project
  - The architecture and design rationale
  - How access control and user roles are handled
  - Example requests or workflows that demonstrate functionality
  - What you would add or improve with more time

## Video Walkthrough (5–10 Minutes)

Record a short walkthrough explaining your implementation.  
Focus on the **architecture and security logic**, not just the UI.

Your video should cover:

- How authentication and RBAC enforcement work
- How guards and service-level checks are implemented

- What happens when a restricted role (e.g., Viewer) tries to modify data
- Key design choices and trade-offs
- What you'd improve with more time

Format: `.mp4` or `.mov` ( $\leq 100$  MB). A simple webcam or screen recording is perfect.  
No need for edits — clarity matters more than production quality.

## Focus Areas (What We Care About Most)

We're looking for sound engineering decisions that reflect secure, maintainable, and scalable full stack design.

Focus on:

- Implementing access control that's enforceable and logically consistent
- Building authentication that protects sensitive data and user context
- Ensuring data visibility aligns with user roles or organizational boundaries
- Delivering core functionality end-to-end (Login, Add User, Task CRUD)
- Tracking or logging meaningful actions
- Writing clear setup documentation and explaining your architectural choices

## Submission Checklist

Do a final pass to ensure:

- Your authentication and access control logic works as intended
- Role restrictions behave predictably
- Your README explains how to run and understand the system
- Sensitive information and secrets are not exposed in code or commits

## Deadline & Submission

Please submit your assessment within **4 business days** of receiving this email — by **11:59 PM EST**.

Need extra time due to scheduling conflicts? No problem, just let us know.

**Submit your solution here:** [Submission Form](#)  
**Download the challenge file:** [Assessment Package](#)

We're excited to see your engineering thinking in action.

**Good luck — and thank you again for applying to TurboVets!**