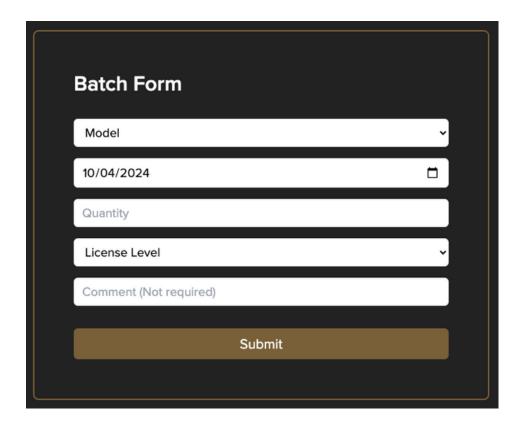




Part 1: Technical Assessment



Dear Candidate,

Welcome to the technical assessment for the Web Developer position at our organisation.

This task aims to evaluate your skills in developing a web application using NextJS, TypeScript, and Docker.

Please carefully follow the instructions below.





Task Overview:

Building a Containerised NextJS Application with PostgreSQL Database.

Frontend Requirements:

Replicate the provided form in NextJS.

Implement the form on the index route, which is the root directory of your site (e.g., localhost:3000/).

Utilise TailwindCSS for styling to a reasonable extent.

Ensure the use of TypeScript in your implementation.

The form should include:

- A dropdown for the "Model" field with options:
- 1. Model 1
- 2. Model 2
- 3. Model 3
- A dropdown for "License Level" ranging from 0 to 9.
- An input field for "Quantity" (integer).
- Other necessary form fields.

Frontend Hint:

• Make proficient use of flexbox for layout purposes.

Docker Requirements:

Containerise the NextJS portion, ensuring it runs in its own Docker container. Set up a PostgreSQL container to host the database.

Use Docker-compose to define your application in a YAML file.





Backend Requirements:

Utilise NextJS's server environment as your backend.
Implement Prisma as your ORM for querying the PostgreSQL database.
Define your database schema using Prisma's schema file.

Form Submission Logic:

When the form is submitted, in the DB an entry in a 'batches' table should be created. The entry should correctly contain each of the submitted fields: model, date, license level, quantity, and optional comment.

In another table called 'machine_numbers', for the quantity provided, create an entry with a unique random string called 'serial_number' and make sure to create a relation to the associated batch.

(E.g., On form submission a row is inserted into 'batches', and if the quantity was 5, then 5 rows would be inserted into 'machine_numbers').

Submission Guidelines:

Create a New Repository:

- Initialise a new GitHub repository for this task (do not mention FCL within the code base).
- Ensure the repository is set up with a Next.js project, and the form is accessible at the index route (e.g., localhost:3000/).

Evaluation Criteria:

- Successful replication of the provided form with the specified features.
- Proper use of NextJS, TypeScript, and TailwindCSS.
- Correct implementation of Docker containers using Docker-compose.
- Prisma integration for database operations.
- Handling of form submissions and batch creation.