Tractor inspector

Imagine that you are part of a small development team in a modern agricultural company. The desire of the management is for the company to become the most modern agricultural company by far. To ensure this, all processes related to the operation of the company are computer-supported.

In our case, tractors and combines are hidden under the term modern agricultural machinery. All of our machines have telemetry, which means that all activities of tractors and combines are traceable.

We have attached data from three tractors that were recorded in one day.

We would like you to create a website that will provide an overview of all tractors, their data and its editing.

Boilerplate

We have prepared a simple boilerplate which should help you get started. It utilizes Docker Compose for a PostgreSQL database inside the `db` folder and a simple ExpressJS application inside the `backend` folder.

The containers can be started via the following command:

```
docker compose up -d
```

As part of this boilerplate, we have also prepared an empty `frontend` folder, which can be used for your assignment.

No part of this boilerplate is required, feel free to modify it or use anything else that allows CRUD operations. The tractor session data is available inside the `db/csv` folder.

Data

The tractor data is inside CSV files and imported into the database as a seed inside the `vehicle_sessions` table. The structure of the table can be seen inside `db/seed.sql`. The `serial_number` column is unique per tractor.

Backend

Once started, the backend is available on `http://localhost:3000`. Initially the backend has two endpoints to get you started:

- `GET /tractors` -> Returns a list of serial numbers
- `GET /tractors/:serialNumber` -> Returns the whole session for a specific tractor

You will have to extend/rework this during your assignment.

Assignment

For the purpose of this application, we have conceived the following web pages that we want you to implement:

List of tractors

This page should contain an overview of all tractors in the database with the following data:

- Tractor serial number
- Total working hours of the tractor (`total_working_hours_counter` in the database or "Total working hours counter [h]" in the CSV file)
- Picture (select any picture of the tractor)

Tractor data

When selecting an individual tractor, open the page with all the tractor data. It should contain at least the following components:

Header

Header with serial number of the selected tractor and navigation button on the previous page

Tractor data table

Table with all tractor data available in the CSV file or in your database - except for the tractor name and serial number .

The first column should be the one named `date_time` inside the database or "Date / Time" in the CSV file. By default, the table should be sorted in ascending order from this column.

The text in this column should link to the edit page.

For an added challenge, implement sorting by all (or only specific) columns to work properly with paging.

Paging component

In addition to the table, we also need a paging component that will allow navigation through the results pages.

The number of simultaneously displayed results ("page size") should be adjustable with a numeric input field or with a dropdown / select menu.

Paging information should be reflected in the URL so that we can share (or save) this link. When reopening these links should see the same subset / page data (assuming that the data in between are not changed)

Edit Data tractor

Click on any of the "Date / Time" value in the previous table to open a page that allows you to edit the information that lines.

Header

Header with serial number and DateTime value of the line to be edited and back navigation button

Form

Offer the option of editing all parameters found in the database/CSV file except the serial number and "Date / Time" value.

Also add a button to confirm the entered values.

It is also desirable to implement some basic validation of input fields.

Other specifications

- Our stack is Vue + TypeScript. It would be preferable for you to use it, but feel free to use any other equivalent SPA framework.
- Follow the "responsive web design" paradigm and support at least one CSS breakpoint and / or use a CSS framework that takes care of you (e.g. Tailwind).
- Include a `Readme.md` with instructions on how to run your solution inside the `frontend` folder.
- If you want you can skip the implementation of an additional tractor data editing page and use inline data editing in the tractor table.
- Imagine you are handing your solution over to the end customer, hence you want to provide a code you'll be proud of. Also don't forget to, where applicable, write down a short description on how your solution works and a set of instructions on how to use it.

Optional for extra credit

The data also includes the locations of the tractors at a given moment. As an added challenge you can implement an overview of the movement of tractors over time.

Imagine a Google Maps map where the tractors are placed and each tractor has a track where it moved during a specific session. The tractor should move along this track with the help of the "Timeline" control, where you can change the time. Whenever this time is changed, the tractor moves to show its location during that time.

In this task, you have a completely free hand on how to implement it. Imagine how you could best visualize the data that we collect during the session.