autoSense FullStack Developer

Take-home exercise

Dear developer, thank you for taking the time to complete this exercise as part of your application to autoSense. Please read the instructions below carefully. If you have any questions please contact edmond@autosense.ch for assistance - we are glad to answer any questions you may have regarding the exercise.

The task

As a fullstack developer, we would like you to build a simple web application that can display fuel stations on a map where the user (administrator) can add new stations and change some basic details of existing ones.

High level requirements

- Build a frontend and a backend application, use separate reposotiries
- Your application is available on Github and provides instructions on how to run it

Frontend requirements

- The fuel stations are displayed on a map
- When clicking on a station, the station details are displayed
- User can update the following fields of the stations:
  - Name
  - Prices
- User can add a new station with all of it's details
- User can delete a station
- No user registration/login is required, consider the user already logged in
- Preferred technology: Vue.js
- Recommendation: Use a CSS framework and component library (e.g Vuetify)

Backend requirements

- The backend provides RESTful APIs to serve the frontend application
- The backend implements API key authentication for HTTP requests (please hardcode the API keys on the frontend)
- Pick your own database technology or use an in-memory database
- Use the sample data provided with the exercise for your schema definition
- Preferred technologies:
  - Node.js with Typescript
  - Express.js

Nice-to-haves

These features are not mandatory, only if time allows. It is better to delivery the above requirements in the highest quality, however, you can get extra points for the following features:

- API documentation (in the repository's Readme or in a Swagger file)
- High level architecture diagram
- End-to-end and unit tests

We will evaluate the following:

- The backend implementation, where we place the most focus compared to the frontend
- Correct usage of RESTful API principles
- Code quality (e.g clean code, readability)
- Code optimization (e.g runtime complexity, logic)
- The architecture of the application (e.g scalability, maintainability)
- Attention to details (e.g unused, uncommented code)

We will not evaluate the following:

- The design decisions for the web application
- Time it takes to provide the solution (we prefer quality over a half baked solution)

Appendices

- The sample dataset can be found in test-data.json attached to this email

When you completed the exercise

Please send the following information to jobs@autosense.ch

- Github repository URL for backend and frontend
- Frontend web application URL
- Backend API URL and authentication
- Link to any additional documentation