Mercedes-Benz

Full-Stack Developer Quiz

Thank you very much for filling out this quiz to the best of your competencies. The purpose of this quiz is to evaluate your areas of expertise.

If a question is not clear to you or using notations which you are not familiar with, feel free to rephrase the question or lay assumptions which help you in solving the problem.

Good luck and thank you again for your time!

1. Star Wars Web Application

The goal of this exercise is to develop a Web application that displays data from a third-party API. The application must be divided into two different microservices running as containers: Frontend and backend.

AS A Star Wars fan
I WANT to see the information about PEOPLE and PLANETS being able to sort the information and search by too **SO THAT** I can have fun searching Star Wars information.

Acceptance criteria

- Web application to display information from SWAPI (https://swapi.co/) in a web browser.
- The Web application should read and display information in two different tables, from the API endpoints:
  - People
  - Planets
- Each table should be paginated displaying 15 items per page.
- There should be a search input to allow filtering by name for each table. All searches will use case-insensitive partial matches. Eg: Searching by "sky" at the PEOPLE table should return at least "Luke Skywalker".
- Each table should allow to sort the data by "name" and "created" fields, in descending and ascending order, both cases. The mechanism for sorting should be designed following the "Open-closed" principle and implemented in the backend.
- For the frontend you can use any JS framework. Currently we are using Angular but feel free to use any framework of your choice.
- For the backend, you can use any of these Java frameworks:
  - Spring Boot
  - Micronaut
- Provide a Docker Compose file to be able to deploy the application on port 6969.

Definition of Done

- The application works as defined in the Acceptance criteria.
- Provide source code of the web application.
- Provide a clear README file about how to run the application, from scratch, using Docker Compose.
- Integration tests green.

We will evaluate:

- Follow Software Engineering principles.
- UX/UI designs.
- API performance.
- Clean code.