A silver logo with a star

AI-generated content may be incorrect.

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!**

Star Wars Web Application

**Goal:** Build a fullstack web application that displays and manages data from the Star Wars API (<https://swapi.info/>) The system must be split into two microservices: a backend and a frontend.

**AS A** Star Wars fan

**I WANT** to explore data about People and Planets using a responsive interface, **SO THAT** I can sort, search, and simulate insights about this 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 and 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.
* Frontend must be built using React (preferred), Angular, or Vue.
* Backend must be written in Python, using FastAPI or Flask.
* Use Docker to containerize backend and frontend.
* Provide a Docker Compose setup that runs the system on port 6969.

**Bonus (Optional)**

The requirements below are not mandatory. The fact of not doing them will not discard you from the hiring process neither reduce points. In case you decide to implement these requirements, they must work as expected, otherwise the ones not working will not be considered.

* Add a mock endpoint /simulate-ai-insight that, given a person or planet name, returns a generated description (fake AI output).
* Add basic logging and/or monitoring to backend (e.g., log search/sort events).
* Use environment variables to prepare the system for deployment on **Azure**, **AWS**, or **GCP**.
* Add a basic frontend loading state and error display.

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.

**May the Force be with you!**