Frontend Test 2

👋 Welcome to the VW DIGITAL:HUB Frontend technical test!

We appreciate the time and effort you've invested in reading this document. We believe that you would make an excellent addition to our team, and we would like to learn more about your technical background to proceed with the application process.

Time is our most precious asset, so this is why we have devised a technical assessment that should be appropriate and it will allow us to identify your technical capabilities. The code challenge is expected to take approximately 20 hours, but take the time you need in order to deliver something that you will be proud of. 🙂

📝 Submitting the technical challenge

We ask that you submit the challenge through GitHub or by email (sending the code as a zip). Kindly add the specified GitHub user (XXXX) as a collaborator to your repository.

- If you need guidance on this process, please consult the "*Inviting collaborators to a personal repository*" section in the GitHub Documentation Page.
- Don’t worry if you can not finish the test completely! You can explain your reasons and with your own words how you will develop the pending parts.

✉️ If you have any doubts or questions, feel free to contact our Human Resources email at any time, and we will gladly assist you.

🔧 Technical aspects to Keep in Mind

- When working on the code challenge, it is important to prioritise writing code that is clean, easy to read, and maintainable.

  This means paying attention to things like variable names, code organization, and overall structure. Well-organized code makes it easier for other developers to understand, modify, and maintain it in the long run.

- Include a `README.md` file in the project, where you should explain the project's deployment, configuration and execution steps.

  Also include separated technical documentation related to the decisions and technologies you have employed.

  AI Tool Usage Documentation: Document the use of AI coding tools (e.g., GitHub Copilot, ChatGPT). In the `README.md`, explain which AI tools were used, how they were used and why you used these ones. Discuss how the developer's own input was integrated with the AI's suggestions.

  Feel free to mention any assumptions you have made while interpreting the challenge. Some aspects of the challenge have been intentionally left open to interpretations, it is crucial for us to understand your reasons for choosing one approach over another.

  Please include in the `README.md` your name and surname for a better tracking.

- As developers, we love to sleep soundly at night. Therefore, it would be ideal if you ensure your code works correctly.

  Feel free to use any testing approach that suits your project best, such as unit tests or integration tests. By thoroughly testing your code, you not only guarantee its functionality and correctness, but also demonstrate a professional commitment to producing reliable and robust software.

  Remember, a well-tested code is a reliable code, providing peace of mind for both you and your team.

- Working as a collaborating team requires some level of coordination and synchronization.

  A proper Git usage: commits, pull requests, branching strategy, etc. will be really appreciated. Pretending to develop the challenge assuming there could

be other people coding other features in parallel, will help us value how you may deal with those escenarios.

💻 The Challenge

Build a dummy app that:

- Required: Use some public API or create your own mocked API with JSONServer.
- Required: React Client that consumes that API and can list, show, create, update, and remove the retrieved data on the APP. Use asynchronous views on templates.
- Required: Use some State Manager (Context, Jotai, Redux, etc...)
- Optional: Decouple your frontend components like in a Micro Framework approach

📒 Functional requirements

- Create a DataTable to sort a list of items with some information, it could have good points as:
  - A search bar that allows search by any field.
  - Sort by field.
  - Add, modify or delete items from the table with an intuitive UX/UI template.
  - Choose one item from the table and display it in a showcase format.
- Optional points:
  - Creativity
    - **Chose a UI / CSS Framework and explain why you have chosen this one**
  - Responsive
    - Today most people use their phone to browse the web. And tables can be a problem. **Explain you solutions for a responsive site in the code or in the** `README.md`

- CLEAN code
- SOLID principles
- Good performance
- Comment your code where needed. Don’t just comment everything, think about useful comments!
- Unit Testing is awesome but you may go further 😉 ?

⚙️ Technical requirements

- Write client side in TypeScript.
- Use a CSS framework if it helps in the UI or develop your own CSS architecture.

📄 Other ideas

- Design and document at least three reusable components.
  - **Task**: Include a section in the `README.md` explaining the design decisions, how the components can be reused, and any trade-offs made during development.
- Implement performance optimizations (e.g., code splitting, lazy loading, memoization). Document the optimizations in the `README.md`, explaining why they were necessary and how they improve the app's performance.
- Ensure the app meets basic accessibility standards (e.g., ARIA roles, keyboard navigation). Include a section in the `README.md` detailing the accessibility features implemented and any tools used to test accessibility.
- Conduct a self-review and refactor the code.
  - **Task**: Document the review process in the `README.md`, including any issues found and how they were addressed. Highlight areas where AI suggestions were improved or modified.
- Set up a CI/CD pipeline for automated testing and deployment.
  - **Task**: Document the CI/CD setup in the `README.md`, including the tools used (e.g., GitHub Actions, Jenkins) and the steps for deployment.