TP4 Web REPORT

Question 1: Why you should not commit credentials on git?

Because it is very important information that is not meant to be public.

Question 2: Why may you want different configurations depending on the environment? Give an example.

Because first of all: security , it is very important to not store all of the important credentials all inside of one file. It can help for testing if we want to test only a certain part of our code and that certainly helps for the performance of the software considering its scalability.

Question 3: While being a well-working solution, it suffers from maintainability issues. Please expose and discuss them.

If there is an intricate between events and props it makes it very hard to debug. Plus there will be a hard time when we will need to access the user object, we will have to pass the object through props which make it harder for us to refractor.

Question 4: What is the bug if the inject is not reactive?

If inject is not reactive, components relying on the injected user object will not re-render when the user data changes. So that means the interface will not be updated for example.

Question 5: Build a comparison table between the various state management strategies available, especially about pros and cons. Optionally, feel free to explore other ways not covered in that tutorial.

	Pros	Cons
Props	-easy to use -simple to learn	-Prop drilling -lot of coupling

Provide & inject	-no prop drilling -cleaner components	- Requires careful management of reactivity.
State libraries	-Centralized state -Easy debug	Huge learning curve Too complex for small scale projects

Question 6: Imagine a developer in your team suggests to exclusively manage the state with stores. Therefore, it recommends not to rely on props and provide anymore. Would you accept this? An argued answer is expected.

Pros :

For a very complex and huge application with multiple components, to centralize using a store would be efficient for the structure and ensures stability. The debugging would become much easier, tracking changes would become easy. Would be a very appropriate solution if the team is experienced.

Cons:

Considering the use of only stores on small projects is not a feasible idea, and if the team is not accustomed with the concept, the learning curve is really steep. And lastly linking the components too close will reduce the reusability of them.

Question 7: What is the performance difference between:

- **<a href="/conversations">Conversation</a>**
- **<router-link to="/conversations">Conversations</router-link>**

The <a> tag when clicked will reload the whole page rendering and reloading all the components.
When <router-link> is clicked, it will render the component and will change the url, so it'll be faster.