Vorbereitungsauftrag - Asynchrones JS, Kommunikation, API’s

|  |  |
| --- | --- |
| Datum | 15.12.2023 |
| NDK | Web Engineer NDK HF |
| Modul | Web-Technologien Aufbau |
| Thema | GraphQL |
| Plattform | Selbststudium |

**Buch Pro MERN Stack / Kapitel 5: The About API**

|  |  |
| --- | --- |
| **Aufgabe** | **Lösung** |
| 1. Use the same URL in the browser as well as command line for cURL. For example, type *curl http://localhost:3000/graphql*, which is the same URL as we used in the browser to invoke the Playground. Or, copy paste the *curl* command we used for doing a GET request for the about field. What do you see? Can you explain the difference? Hint: Compare the request headers. | Das Aufrufen derselben URL hat unterschiedliche Auswirkungen:   * Browser: Widergabe der GraphQL Playground-Benutzeroberfläche * cURL: Aufruf der API   Der Unterschied wird durch die Auswertung des “Accept”-Headers durch den Apollo Server verursacht.  Der “Accept”-Header ist ein Teil der HTTP-Anfrage, der angibt, welche Medientypen der Client akzeptieren kann. In diesem Fall analysiert der Apollo Server den “Accept”-Header. Wenn er den Wert “text/html” erkennt, was typisch für Browser-Anfragen ist, liefert er die GraphQL Playground-Benutzeroberfläche. Andernfalls führt er die entsprechende API-Aktion aus. |
| 1. What are the pros and cons of using GET vs. POST for read-only API calls? | Caching: GET-Anfragen werden zwischengespeichert im Cache. Das hat unterschiedliche Auswirkungen je Browser, welche sich nicht vorhersagen lassen. Deshalb sollten API-Ergebnisse vom Server abgerufen werden und nicht gecacht werden.  Deshalb ist in diesen Fällen die Verwendung von POST sicherer, da Browser die Ergebnisse bei POST nicht zwischenspeichern. |