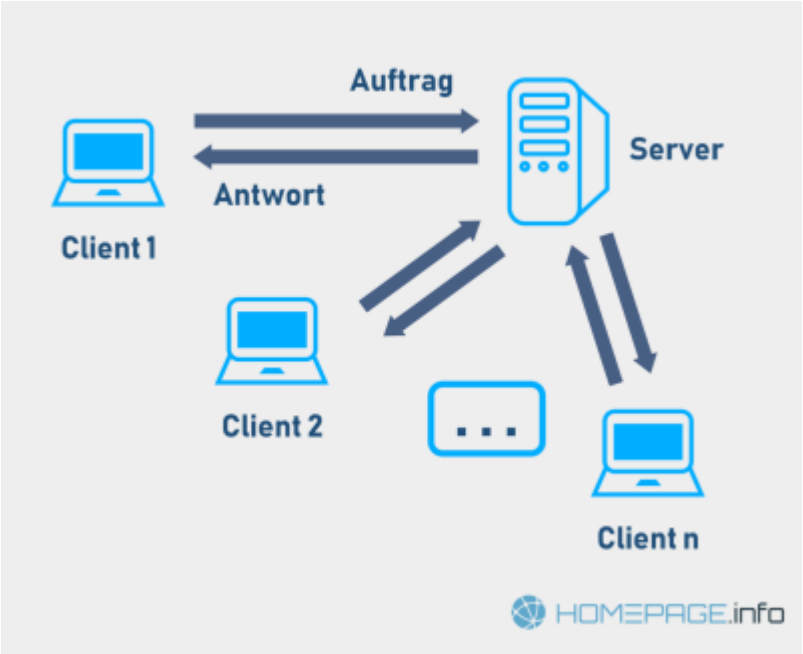
LU03c - CRUD TBD

1. Einleitung: Webkommunkation bzw. das Client-Server-Modell

Bevor wir uns mit dem Quellcode befassen, ist es wichtig, den Kommunikationsfluss im Internet zu verstehen. Dieser Austausch folgt dem Client-Server-Modell.



Stellen Sie sich diesen Prozess wie einen Briefwechsel vor:

- **Der Request (Anfrage):** Sie (der Client) senden einen Brief an den Server. Auf dem Umschlag steht, ob Sie nur Informationen erhalten möchten (**GET**) oder ob Sie ein Paket zur Verarbeitung mitschicken (**POST**).
- **Die Verarbeitung:** Der Node.js-Server nimmt Ihren Brief entgegen, liest die Absicht aus und entscheidet, welche Logik ausgeführt werden muss.
- **Die Response (Antwort):** Der Server sendet Ihnen eine Rückmeldung (Antwort), die das Ergebnis Ihrer Anfrage sowie einen Statuscode enthält.

2. Theorie: Die HTTP-Methoden im Detail

In der Webentwicklung nutzen wir das **HTTP-Protokoll**. Die Wahl der Methode definiert die Semantik (den Sinn) Ihrer Anfrage. Solche Anfragen können, je nach Anforderung, auf unterschiedliche Arten durchgeführt werden. Die wichtigsten Arten/Methoden sind **GET** und **POST**.

GET: Informationen abrufen

Die GET-Methode dient dazu, Daten vom Server anzufragen, ohne diese zu verändern. Es werden also

Daten nur abgefragt, nicht aber in irgend einer Weise geschrieben (Einfügen, Verändern, Löschen).

- **Datenübertragung:** Parameter werden offen an die URL angehängt (z. B. suche?artikel=laptop).
- **Sichtbarkeit:** Da die Daten in der Adresszeile stehen, sind sie für jedermann sichtbar und werden im Browserverlauf gespeichert.
- **Einsatzgebiet:** Anzeigen von Webseiten, Suchanfragen, Abrufen von Profilbildern.

POST: Daten übermitteln

Die POST-Methode wird verwendet, um Daten zur Verarbeitung an den Server zu senden. Unter Verarbeitung verstehen wir **Einfügen**, **Verändern** oder **Löschen**.

- **Datenübertragung:** Die Informationen befinden sich im sogenannten **Body** (Körper) der Anfrage. Sie sind nicht in der URL sichtbar.
- **Sicherheit:** POST ist sicherer für sensible Daten wie Passwörter, da diese nicht in Logs oder im Verlauf auftauchen.
- **Einsatzgebiet:** Benutzerregistrierungen, Login-Formulare, Hochladen von Dateien.

3. Der Node.js Server und seine Komponenten

Ein Node.js-Server ist die Laufzeitumgebung, die Ihre JavaScript-Logik auf der Serverseite ausführt. Er besteht aus folgenden Kernkomponenten:

- **Die V8-Engine:** Sie sorgt für die schnelle Ausführung Ihres Codes.
- **Der Event-Loop:** Diese Komponente ermöglicht es dem Server, viele Anfragen gleichzeitig zu bearbeiten, ohne bei langsamen Aufgaben (wie Datenbankabfragen) zu blockieren.
- **Routing:** Dies ist der Wegweiser Ihres Servers. Er entscheidet: „Wenn eine GET-Anfrage an /kontakt kommt, sende die Kontaktseite zurück.“
- **Middleware:** Dies sind Hilfsfunktionen, die die Anfrage vor der eigentlichen Logik untersuchen (z. B. um zu prüfen, ob die mitgeschickten Daten im JSON-Format vorliegen).
