Architekturbegründung

Die Anwendungsarchitektur basiert auf Datensätzen, die von Nutzern aus einer Datensammlung abgerufen und geschrieben werden können. Wichtig ist dabei, dass die Datensätze nicht verteilt über verschiedene Anwender gespeichert werden, sondern an einer zentralen Stelle gesammelt und gespeichert werden. Im Gegensatz zu anderweitig verteilten Architekturen wie Peer-to-Peer, bietet sich eine Client-Server-Architektur für dieses Projekt bestens an.

Server

Der Server übernimmt dabei die Schnittstelle hauptsächlich zum Lesen und Schreiben von persistenten Daten. Dazu kommt noch Logik zum Versenden von Benachrichtigungen, Mails und Authentifizierung sowie Authorisierung der Nutzer. Jene Funktionalitäten benötigen eine zentrale Anlaufstelle und können von Client nicht übernommen werden. Trotzdem soll der Server nicht mit vielen zusätzlichen Funktionalitäten ausgestattet werden, um ihn als simpel aufrufbare Schnittstelle mit wenig unterschiedlichen Rückgabewerten nutzen zu können. Dies spart außerdem Zeit bei der Entwicklung des Clients, da Aufrufe generalisierbar sind.

Client

Neben der zunächst einfach anmutenden Anzeige der Datensätze, übernimmt der Client doch einige Aspekte der Anwendungslogik. Als offensichtlichster Punkt zählt hierzu die Strukturierung der vom Server ausgelieferten Daten, da der Server (wie oben beschrieben) die Rolle einer sehr einfachen Schnittstelle einnimmt. Weiterhin soll die Verifizierung von Studenten nur über die Client-Applikation und nicht über ein weiteres Element wie einem Browser stattfinden, um nach außen hin die Wirkung zu erzielen, dass der Client das einzige Element der Applikation ist.

Internet

Zwischen Client und Server befindet sich das Internet als Kommunikationsschicht. In dieser Schicht befinden sich Komponenten wie Mailserver oder Google Cloud Messaging, die während des Ablaufs in Anspruch genommen werden.

Funktionalitäten

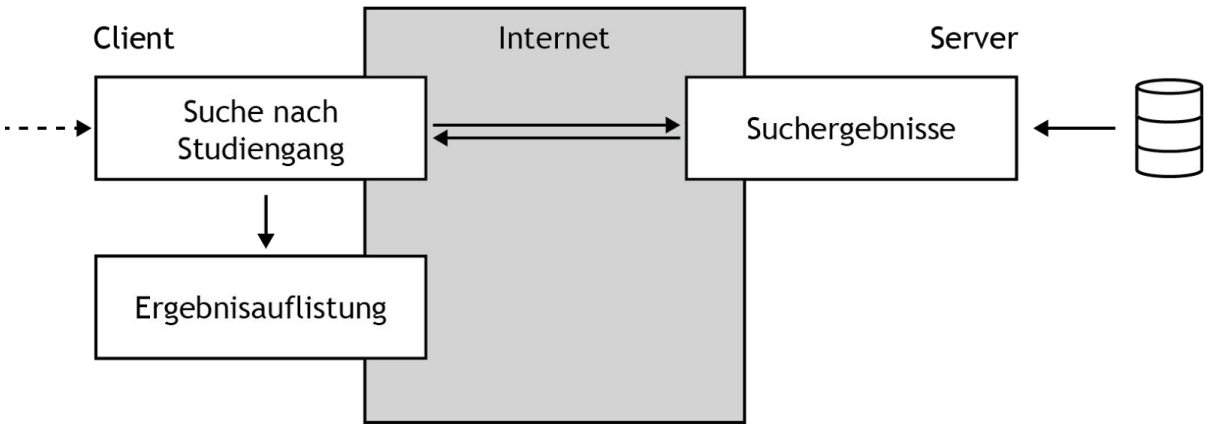
Diagramm

Um für die Entwicklung gerüstet zu sein, wurden die einzelnen Funktionalitäten getrennt betrachtet und zunächst auch als einzelne Diagramme erstellt. Die Diagramme sollen letztendlich die Entwicklung vorbereiten und unterstützen, weshalb sich für einen nicht standardisierten Diagrammtyp als Mischung aus Kommunikations-, Ablauf- und

Verteilungsdiagramm entschieden wurde. Die spezifischeren Funktionalitäten wurden generalisiert und auf Client- und Server aufgeteilt.
Nichtsdestotrotz können einige dieser abgegrenzten Diagramme aneinandergehängt werden, um einen größeren Ablauf darzustellen.

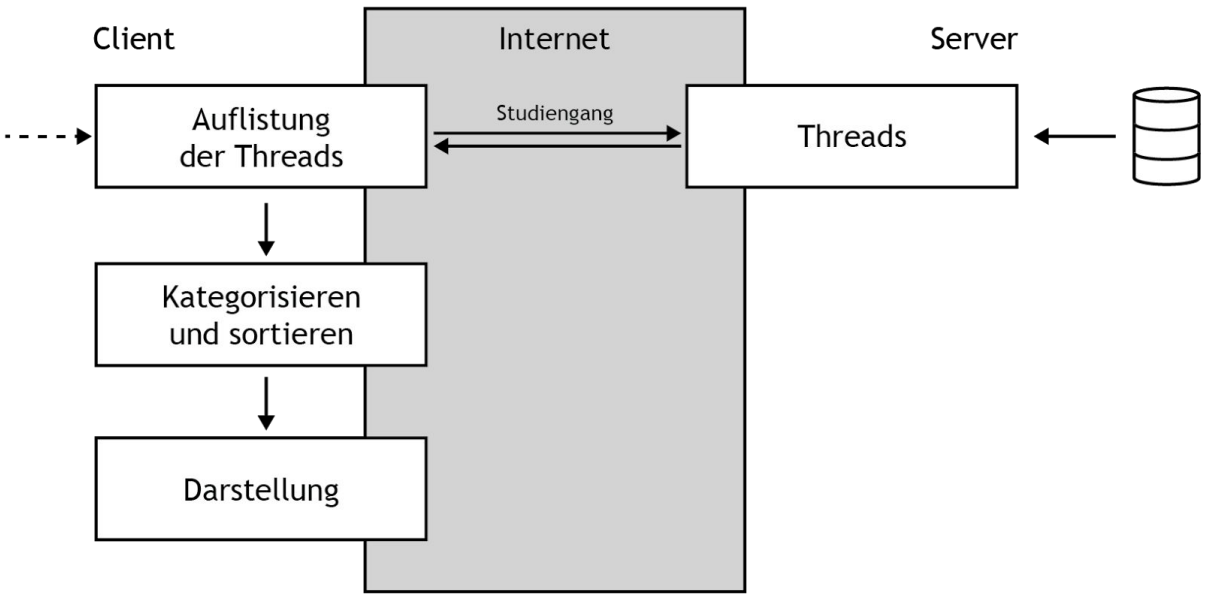
Suche

Die erste Funktionalität, die ein unregistrierter Nutzer durchgeht, ist die Suche nach einem Studiengang an einer Hochschule. Dabei kann zuerst nach einem Studiengang gesucht oder aber auch zuerst nach einer Hochschule gesucht werden. Dies wurde als *Suche nach Studiengang* generalisiert. Die Suche besteht aus dem Suchbegriff vom Client, welche dann am Server über eine Datenbank Suchergebnisse ausgibt und vom Client aufgelistet wird.



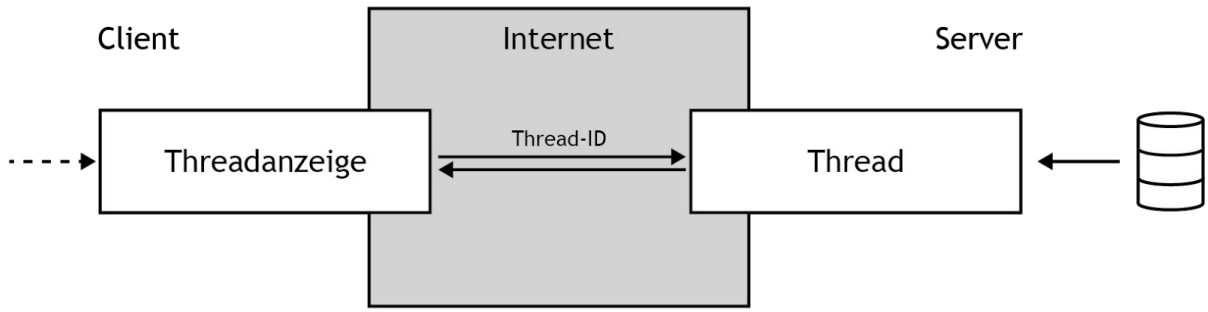
Threads auflisten

Ein *Thread* bezeichnet einen Beitrag samt Diskussionen in einem spezifischen Studiengang. Folglich muss zu einem spezifischen Studiengang vom Server eine Liste von Threads abgefragt werden, die dann nach Kategorien vom Client sortiert und ausgegeben werden müssen.



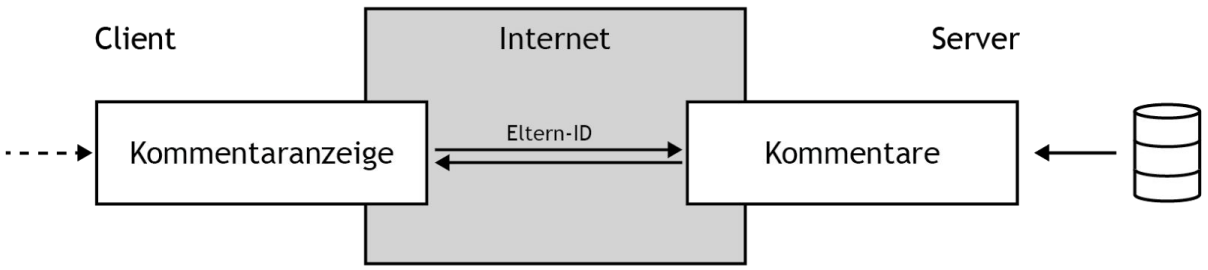
Thread anzeigen

Bei der Auflistung von Threads werden zur Datensparsamkeit nicht die Inhalte der Threads übertragen. Folglich muss ebenfalls der Inhalt eines einzelnen Threads abrufbar sein. Dies entspricht einem einfachen Request und Response, es ist keine zusätzliche Logik notwenig. Vom Server werden unter gewissen Umständen nicht alle geschachtelten Kommentare in einem Thread ausgegeben, hierfür muss eine gesonderte Funktionalität genutzt werden (im nächsten Abschnitt beschrieben).



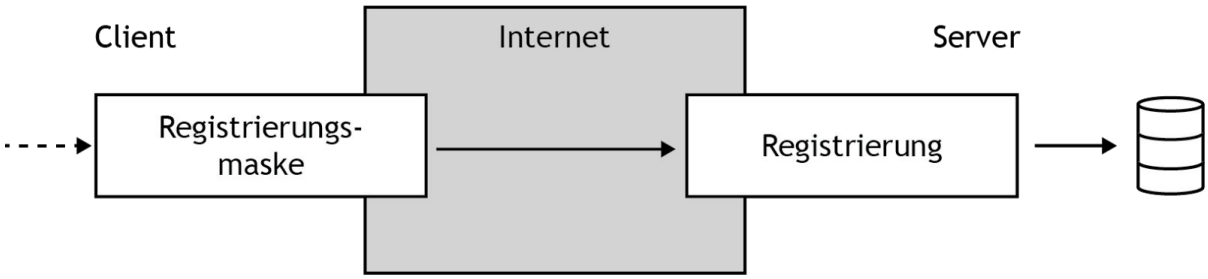
Kommentare anzeigen

Einerseits werden ab einer bestimmten Schachtelung von Kommentaren in der Threadausgabe jene tiefgeschachtelten Kommentare nicht mit ausgegeben, andererseits sollen auch einzelne Kommentar-Diskussionen aufrufbar sein. Hierfür wird eine Funktion benötigt, die nur diesen Kommentar-Strang anzeigt. Dies wird auch als Ziel-Ansicht einer Benachrichtigung genutzt.
Wie bei der einzelnen Thread-Anzeige handelt es sich hierbei um ein einfaches Request-Response-Verfahren.



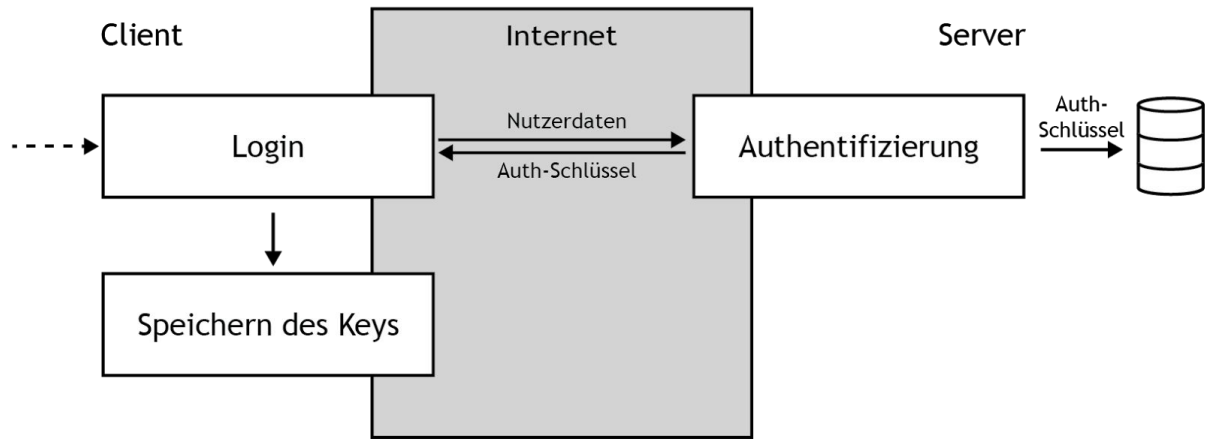
Registrierung

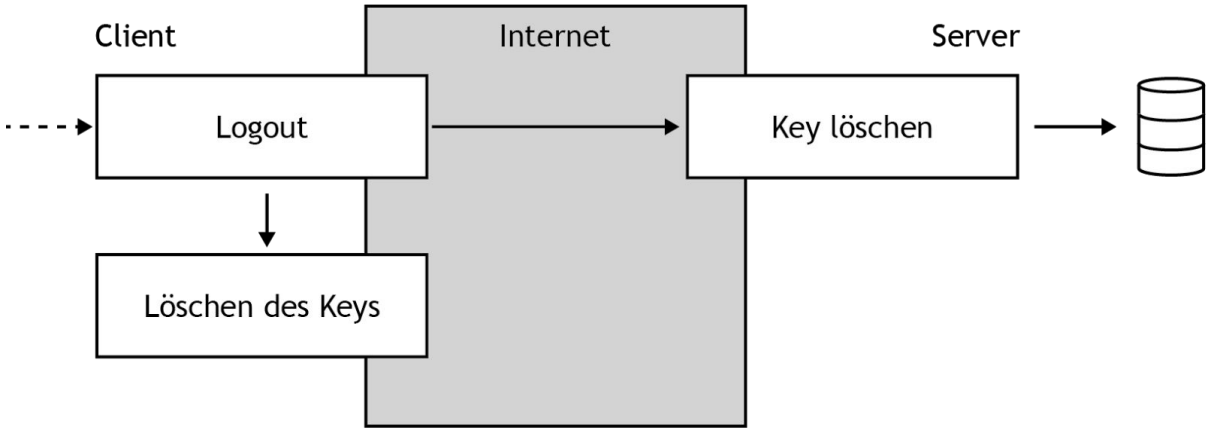
Die grundlegende Registierung hat keine zusätzlichen Funktionalitäten außer dem Anlegen eines Nutzers. Der Nutzer muss sich registrieren, um Beiträge oder Kommentare verfassen zu können.



Login / Logout

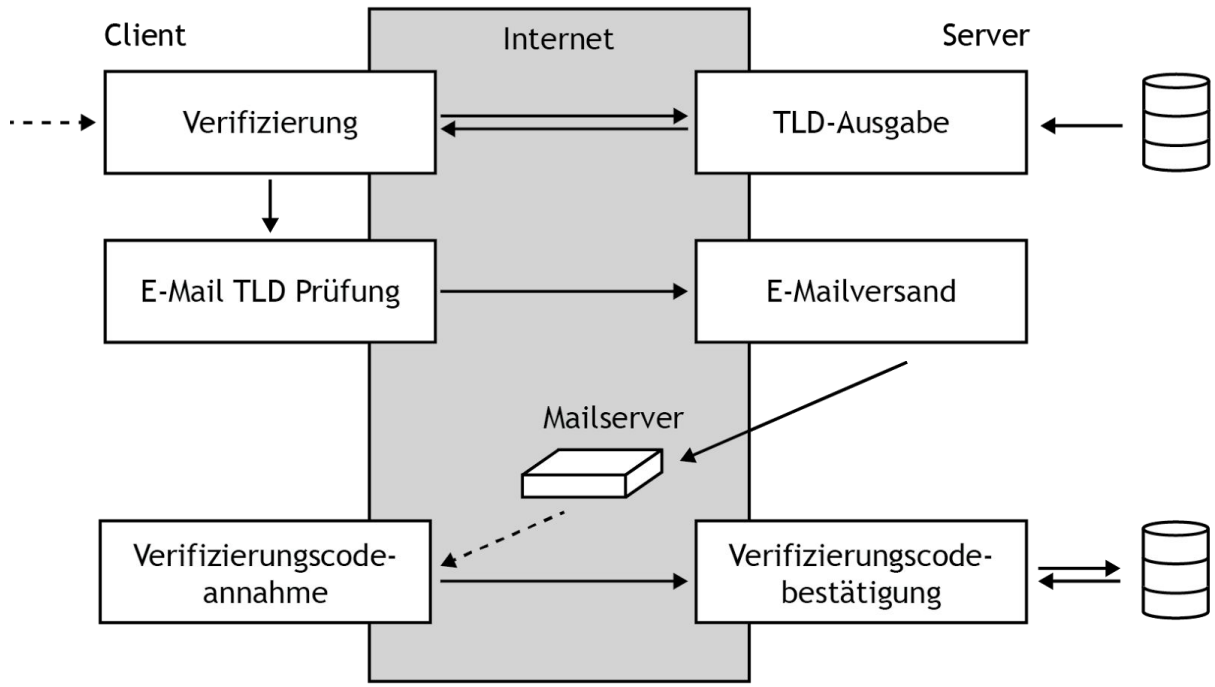
Für die Authentifizierung eines Nutzers wurde ein simples System genutzt. Beim Login sendet der Client Nutzername und Passwort an den Server. Nach der erfolgreichen Authentifizierung am Server wird dem Client ein Schlüssel übermittelt, der zu weiteren Schreib-Interaktionen notwendig ist. Folglich muss der Schlüssel vom Client lokal gespeichert werden. Um dabei Sicherheit zu gewährleisten, müsste die Kommunikation über HTTPS stattfinden, damit die übermittelten Daten verschlüsselt werden. Dies entspricht einer simplen OAuth2.0 Authentifizierung ohne dem Ablaufen des Schlüssels. Der Logout ist somit nur ein löschen des Schlüssels auf dem Client und dem Server.





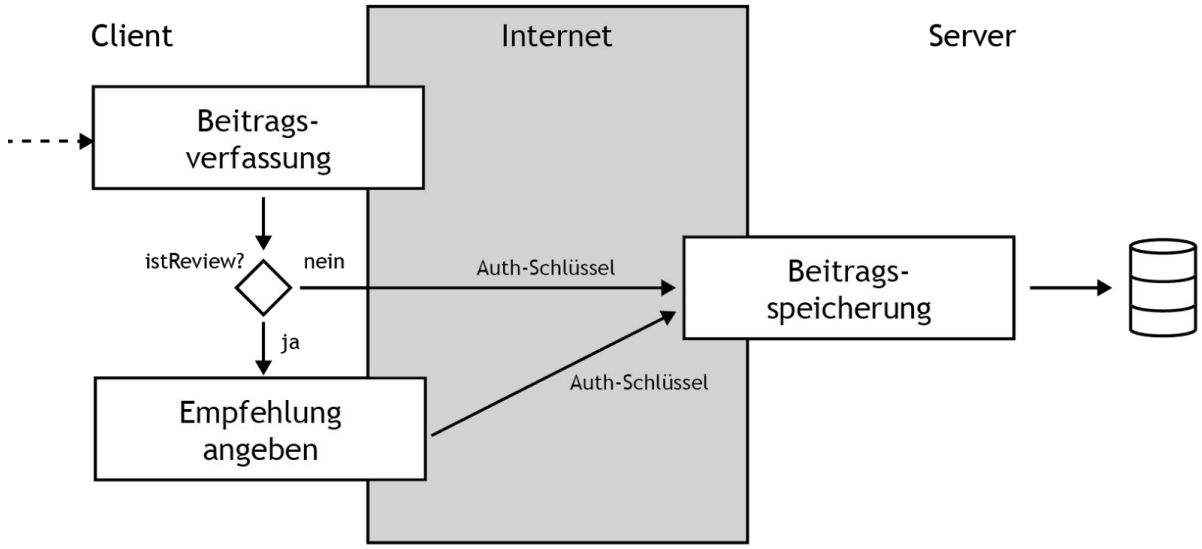
Verifizierung

Die Verifizierung eines Nutzers (also sicherstellen der Immatrikulation eines Studenten) ist die verteilteste Funktionalität der Anwendung. Der Client benötigt dazu vom Server eine Liste von TLDs, die den Hochschulen zugeordnet ist. Nach der clientseitigen Verifikation der Email-TLD wird vom Server eine E-Mail an die angegebene Mailadresse mit einem Verifikations-Link versendet. Dieser Link beinhaltet einen Code und muss von der Client-Applikation aufgerufen werden, um den Code abzufragen und zu validieren. Bei korrektem Code ist der Nutzer verifiziert.



Beitrag verfassen

Bei der Beitragsverfassung wird außerdem angegeben, ob es sich um eine Review oder einen anderen Beitrag handelt. Nur verifizierte Nutzer können eine Review verfassen. In beiden Fällen wird der Beitrag mittels Authentifizierungs-Schlüssel an den Server übergeben.



Kommentar verfassen

Auch das Kommentieren eines Beitrags (oder auch eines Kommentars) benötigt den Authentifizierungs-Schlüssel. Zudem wird eine ID übergeben, auf welchen Datensatz geantwortet wird. Nach erfolgreicher Kommunikation mit dem Server und Speichern des Kommentars wird über Google Cloud Messaging der Verfasser des Eltern-Datensatzes (Eltern-Kommentar oder Eltern-Beitrag) benachrichtigt und kann über die Funktion “Kommentare anzeigen” direkt zum benachrichtigten Inhalt wechseln.