# Dokumentation: Forum-Schnittstelle mit NoSQL MongoDB

Team-Mitglieder:

4649749  
1990138

## Einleitung

Diese Dokumentation beschreibt eine Schnittstelle, die als Datenhaltung eine NoSQL-Datenbank verwendet und dabei den Vorteil einer NoSQL-Datenbank verdeutlicht. Die Schnittstelle wurde für ein Forum entwickelt. Dazu wird als NoSQL-Datenbank die Dokumenten orientierte Datenbank MongoDB verwendet, welche JSON ähnliche Dokumente verwaltet und in C++ geschrieben ist. Diese Dokumentation beschreibt die Architektur und Implementierung der Forum-Schnittstelle.

# Problemstellung

Es soll eine Webanwendung oder Schnittstelle, die als Datenhaltung eine NoSQL-Datenbank verwendet und dabei den Vorteil einer NoSQL-Datenbank verdeutlicht.

# Architektur

Die Architektur der Forum-Schnittstelle basiert auf TypeOrm, einer Objekt-Relationen-Mapping-Bibliothek, die uns dabei hilft, die NoSQL-Datenbank MongoDB mit unserer Anwendung zu verbinden. Die Entitäten der Datenbank sind in einem eigenen Verzeichnis namens ‚src/Entities‘ gespeichert und werden in JSON-Format (anstatt SQL) geschrieben. Die Entitäten Comment und Post dienen als Struktur in der Datenbank.

Zur Verwaltung der MongoDB-Datenbank haben wurde MongoDB-Compass installiert, welches ermöglicht, den Container einfacher als in Docker zu verwalten. So kann die Datenbank einfach durchsuchen, bearbeiten und erweitern werden.

In der Index.js-Datei sind alle Routen für die Schnittstelle geschrieben und die Verbindung zur Datenbank wird hergestellt. Dadurch wird eine nahtlose Integration zwischen Frontend und Backend ermöglicht.

## Vorteile von NoSQL-Datenbanken

## Einführung in NoSQL-Datenbanken

Die traditionelle Auffassung von Datenbanken sind relationale Systeme, die Tabellen verwenden, um Daten zu speichern. Dabei setzen sich komplexe Daten aus Einträgen in mehreren Tabellen zusammen, die mithilfe von Fremdschlüsseln verknüpft sind. NoSQL-Datenbanken bieten hierfür eine Alternative und stellen neue Datenbanktechnologien dar.[[1]](#footnote-1) NoSQL steht für “Not only SQL”. Das bedeutet, dass NoSQL-Systeme die herkömmlichen Eigenschaften von relationalen Datenbanken zwar nicht erfüllen, diese jedoch auch nicht explizit ausschließen. Vielmehr werden relationale Datenbanken als Teilgebiet von NoSQL-Systemen betrachtet.[[2]](#footnote-2) NoSQL-Datenbanken haben verschiedene Vorteile und können Schwachstellen von relationalen Datenbanken beheben.[[3]](#footnote-3) Es gibt verschiedene Hauptuntergruppen von NoSQL-Systemen, die sich vor allem in der Form der Datenspeicherung unterscheiden: Document Stores, Graphen-Datenbanken, Key/Value-Stores und Column Family Systeme.[[4]](#footnote-4) Bekannte Anbieter von No-SQL-Datenbanken sind MongoDB, CouchDB und Cassandra.[[5]](#footnote-5) In diesem Projekt wird ein Document Store von MongoDB verwendet. Dabei werden Daten als Dokumente gespeichert, wobei ein Dokument eine Zusammenstellung strukturierter Daten ist.[[6]](#footnote-6)

## Allgemeine Vorteile von NoSQL-Datenbanken

Ein Vorteil vieler NoSQL-Systeme ist die Möglichkeit, Daten außerhalb des klassischen relationalen Schemas zu speichern. Datenstrukturen müssen somit vor der Datenspeicherung noch nicht im Detail feststehen, wie es bei relationalen Systemen der Fall wäre. Somit können die Daten flexibler gespeichert und auf Änderungen im Laufe der Zeit kann besser reagiert werden. Es gilt jedoch zu beachten, dass auch bei NoSQL-Datenbanken ein gut durchdachtes Datenmodell wichtig ist, um kurz- und langfristig den performanten Umgang mit variierenden Datenstrukturen zu sichern.[[7]](#footnote-7) Ein weiterer Vorteil ist die vereinfachte horizontale Skalierbarkeit, also das Hinzufügen weiterer Server. Dadurch wird zum einen die Ausfallssicherheit erhöht, indem Daten auf mehrere Server repliziert werden. Zum anderen bildet dies einen Kostenvorteil in der Datenverarbeitung gegenüber traditionellen Systemen. Bei Performanceschwierigkeiten in Relationalen Datenbanken ist die Skalierung in horizontaler und vertikaler Richtung nämlich eingeschränkt und teuer. Auch die Flexibilität ist insbesondere bei der Verwaltung großer Datenmengen bei SQL-Systemen eingeschränkt. So gestalten sich Erweiterungen beispielsweise in Form von dem Hinzufügen einer Tabellenspalte als schwierig. NoSQL-Datenbanken hingegen bieten Flexibilität und lassen sich leicht erweitern.[[8]](#footnote-8)

## Projektspezifische Vorteile

Auch bei der im Projekt genutzten Datenbanktechnologie gibt es zahlreiche Vorteile gegenüber relationalen Datenbanken.Es wurde sich bei unserem Projekt für Mongo DB entschieden, da Mongo im Kontext eines Forums die Möglichkeit hat eine unendliche Verschachtelung von Kommentaren darzustellen. So kann einem Mongo Document „Post“ ein nested Document „Kommentar“ angefügt werden, welchem wiederum ein subdocument angehängt werden kann. Dies steht im Kontrast zu allen Relationalen Datenbanken. Darüber hinaus können aufgrund der Flexibilität des Stores schnell Änderungen gemacht werden, ohne dass diese, wie in einem Relationalen Datenbank Modell, alle anderen Daten beeinflussen.

Es gibt aber auch Vorteile, die relationale Datenbanken gegenüber NoSQL-Datenbanken besitzen. Ein Beispiel hierfür ist, dass die Abfrage und Analyse der abgespeicherten Datenobjekte in NoSQL-Systemen mit höherer Komplexität einhergehen.[[9]](#footnote-9) Dies liegt daran, dass die Daten abhängig von bestimmten Applikationen und nicht in starren Formen wie ursprünglich gespeichert werden können. Auch für das Anzeigen von Dashboards, Adhoc-Reports und Analysen sind SQL-Datenbanken immer noch wichtig.[[10]](#footnote-10)

Insgesamt ist zu sagen, dass NoSQL Datenbanken unterschiedliche Vorteile gegenüber herkömmlichen Datenbanken haben. Es kommt aber immer auf die Problemstellung an – je nach Ausgangslage sind die Eigenschaften von NoSQL Datenbanken vorteilhaft oder gar hinderlich. Die Kunst bei der Entwicklung einer Datenbankanwendung ist daher, die passende Technologie für eine spezifische Problemstellung zu wählen.[[11]](#footnote-11) Für das Vorhaben des Projektes war die Entscheidung für eine dokumentenorientierten Datenbank die richtige, da, wie oben schon genannt, für ein Forum die hohe Flexibilität und vor allem die feste Verbindung zu einem übergeordneten Dokument ohne aufwändige Querys von Vorteil ist.

1. (Fasel, 2016) [↑](#footnote-ref-1)
2. (Fasel, 2016), (Müller, 2014) [↑](#footnote-ref-2)
3. (Müller, 2014) [↑](#footnote-ref-3)
4. (Fasel, 2016) [↑](#footnote-ref-4)
5. (Müller, 2014) [↑](#footnote-ref-5)
6. (Fasel, 2016) [↑](#footnote-ref-6)
7. (Fasel, 2016) [↑](#footnote-ref-7)
8. (Müller, 2014) [↑](#footnote-ref-8)
9. (Fasel, 2016) [↑](#footnote-ref-9)
10. (Müller, 2014) [↑](#footnote-ref-10)
11. (Fasel, 2016) [↑](#footnote-ref-11)