In diesem Abschnitt der Arbeit geht es um die Realisierung des Bulletin Boards. Im Folgenden wird der verwendete Tech Stack vorgestellt und die Architektur der Webanwendung in Form eines UML-Diagramms gezeigt. Weiterhin werden Sicherheitsmaßnahem erläutert, die für das Forum implementiert wurden.

Dieses Kapitel vermittelt das technische Wissen der Arbeit. Es werden auch Entwicklungshürden gezeigt und persönliche Evaluation am Ende.

**Tech Stack**

Das Forum wird mithilfe eines MERN Stacks (MongoDB Express React.js Node.js) realisiert. Der Stack ist heute eine beliebte Wahl für die Entwicklung von modernen Single-Page Web Applikationen. Die Technologien, die den Stack bilden, ermöglicht es die gesamte Anwendung in einer einzigen Programmiersprache zu schreiben, nämlich JavaScript.

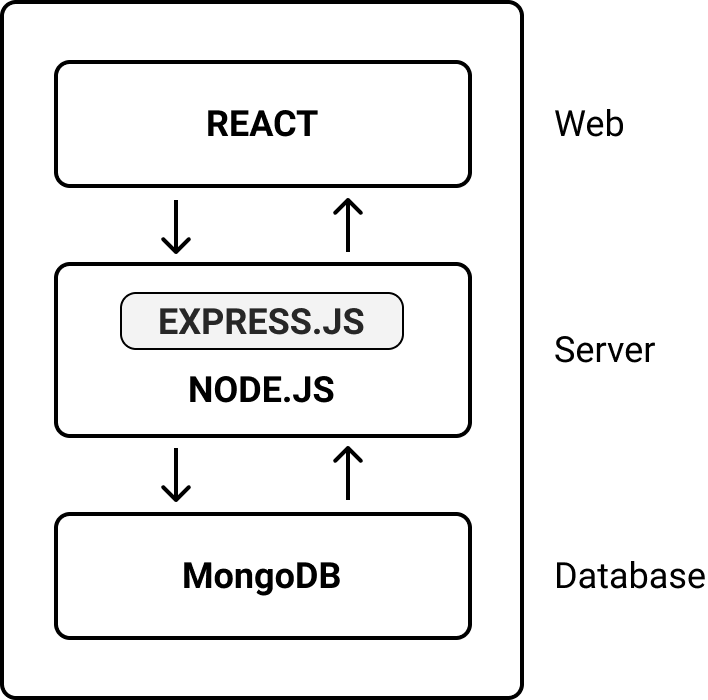
Auch die Datenbank wird mithilfe von JavaScript konfiguriert.

MongoDB ist eine no-SQL Datenbank, die anstatt mit Tabellen (wie in traditionellen SQL-Datenbanken), mit Dokumenten arbeitet. Daten werden hierbei im JSON-Format im jeweiligen Dokument gespeichert. Diese Dokumente werden in Collections gruppiert.

Express ist ein Framework, dass für die Kommunikation mit der Datenbank verwendet wird und Funktionen bereitstellt, um eine RESTful API zu bauen.

Node.js ist eine Laufzeitumgebung für JavaScript. Mithilfe des Node.js Servers wird das Backend der Anwendung hochgezogen. Hier wird der JavaScript Code kompiliert und die Kommunikation zwischen Datenbank und Anwendung geregelt.

**Architektur**



Die Architektur der Anwendung ist nach einem Schicht Model aufgebaut. Genauer genommen eine drei Schichten Architektur. Dabei Bilden React, Node mit Express und MongoDB die drei unterschiedlichen Schichten, wie in Grafik xy dargestellt.

Grafik xy 3-Tier Applikationsmuster1

Eine Schichtarchitektur bringt den Vorteil der Unabhängigkeit mit. Die Schichten arbeiten zwar miteinander, sind aber nicht voneinander abhängig. Jede Schicht hat ihre eigene Logik abgekapselt. So ist React beispielsweise für das Rendering von Daten im Frontend zuständig und sorgt für deren Darstellung. Direkt hat React nichts mit der Datenbank zu tun und spielt die Daten aus, die vom Server geliefert werden. Durch die Unabhängigkeit wird der Austausch oder die Optimieren einer Schicht vereinfacht.

**Node Package Manager (npm)**

Der Node Package Manager ist ein Paketmanager für Applikationen, die mit Node.js laufen. Über das npm repository stehen Entwicklern eine Ansammlung von über 350.000 Paketen zur Verfügung, die sie in ihre Applikation integrieren können. Die meisten Packages sind Open Source und werden Entwicklern über eine freie Lizenz zur Verfügung gestellt.

Für die Entwicklung des Forums werden auch Packages verwendet, um Aufgaben wie Routing und Styling der Applikation zu vereinfachen. Im Folgenden werden die wichtigsten Packages aufgelistet, die in die Applikation integriert werden und ihr Nutzen kurz erklärt.

Frontend

create-react-app

*Dieses package erstellt ein build setup für react apps und ist von React.js ein offiziell*

*anerkanntes package, um einfach und ohne weitere Konfigurationen die Entwicklung an einer React Applikation zu beginnen. Nachteil hier ist jedoch, dass nachträgliche Konfigurationen und Änderungen, z.B. am webpack Modul, erschwert werden.*

react-bootstrap

*Bootstrap ist ein CSS-Framework, dass Klassen bereitstellt, um das Styling von HTML-Elementen zu vereinfachen. Das npm Package bringt React Components mit, die nach bootstrap Manier gestylete Elemente mit React Logik kombiniert.*

react-router *Das react-router Package bietet Navigationskomponenten, die es ermöglichen React Components miteinander zu verlinken, um das Routing in der Applikation zu konfigurieren.*

Backend

express

*Mehr als nur ein Package, ist Express ein essentieller Teil in der Entwicklung von M****E****RN Applikationen. Express ist ein minimalistisches Framework für node.js Applikationen. Es stellt Entwicklern Features zur Verfügung, um eine robuste API für die Kommunikation mit Datenbanken zu bilden. In der Entwicklung des Forum wird Express genau dafür verwendet.*

body-parser

*Ein beliebtes Package, dass eine Middleware bereitstellt, um eingehende Request Bodies zu parsen.*

cors

*Das cors Package stellt eine Middleware bereit, um in Applikationen das sogenannte ‚Cross-origin resource sharing‘ zu konfigurieren. CORS ist ein Mechanismus, der in Web Applikationen, den Zugriff auf Daten von dritten konfiguriert.*

mongoose   
*Ein Package dass Funktionen zur Verfügung stellt, die den Zugriff auf eine MongoDB auf Objekt orientierte Art ermöglichen.*

**Notes**

**Datensicherheit des Forums**

Das Board soll eine offene Kommunikation fördern und es den Usern so einfach wie möglich machen, über ihre Erfahrungen zu sprechen. Auch wenn die Kommunikation anonym ist, die Sicherheit der Daten spielt eine große Rolle. Sowohl für das grundlegende Sicherheitsgefühl der User als auch zum Schutz vor Cyberangriffen.

Das Wissen, dass Einträge nicht außerhalb des Boards einsehbar und geschützt vor Cyberangriffen sind, gibt den Usern ein Gefühl von Sicherheit und kann die offene Kommunikation unterstützen.

Um das zu erreichen, wurden für die Applikation Sicherheitsmaßnahmen implementiert, die im Folgenden beschrieben werden. Die Absicherung von Datenbank und den Einträgen spielt dabei die Zentralle Rolle und bildet die Grundlage der Datensicherheit im Board.

**Sicherheit der Datenbank**

Datensicherheit ist ein grundsätzlich wichtiges Thema für jede Art von Anwendung im Internet, die Daten von Usern sammelt. Zur Implementierung und Auswahl an Sicherheitsmaßnahmen für das Board, bietet das Zweite Kapitel *„Password Encryption, Hashing and Salting“* aus dem Buch *„Identity and Data Security for Web Development“* grundlegende Konzepte und Richtlinien, die für die Arbeit herangezogen wurden. Aus diesem Kapitel stammt auch folgendes Zitat:

“*Lastly, one underlying concept should be understood and implemented. You should store only the minimum amount of sensitive user data that is required to run your application, site, or services. A major industry trend over the past few years has been to capture as much information about your users as possible, store it, and then figure out whether it is useful and is viable at a later date.”*

Nutzerdaten sind kostbar. Aus verschiedenen Gründen sammeln Applikationen im Internet Daten von Usern. Sei es z.B., um Werbungsschaltungen zu optimieren, die Handhabung mit der Anwendung zu vereinfachen oder einfach nur zum Weiterverkauf an Dritte. Daten werden gesammelt, auch wenn nicht immer klar ist wofür.

Je größer also die Menge an gesammelten Daten ist, desto mehr Daten sind auch potentiellen Cyberangriffen ausgeliefert. Das lässt sich aus dem Zitat schließen.

Eine Grundlage für die Sicherheit des Boards wird dadurch gebildet, dass von den Usern keine persönlichen Daten zur Registrierung im Board nötig sind. Nur ein Nutzernamen, der als Psuedonym für die User dienen soll und ein Passwort werden verlangt. Dadurch können die hoch persönliche Informationen, aus den Beiträgen der User, mit keiner Identität verknüpft werden und sind auf diese Weise bereits geschützt. Die Identifikation der User kann jedoch auch über andere Methoden erfolgen. Diese werden im Nachhinein erläutert.

Resting data:

Data Security: Database needs tob e secured, multiple times even

Strong encryption Methods that should be used:

* SHA-256, AES, RSA 🡪 All NIST approved

Best Practises:

* Separate Access Control from DB encryption (multiple security layers)
* Update keys and store outside from data

Data federation as a method to provide information only to needing regions (don’t save everything in a server in the US)

Zitat:

Data in motion:

* Signup information from a user that will be used for account access and identity
* Transmission of profile information to and from service APIs
* Other data collected through the application or website and transmitted for database storage

Possible attacks on databases and their counters

-Brute force (key stretching)

-Dictionary (salting)

-rainbow tables (salting)

-maleware (second auth 2fa)

-Offline cracking (hard salt and good hashes)

Example and Implementation: Hashing and Salting (Node Crypto Library)

* Hash and Salt generated for every user register and password change
* Salt length at least 8bytes better 16 bytes
* Bcrypt library