Erstellt von

183843 – Jan van der Horst  
192495 – Daniel Liebenau  
172458 – Achref Abdelkefi

Webtechnogolien 2

Projekt-Report

**Übersicht**

Das Projekt umfasst eine Web-Anwendung, welche als eine Art Social-Media Plattform für das Austauschen von Nachrichten dient. Es ist möglich, alle Nachrichten ohne vorherige Registrierung einzusehen. Das Hinzufügen von neuen Nachrichten geht jedoch erst nach vorheriger Anmeldung. Ebenfalls können eigene Nachrichten bearbeitet oder entfernt werden.

Priorität war, neben der Implementierung aller Features, die Navigation übersichtlich zu halten und so eine benutzerfreundliche Umgebung zu schaffen.

**Struktur des Projektes**

Die Anwendung folgt einem einfachen 3-Schichten Aufbau und wird unterteilt zwischen der Persistenz-, Business- und Präsentationsschicht. Im Folgenden wird detaillierter auf die einzelnen Schichten eingegangen.

**Persistenzschicht**

Als Persistenzschicht versteht man den Aufbau und die Art der Speicherung von Daten im Backend. In unserem Fall wird trivialerweise Java JPA verwendet. Hierbei ist zu beachten, dass die Datenbankverbindung in einer entsprechenden XML-Datei vorkonfiguriert ist und von dem Anwendungsserver genutzt wird, nicht jedoch von der Webanwendung selbst. Zur Vereinfachung der Datenstruktur und zur Übersichtlichkeit wurde ebenfalls Hibernate implementiert, um objektrelational arbeiten zu können.

**Businessschicht**

Die Businessschicht repräsentiert die REST Anwendung, bei welcher die aus der Vorlesung bekannten Werkzeuge der JAX-RS Spezifikation angewendet werden, um Anfragen von der Webanwendung zu verarbeiten. Zusätzlich werden in einer Konfigurationsdatei beispielsweise der Aktivator gespeichert, der für das JAX-RS Subsystem genutzt wird, oder auch ein Provider für den JSON (de-)serialization Standard.

Außerdem ist es möglich bei Start der Anwendung zusätzliche Daten zu erstellen. Damit dies funktioniert nutzen wir die Enterprise JavaBeans „StartupBean“. Beispielsweise wird der Administrator-Account, sowie ein paar vorkonfigurierte Nachrichten im Startup-Bean erstellt.

**Präsentationsschicht**

Die letzte Schicht in der Struktur der Anwendung ist die Präsentationsschicht. Diese verarbeitet alle Webressourcen und stellt dem Endbenutzer ein „fertiges“ Produkt zur Seite.

Es werden die für das Angular-Frontend benötigten Ressourcen verwaltet.

Während des Developments wird die Präsentationsschicht separat zu den anderen Schichten ausgeführt, um die von Angular bereitgestellten Development-Tools benutzen zu können. So werden beispielsweise alle Änderungen im Frontend direkt neu kompiliert und im jeweiligen Browser angezeigt. Bei dem endgültigen Deployment wird dann das Profil „with-frontend“ aufgerufen, welches alle drei Schichten in eine WAR-Datei packt. Damit ist eine optimale Entwicklungsumgebung gewährleistet und die Implementierung deutlich vereinfacht, da der ganze Deployment-Prozess bei rein frontendseitigen Änderungen wegfällt. Für das Backend gibt es keine gesonderte Development-Umgebung, hier muss für jede Änderung neu kompiliert und der Server neugestartet werden.

**Deployment**

Zur Ausführung sollte die Anwendung mit einem Java EE 8 operierenden Anwendungsserver genutzt werden, der die Datenverarbeitung zwischen Client und dem Webserver verarbeitet und den Content für die Web-Anwendung liefert.

In diesem Fall benutzen wir entweder den Open-Source Applikationsserver Wildfly mit der Version 16.0.0 Final oder das Thorntail-Maven-Plugin, das als unser Webserver fungiert.

Es lassen sich ebenfalls andere Anwendungsserver für die Ausführung des Anwendungsservers nutzen, jedoch wird dafür weite Konfiguration benötigt.

**Angular**

Für die Frontend-Entwicklung haben wir uns für die ebenfalls aus der Vorlesung vorgestellte Open-Source-Software Angular entschieden.

Dadurch, dass das Frontend-Webapplikationsframework dem Entwickler erlaubt, Veränderungen schnell optisch darzustellen, ermöglicht es eine einfache und unkomplizierte Programmierung des Frontends.

Zum einen lässt sich das Angular Framework, wie bereits erwähnt, im Development- oder auch im Production Modus ausführen. Der Development Modus von Angular verschafft uns die Möglichkeit, auftretende Bugs schnell zu erkennen und zu beseitigen, da Fehlermeldungen (Source Maps) erzeugt werden.

Der Production Mode erzeugt eine einzelne .war Datei, die jeweils kompilierten Backend und Frontend-Code enthält. Diese Datei kann zur einfachen Softwareverteilung genutzt werden. Zusätzlich wird im Production Modus „Tree Shaking“ benutzt, um ungenutzten Code zu erkennen und diesen anschließend zu entfernen. Das bringt den Vorteil der Übersichtlichkeit, sowie auch das Angular AoT (Ahead-of-Time) den Nutzen für eine schnelle Entwicklung bietet.

**Apache Shiro**

Das Apache Shiro Framework dient unserer Anwendung zur Autorisierung. Die dafür benötigte Konfigurationsdatei liegt in der Präsentationsschicht des Projekts.

Die Autorisierung wird durch die „ReadNewsItemPermission “ Berechtigung erreicht, welche jedem Benutzer Profil zugeteilt wird. Die Methode implementiert eine weitere Berechtigung und zwar die „ViewFirstFiveNewsItemsPermission“, die eine check-Methode aufruft, mit der das Shiro Permission-System überprüft ob der Nutzer entsprechende Berechtigung hat.

Zur Authentifizierung besitzt das Projekt eine Datenbanktabelle „DBUser“ mit den Spalten Username, Password,Firstname,Lastname,E-mail und isAdmin.

Jedes erstelle Passwort wird mit MD5 verschlüsselt und in der Datenbank gespeichert. Die Benutzernamen, als auch E-Mail Spalten in der Tabelle sind dementsprechend als UNIQUE deklariert. So wird bei einem versuchten Login der Benutzername und das Passwort über POST an die REST API geschickt. Beide werden in der Datenbank abgeglichen. Falls diese gültig sind, wird ein JSON Web Token an den Nutzer zurückgesendet. Bei nachfolgenden Anfragen wird einerseits überprüft ob man als Nutzer authentifiziert ist und ob das erhaltene JSON Web Token gültig ist.

**Aufgetretene Probleme**

Während der Entwicklung der Anwendung sind häufiger Probleme hinsichtlich der Programmierung, als auch Probleme in der Kompatibilität aufgetreten.

Zur Anfangsphase der Projektarbeit gab es Komplikationen mit den unterschiedlichen Entwicklungsumgebungen. Da nicht jeder dieselbe Entwicklungsumgebung zur Programmierung nutzte, sondern zwischen IntelliJ und Eclipse oder auch anderen IDE präferiert wurde, gab es häufig Fehlermeldungen, die sich jedoch nach einiger Recherche im Internet wieder lösen ließen.

Zum parallelen arbeiten an dem Projekt sollte sich jedes Mitglied an ein gemeinsam genutztes GitHub-Repository anbinden. Dies funktionierte relativ reibungslos, im anschließenden Schritt, bei der das angelegte Repository von GitHub geklont werden musste, gab es erneut Probleme bezüglich der einzelnen Entwicklungsumgebungen. Jede IDE hatte andere Funktionen um das Repository zu klonen, so hatte beispielsweise IntelliJ eine eingebaute Funktion um dies über den GitHub-Repository Link zu bewerkstelligen. Allerdings traten ebenfalls Fehlermeldungen auf. Mittels Konsolenbefehle funktionierte das Klonen allerdings reibungslos.

Eine gute Grundlage, wie das Projekt aufgebaut sein soll und mit welchen Komponenten wir das Projekt beginnen mussten, gab das aus der Vorlesung bereitgestellte Beispiel „example06\_app“. Dieses diente uns im weiteren Verlauf als Grundlage für das Projekt.

Dennoch war es schwierig, den ersten Schritt in Richtung Programmierung der Anwendung zu gehen, da man sich unschlüssig darüber war, ob das Frontend oder Backend der erste Startpunkt des Projekts ist, an dem man vorläufig arbeiten sollte. So ist daraus zwar nicht kein ganz paralleler Arbeitsverlauf beider Ebenen entstanden, sondern eher eine abwechselnde Überarbeitung beider.

Weiterhin war man nicht mit allen benötigten Projektkomponenten vertraut. So musste man sich beispielsweise in Angular erst zurechtfinden, auch wenn die in der Vorlesung bearbeiteten Folien dabei gute Anhaltspunkte boten. Dennoch gab es dann projektspezifische Änderungen, die man machen musste, bei denen man sich verzweifelt Hilfe in Foren suchen musste. Die Sprache Typescript war ebenfalls nicht jedem Gruppemitglied vertraut, weshalb man sich in diese vorerst eingelesen hat und daraufhin weiterarbeiten konnte.

Häufige Probleme sind während der Installation von NodeJS aufgetreten. In diesem Fall lag es an der von uns genutzten CSS Alternative Sass. Die für das Projekt benötigten Sass Dependencies wurden bei der Installation ständig ausgelassen und somit konnte man das Angular Framework nicht kompilieren. Dieses oft auftretende NPM Problem wurde schon in vielen Foren besprochen und es wurde nach Problemlösungen gesucht. Eine entsprechende Lösung stellte ein Konsolenbefehl dar, welcher den NPM Paketmanager neu installierte, jedoch optionale Dependencies nicht ignorierte, sodass diese dennoch installiert werden und so die Sass Dependency korrekt installiert wurde. Nach Abschluss wurden alle Dependencies korrekt installiert und somit ließ sich Angular ohne weitere Fehlermeldung ausführen.

Ein immer wieder auftretender Faktor, der das Projekt teilweise stagnieren ließ, war, dass mehrere Personen auf dieselben Dateien im Projekt zugriffen und dort Code programmierten. Allerdings wurde dabei auch oftmals die Struktur des Codes von einem jeweils anderen unbrauchbar gemacht und man im Anschluss die aufgetretenen Änderungen aufsuchen und neu programmieren musste, was nicht zur Entwicklungsgeschwindigkeit beigetragen hatte.

Da unsere Projektgruppe aus vier Mitgliedern bestand, war die Lastverteilung für die Aufgaben relativ ausgeglichen, sodass die verschiedenen Aufgabenfelder gut untereinander aufgeteilt werden konnten. Im späteren Verlauf ist die Gruppengröße auf drei Mitglieder gefallen und eine entsprechende Verteilung war dennoch möglich, allerdings lag nun mehr Last auf jedem Gruppenmitglied, was in Mehrarbeit für die Gruppe resultierte.

**Design**

Bezüglich des Namens haben wir uns entschieden „Drück Dich Aus“ zu verwenden. Dies fanden wir passend, da die Web-Anwendung im Hauptaspekt die Aufgabe hat, Posts zu verfassen und somit seine Meinung an andere kund zu tun.

Folglich startet der Benutzer zu aller erst im Login Fenster, um sich über dieses in der Anwendung anzumelden. Falls vorab noch keine Registrierung erfolgt ist, so können Benutzer intuitiv über die Navigationsleiste einfach in die Registrierung wechseln. Hier kann man dann seine Daten eingeben und sich anschließend über die „Sign Up“ Taste registrieren.

Weiterhin haben wir vor den angezeigten Eingabefeldern Symbole eingefügt um dem Benutzer die Navigation weiter zu vereinfachen. Eine leichte Navigation war ein sehr wichtiger Aspekt, weshalb diesbezüglich Indikatoren neben den Eingabefeldern anzeigen, ob die von dem Benutzer erfolgte Eingabe eine gültige Eingabe ist oder nicht. Auch wenn eine gültige Eingabe vorliegt, so kann trotzdem erstmal nicht gewährleistet werden, ob der eingegebene Nutzername nicht schon vergeben ist. Deshalb wird bei betätigen der „Sign Up“ Taste in roter Schrift ein Fehler angezeigt, falls dies der Fall ist. Hierbei kann die E-Mail, der Nutzername oder auch das Passwort von Betroffen sein. Sind vorhergehende Schritte erledigt, kann der Benutzer wieder über die Navigationsleiste in das Login Fenster wechseln und sich einloggen. Ist dieser eingeloggt, befindet man sich im Home Fenster. In diesem sind wesentliche Bestandteile: zum einen das Nachrichten Fenster, in welchem Nachrichten verfasst werden können und anschließend über die verschicken-Taste veröffentlicht werden können. Das Nachrichten Fenster lässt sich in ihrer Größe individuell nach unten vergrößern, um größere Nachrichten weiterhin komplett zu sehen. Wurde eine Nachricht veröffentlich, kann diese in dem Nachrichten-Feed auf der rechten Seite eingesehen werden. Ein von dem Benutzer selbst erstellter Post wird in einem hell blau hervorgehobenen Fenster erscheinen, um zu verdeutlichen, dass dieser von ihm erstellt wurde. Von Anderen erstellte Posts werden jedoch nicht hervorgehoben, um die Übersichtlichkeit zu bewahren. Außerdem wird jede Nachricht mit einem Zeitstempel versehen, welcher in kleiner Schrift unten links neben der Nachricht angezeigt wird. Natürlich soll es möglich sein seine Nachrichten auch zu löschen oder zu bearbeiten. Dafür werden die oben rechts gesetzten Symbole genutzt. Im Bearbeiten-Modus kann der Benutzer nun seine Nachricht bearbeiten oder den Vorgang auch abbrechen. Wird die Nachricht gelöscht, so erhält der Benutzer eine Benachrichtigung, dass diese entfernt wurde und wird aus dem Nachrichten-Feed gelöscht. Möchte der Benutzer sich ausloggen so, kann er dies über die „Logout“ Fläche in der Navigationsleiste tun.