Service & Cloud Computing WS 2016/17
Dokumentation der Ergebnisse

Gruppe 6: Tim George, Dennis Körte, Sven Zedlick

1. Projekt

Das Thema des Webservice stellt eine Lernplattform in Form eines Karteikartensystems mit Nutzern und Nutzergruppen dar. Dieses Karteikartensystem besitzt als besonderes Feature ein individuelles Lernprofil des Users aus und hilft ihm dabei, besser zu lernen. Weiterhin verfügt der Service über verschiedene Inputmedien, die als Karteikarte verwendet und abgerufen werden können.

2. Vorgehensweise

a. Ideenfindung

Unser erster Schritt war es eine Idee für einen möglichst praktischen, unkomplizierten Web Service zu finden, den wir selbst nutzen würden. Dazu haben wir unsere Gedanken anhand eines Brainstormings gesammelt. Die Idee, die sich am Ende durchsetzte, war ein Lernassistent in Form eines Karteikartensystems mit Nutzerprofilerstellung.

b. Art des Webservice

Im zweiten Schritt einigten wir uns darauf, einen REST Webservice zu erstellen, da REST als einfach zu implementieren gilt und als Zustandsloses Protokoll optimal zur Übertragung unserer Inhalte geeignet ist.

c. Auswahl Technologien

Als grundlegende Programmiersprache fiel unsere Wahl auf Java, da wir alle schon in Java programmiert haben, aber noch keinen Webservice und wir so wieder neue Techniken lernen konnten. Ein weiterer Grund für Java war die hohe Zahl an nützlichen Frameworks. Deren Auswahl entschied sich aufgrund von Arbeitserleichterungen unserer Programmieransätze und Entwürfen zur Datenhaltung und Datenverarbeitung. Zur Datenhaltung verwenden wir eine MySQL Datenbank. Einen genaueren Überblick über verwendete Frameworks und Technologien wird im Abschnitt 4 gegeben.

d. Implementierung des REST Webservices

Als die grundlegenden Entscheidungen getroffen waren, begannen wir mit der Implementierung des Webservice. So konnten wir zur Zwischenpräsentation am 20.12.2016 schon einen lauffähigen Prototypen vorstellen, der bereits über die meisten GET, POST und DELETE Befehle verfügte.

e. Schnittstellenbeschreibung

Die Schnittstellenbeschreibung des Webservice lieferte uns das Framework Swagger. Mithilfe von Swagger Annotationen wurden unsere Methoden erweitert und eine Schnittstellenbeschreibung generiert. Gleichzeitig konnten wir unsere Webserviceabfragen über die Swagger-UI testen.

f. Entwicklung des Webclients

Mit Erhalt der Schnittstellenbeschreibung begannen wir die Umsetzung eines Webclients. Das JavaScript Framework AngularJS half uns dabei, die Abfragen und Antworten des Webservice zu organisieren und erleichtert darzustellen. Mit dem Model View Muster, das AngularJS anbietet, war eine unkomplizierte Einbindung des REST Web Services ohne weitere Schwierigkeiten möglich.

g. Implementierung Security

Als letzte Schwierigkeit galt es, eine Security Komponente in unseren Webservice zu implementieren. Hierfür verwendeten wir das Java Framework Apache Shiro, welches gleichzeitig Passwörter salted, gehashed abspeichert und neben einer Rollenverwaltung von Nutzern auch über eine sehr gute Dokumentation verfügt.

3. Arbeitsteilung

Nachdem wir als Team die grundlegende Idee entwickelt und die zur Umsetzung benötigen Technologien bestimmt hatten, konnten wir uns nun in einzelne Tätigkeitsbereiche aufteilen. Bei der Umsetzung des Webservices muss ganz klar Tim als Hauptanteilsnehmer erwähnt werden. Vor allem für die Erstellung des Datenmodells und des Benutzerprofilmodells ist er fast allein verantwortlich. Sven erweiterte den Webservice um einige Klassen und investierte sehr viel Zeit in die Integration des Security Frameworks. Weiterhin teilte er sich mit Dennis die erstellten Oberflächen des Webservice. Dennis fügte darüber hinaus weite Teile Clients zusammen und integrierte AngularJS und füllte die Oberfläche so mit Daten des Webservice.

4. Technologien / Plattform

- IDE: Eclipse IDE for Java – Jee Neon.1
- Webserver: Apache Tomcat 9
- Webservice:
    - JAX-RS:
        - mappen einer Klasse als Web-Ressource
        - Methoden Parameter
    - Jersey:
        - Referenzimplementierung von JAX-RS
        - JSON Support
    - Hibernate:
        - Persistieren der Daten in MySQL Datenbank
    - Swagger – RESTful API Dienstbeschreibung
    - Apache Shiro:
        - Security Framework
    - MySQL
    - Maven

- Webclient:
    - AngularJS:
        - Webserviceabfragen
        - effiziente Darstellung der Daten
    - BootStrap:
        - UI-Framework zur Contentgliederung

5. Zugriffspunkte

Client:	http://87.190.238.251:8080/flashcards/client/
Swagger-UI:	http://87.190.238.251:8080/flashcards/swagger-ui/

6. Evaluation des Praktikum

Das Arbeiten an der Praktikumsaufgabe in Verbindung mit den Inhalten der Vorlesung hat uns sehr dazu motiviert ein möglichst umfassendes Wissen über die aktuellen Technologien/Frameworks zu erlangen. An der Durchführung des Praktikums haben wir nichts auszusetzen. Auch das selbstständige Aufsetzen eines virtuellen Servers und das Aufspielen unseres Webservers empfanden wir als sehr praxisnah und interessant, obwohl es anfänglich Schwierigkeiten bereitete.

Als einen Kritikpunkt sehen wir, dass nicht klar genug definiert wurde, welchen Umfang die Webservices haben sollen und sich so die Ergebnisse stark im Anspruch unterscheiden. Ein zweiter, vielleicht noch größerer Kritikpunkt ist, dass das Praktikum lediglich als Prüfungsvorleistung gewertet wird und nicht in die spätere Prüfungsbenotung einfließt. Bei einem größeren Einfluss des Praktikums auf die Gesamtbenotung, wäre noch mehr Ehrgeiz in das Projekt geflossen und man hätte seine Arbeit eventuell noch etwas mehr gewürdigt gefühlt.

Insgesamt schließen wir das Praktikum aber mit einem sehr positiven Erlebnis ab und nehmen in jedem Fall viele wichtige Praxiserfahrungen mit, die mit Sicherheit auch späteren Projekten dienlich sein werden.