JUST DO IT
HÖHERE ABTEILUNG FÜR INFORMATIONSTECHNOLOGIE

tgm
Die Schule der Technik

Laborprotokoll

Web Services in JAVA

Systemtechnik Labor
5BHITT 2015/16, Gruppe Z

Simon Wortha

Version 1

Note:

Betreuer: Prof. Borko

Begonnen am 11. März 2016

Beendet am 11. März 2016

Inhaltsverzeichnis

1 Einführung

Diese Übung zeigt die Anwendung von mobilen Diensten in Java.

1.1 1.1 Ziele

Das Ziel dieser Übung ist eine Webanbindung zur Benutzeranmeldung in Java umzusetzen. Dabei soll sich ein Benutzer registrieren und am System anmelden können.

Die Kommunikation zwischen Client und Service soll mit Hilfe von JAX-RS (Gruppe1) umgesetzt werden.

1.2 1.2 Voraussetzungen

- Grundlagen Java und Java EE
- Verständnis über relationale Datenbanken und dessen Anbindung mittels JDBC oder ORM-Frameworks
- Verständnis von Restful Webservices

1.3 1.3 Aufgabenstellung

Es ist ein Webservice mit Java zu implementieren, welches eine einfache Benutzerverwaltung implementiert. Dabei soll die Webapplikation mit den Endpunkten /register und /login erreichbar sein.

Registrierung

Diese soll mit einem Namen, einer eMail-Adresse als BenutzerID und einem Passwort erfolgen. Dabei soll noch auf keine besonderen Sicherheitsmerkmale Wert gelegt werden. Bei einer erfolgreichen Registrierung (alle Elemente entsprechend eingegeben) wird der Benutzer in eine Datebanktabelle abgelegt.

Login

Der Benutzer soll sich mit seiner ID und seinem Passwort entsprechend authentifizieren können. Bei einem erfolgreichen Login soll eine einfache Willkommensnachricht angezeigt werden.

Die erfolgreiche Implementierung soll mit entsprechenden Testfällen (Acceptance-Tests bez. aller funktionaler Anforderungen mittels JUnit)

dokumentiert werden. Es muss noch keine grafische Oberfläche implementiert werden! Verwenden Sie auf jeden Fall ein gängiges Build-Management-Tool (z.B. Maven). Dabei ist zu beachten, dass ein einfaches Deployment möglich ist (auch Datenbank mit z.B. file-based DBMS).

2 Ergebnisse

2.1 Implementierung

GitHub-Repo

https://github.com/swortha-tgm/DezSys-09

Datenbank

Die Datenbankanbindung erfolgt mittels der Spring-Data-JPA. Als Datenbank wird eine eingebettete H2-Datenbank verwendet. Im applications.properties File bewirkt folgende Zeile:

```
spring.datasource.url = jdbc:h2:file:./UserDB;FILE_LOCK=FS
```

dass die Datenbank in einer Datei gespeichert wird.
Das POJO was gespeichert wird ist User und enthält Email, Name und Passowort.

Um auf die Datensätze zuzugreifen bzw. diese auf Objekte zu mappen, wird das Interface UserRepository benötigt.

```
public interface UserRepository extends CrudRepository<User,
String> { }
```

Das Interface CrudRepository enthält die notwendigen Methoden um User zu erstellen und abzurufen.

Rest-Schnittstelle mittels JAX-RS

Für die REST-Schnittstelle wurden drei Endpoints verwendet. /user zum Abfrufen von Benutzern, /login für den Login und /register für die Registrierung. Hier als Beispiel der Userendpoint:

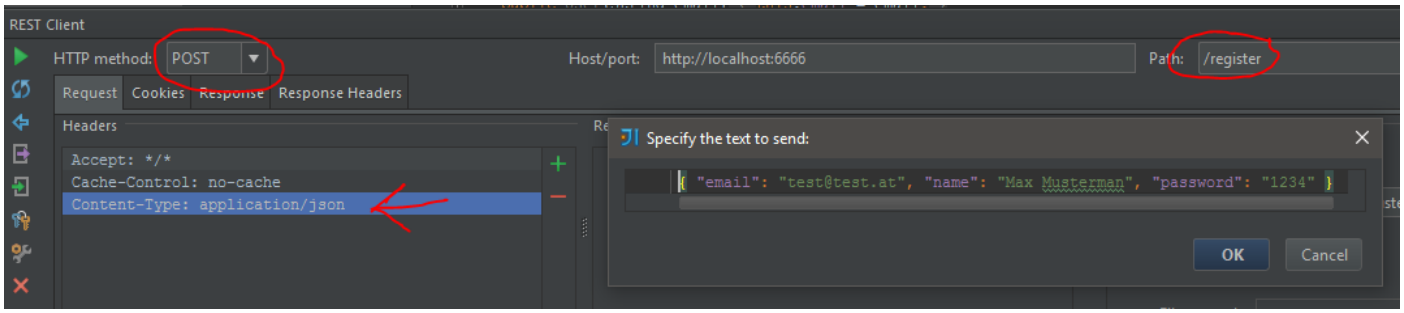
```
    @GET
    @Path("/{id}")
    public Response get(@PathParam("id") String id) {
        User user = this.userRepository.findOne(id);

        if (user == null) {
            return
Response.status(Response.Status.NOT_FOUND).entity("User not found
for id: " + id).type(MediaType.TEXT_PLAIN).build();
        }
        return Response.ok(user,
MediaType.APPLICATION_JSON).build();
    }
```

2.2 Acceptance Test

Mit Hilfe des Intellij-Tools „Test RESTful Web Service" kann das Programm getestet werden.

Zuerst sollte ein User angelegt werden:



Der Body der Nachricht sollte folgender Maßen aussehen:

```
{
 "email" : "test@test.at",
"name" : "Max Musterman",
"password" : "1234"
}
```

Anschließend kommt folgende Rückmeldung:

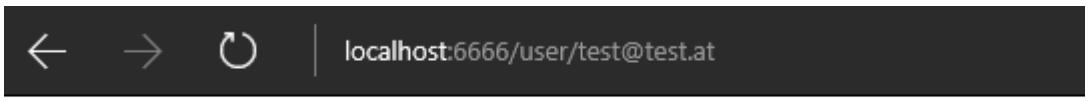
```
User test@test.at saved!
```

(Wenn die Email-Adresse bereits in der Datenbank vorhanden ist, kommt natürlich eine entsprechende Fehlermeldung)

Hier ein Beispiel mit fehlenden Parameter:

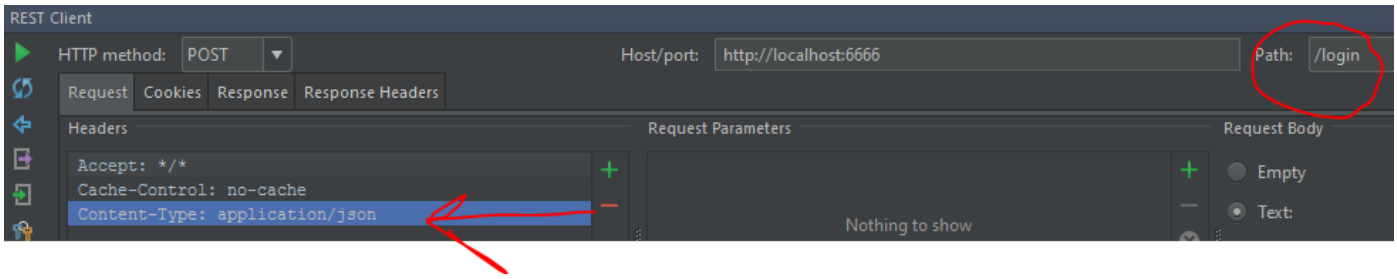
```
Missing Parameter!
```

Wenn der neue User erfolgreich erstellt wurde, kann man mittels einer ID (Email) seine Daten über einen Webbrowser aufrufen:




```
{"email":"test@test.at","name":"Max Musterman","password":"1234"}
```

Wenn dies der Fall ist, sollte sich der neue User auch über /login anmelden können.



Der Body sollte wie folgt aussehen:

```
{
"email": "test@test.at",
"password": "1234"
}
```

Bei richtigen Anmeldedaten, würde das Ergebnis folgendermaßen aussehen:

```
Welcome Max Musterman!
```

Wenn das Passwort jedoch auf „`abcd`" gesetzt wird, und somit ungültig ist, wird folgende Meldung zurückgegeben:

```
Invalid account data!
```

3 Probleme

Bei den herunterladen der Dependancies beim Gradle Projekt, kam es durch das TGM-Internet zu starken Verzögerungen.

4 Feedback

4.1 Lessons learned

Verwendung von JAX-RS in Kombination mit Spring

4.2 Zeitaufzeichnung

11.03.2016	Schule	100 Minuten
12.03.2016	Schule	90 Minuten
14.04.2016	Schule	20 Minuten
Gesamt:		**210 Minuten**

5 Quellen

"Android Restful Webservice Tutorial – Introduction to RESTful webservice – Part 1"; Posted By Android Guru on May 1, 2014; online: http://programmerguru.com/android-tutorial/android-restful-webservice-tutorial-part-1/

"REST with Java (JAX-RS) using Jersey - Tutorial"; Lars Vogel; Version 2.5; 15.12.2015; online: http://www.vogella.com/tutorials/REST/article.html

"O Java EE 7 Application Servers, Where Art Thou? Learn all about the state of Java EE app servers, a rundown of various Java EE servers, and benchmarking."; by Antonio Goncalves; Java Zone; Feb. 10, 2016; online: https://dzone.com/articles/o-java-ee-7-application-servers-where-art-thou

"Heroku makes it easy to deploy and scale Java apps in the cloud"; online: https://www.heroku.com/