
Laborprotokoll

DEZSYS09

**Systemtechnik Labor
5BHITT 2015/16, Gruppe Z**

Jakub Kopec

Version 0.1

Note:

Betreuer: Michael Borko

Begonnen am 26. Februar 2016

Beendet am 4. März 2016

Inhaltsverzeichnis

1 Einführung.....	3
1.1 Ziele.....	3
1.2 Voraussetzungen.....	3
1.3 Aufgabenstellung.....	3
2 Ergebnisse.....	4
2.1 Umsetzung.....	4
2.2 Testung.....	5
Registrierung.....	5
Login.....	7
3 Quellen.....	9

1 Einführung

Diese Übung zeigt die Anwendung von mobilen Diensten in Java.

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+2) umgesetzt werden.

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 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 Datenbanktabelle 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 Umsetzung

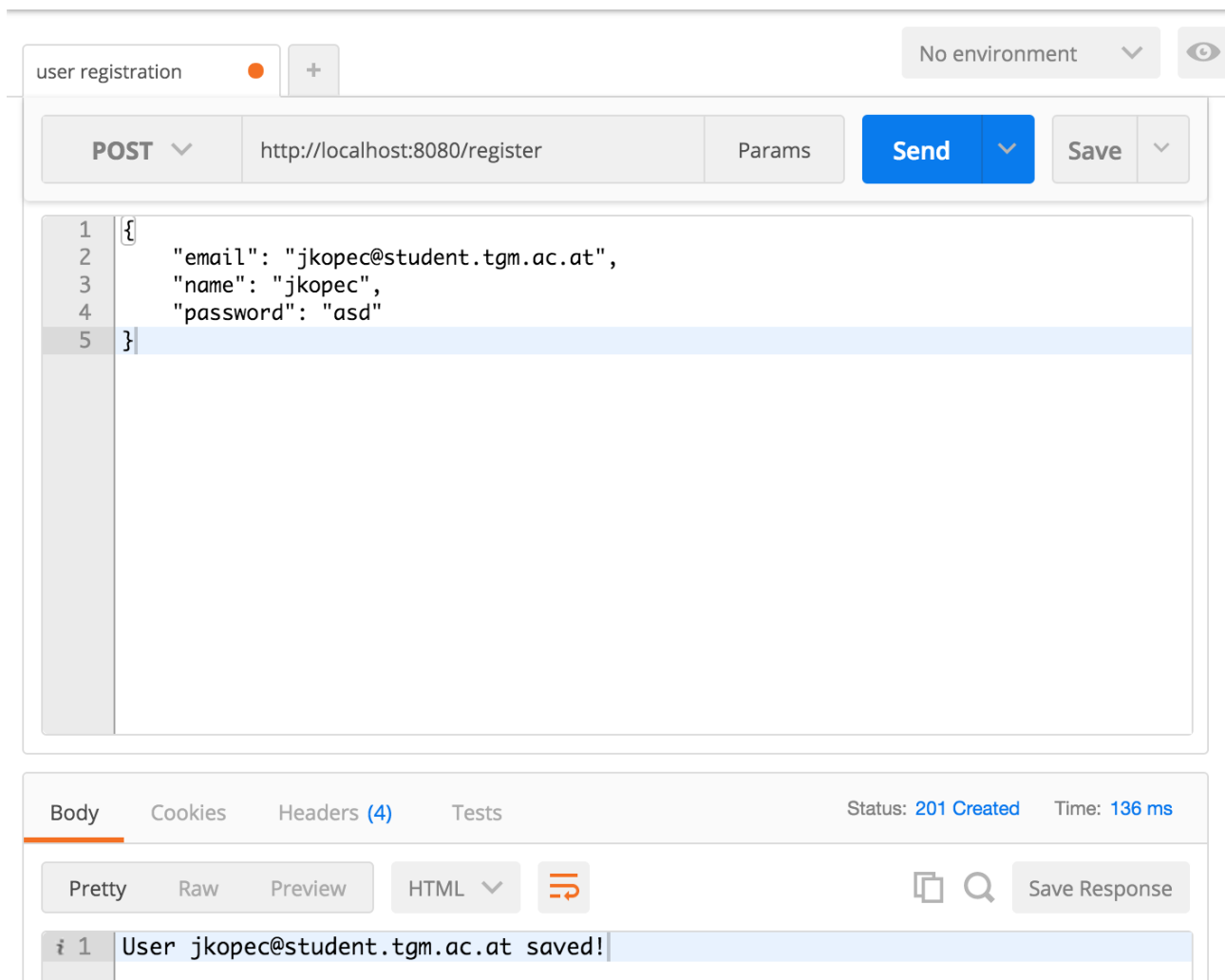
Für diese Labor-Übung wurde als Buildtool Maven verwendet. Das pom.xml – File wurde im Laufe der Implementierung erweitert. Als erstes wurde eine Entität des Benutzers erstellt, welcher sich bei diesem Webservice registrieren und anmelden können soll. Für die weitere Umsetzung der REST-Schnittstelle wurde das Spring-Framework [1] benutzt. Dies bringt gleich mehrere Vorteile. Einerseits bietet Spring ein CRUD-Repository [2] an, was zur Folge hat, dass dieses in dieser Labor-Übung verwendet werden kann und somit die ganze CRUD-Funktionalität nicht selbst implementiert werden muss. Andererseits kann dank Spring Boot [3][5] die Applikation sehr einfach gestartet werden. Als Datenbank wurde H2 [4] gewählt, damit die Abhängigkeit zu externen Datenbanken vermieden wird. Gewählt wurde diese Datenbank aus dem Grund, dass sie im Maven-Repository enthalten ist und auch von Spring ein guter Support angeboten wird. Die Datenbank wird automatisch im Verzeichnis erstellt, in welchem das Programm ausgeführt wird. Sobald die Entität und die Datenbank feststanden mussten lediglich die zwei Endpoints /register und /login ausprogrammiert werden.

2.2 Testung

Bei der Implementierung wurde die Funktionalität mit dem Tool Postman, einer Google Chrome Erweiterung, getestet. In weiterer Folge wurden auch Junit-Tests erstellt.

Registrierung

Wurden alle nötigen Daten angegeben erhält der Benutzer eine positive Rückmeldung.



Will sich ein Benutzer mit einer bereits vergebenen E-Mail registrieren, erhält dieser eine Fehlermeldung.

The screenshot shows a REST client interface with a tab labeled "user registration". The request is a POST to `http://localhost:8080/register`. The request body is a JSON object:

```
{
  "email": "jkopec@student.tgm.ac.at",
  "name": "jkopec",
  "password": "asd"
}
```

The response status is **400 Bad Request** with a time of **9 ms**. The response body, viewed in "Pretty" mode, contains the message:

```
1 User jkopec@student.tgm.ac.at already exists!
```

Login

Versucht sich ein Benutzer mit den richtigen Accountdaten anzumelden, wird er in weiterer Folge willkommen geheißen.

The screenshot displays a REST client interface with a tab labeled "user registration". The request is a POST to `http://localhost:8080/login`. The request body is a JSON object: `{ "email": "jkopec@student.tgm.ac.at", "name": "jkopec", "password": "asd" }`. The response status is `200 OK` with a time of `12 ms`. The response body is `Welcome jkopec!`.

user registration

POST `http://localhost:8080/login` Params Send Save

```
1 {  
2   "email": "jkopec@student.tgm.ac.at",  
3   "name": "jkopec",  
4   "password": "asd"  
5 }
```

Body Cookies Headers (4) Tests Status: 200 OK Time: 12 ms

Pretty Raw Preview HTML Save Response

1 Welcome jkopec!

Gibt der Benutzer jedoch falsche Daten ein, bekommt er eine Fehlermeldung zurück.

The screenshot shows a REST client interface with the following details:

- Environment:** No environment (dropdown)
- Method:** POST (dropdown)
- URL:** http://localhost:8080/login
- Params:** (button)
- Buttons:** Send (blue), Save (dropdown)
- Request Body (JSON):**

```
1 {  
2   "email": "aernhofer@student.tgm.ac.at",  
3   "name": "aernhofer",  
4   "password": "asd"  
5 }
```
- Response Status:** 403 Forbidden (blue text)
- Response Time:** 10 ms (blue text)
- Response Body (HTML):**

```
1 Invalid account data!
```


3 Quellen

- [1] Spring Framework:

<http://spring.io>

[zuletzt abgerufen am 14.04.2016]

- [2] Spring repository resources:

<http://docs.spring.io/spring-data/rest/docs/2.1.5.RELEASE/reference/html/repository-resources.html#repository-resources.collection-resource>

[zuletzt abgerufen am 14.04.2016]

- [3] Spring Boot:

<http://projects.spring.io/spring-boot/>

[zuletzt abgerufen am 14.04.2016]

- [4] H2:

<http://www.h2database.com/html/main.html>

[zuletzt abgerufen am 14.04.2016]

- [5] Spring Bootiful:

<http://spring.io/blog/2014/11/23/bootiful-java-ee-support-in-spring-boot-1-2>

[zuletzt abgerufen am 14.04.2016]