

DezSys07

SOA and RESTful Webservice

Dezentrale Systeme

5BHIT 2015/16

Geyer Stefan, Ritter Mathias

Version 1.0

Betreuer: Prof. Micheler Begonnen am 4. Dezember 2015

Note: Beendet am 17. Dezmber 2015

Inhaltsverzeichnis

[Einführung 3](#_Toc431895375)

[Ziele 3](#_Toc431895376)

[Voraussetzungen 3](#_Toc431895377)

[Aufgabenstellung 3](#_Toc431895378)

[Ergebnisse 4](#_Toc431895379)

# Einführung

Dieses Protokoll-Template soll helfen den Laborübungsteil entsprechend dokumentieren zu können.

## Ziele

Hier werden die zu erwerbenden Kompetenzen und deren Deskriptoren beschrieben. Diese werden von den unterweisenden Lehrkräften vorgestellt.

Dies kann natürlich auch durch eine Aufzählung erfolgen.

## Voraussetzungen

Welche Informationen sind notwendig um die Laborübung reibungslos durchführen zu können? Hier werden alle Requirements der Lehrkraft detailliert beschrieben und mit Quellen untermauert.

Hier zum Beispiel die Architektur der Common Object-Request-Broker Architecture:

## Aufgabenstellung

Hier wird dann die konkrete Aufgabenstellung der Laborübung definiert.

Nun kommt ein Seitenumbruch, um eine klare Trennung der Schülerarbeit zu bestimmen.

# Ergebnisse

## Datenmodell & Generierung der Testdaten

Als Datenmodell wurde ein JSON-Objekt definiert, welches ein Attribut name und ein Attribut description enthält. Diese werden in einer dokumentenorientierten NoSQL Datenbank, MongoDB, abgelegt.

Um eine Million Testdaten zu generieren, wurde der Datengenerator der Seite "GenerateData" verwendet. [1] Dieser muss allerdings heruntergeladen werden, da online maximal 100 Datensätze generiert werden können.

Nach dem Herunterladen benötigt man einen funktionsfähigen Webserver, PHP & MySQL Setup, um das Installationsscript auszuführen. Im ersten Schritt gibt man die Daten der Datenbank an. Falls noch kein User und keine Datenbank erstellt wurde, muss diese vorher noch erstellt werden. Danach baut das Script eine Datenbankverbindung auf. Nun werden die Testdaten in die Datenbank geladen. Im letzten Schritt gibt man Username & Passwort zum Aufruf der Seite an. Danach kann man den DataGenerator wie bei der Online-Version verwenden.

Um die oben erwähnten Daten zu generieren, gibt man nun bei der ersten Spalte als Bezeichnung "name" ein. Als Datentyp wählt man Name. In die Optionsspalte schreibt man "Name Name Surname". In der zweiten Spalte gibt man als Bezeichnung "description" ein und wählt "random number of words" aus. In der Optionsspalte stellt man zwischen 20 und 30 Wörter ein. Zuletzt stellt man 100.000 Datensätze (Maximum) und JSON (Simple) ein. Nun klickt man auf Generate und nach ein paar Sekunden lädt das generierte JSON-File herunter. Diesen Vorgang wiederholt man 10 Mal und fügt danach alle Daten zu einer JSON-Datei zusammen.

Ein Datensatz sieht beispielsweise folgendermaßen aus:

{

"name": "Iris Carly Schroeder",

"description": "dolor. Fusce mi lorem, vehicula et, rutrum eu,

ultrices sit amet, risus. Donec nibh enim, gravida sit amet,

dapibus id, blandit at, nisi. Cum sociis"

}

## Rest-Schnittstelle

## Konfiguration der Vagrant VM

Um das Deployment der Webapplikation und das Hosten von MongoDB einfacher zu gestalten, wurde eine virtuelle Maschine mittels Vagrant konfiguriert.

In der Datei /vagrant/Vagrantfile wurden folgende Grundparameter eingestellt:

* Betriebssystem: Debian/Jessie64
* IP: 192.168.10.200
* Shared Folder, um auf die Testdaten in der VM zugreifen zu können
* Shellscript für Shell-Provisioning unter shell/default

Im Shell-Script wurde dann die weitere Installation/Konfiguration festgelegt, welche nun in den folgenden Unterkapital genauer beschrieben wird.

### Installation & Einrichten MongoDB

Mongo DB wird installiert, indem ein zusätzliches Repository hinzugefügt wird. Danach muss noch konfiguriert werden, dass MongoDB von Außen aufrufbar ist. Dabei wird in der /etc/mongod.conf die IP 0.0.0.0 eingestellt. Um die Änderungen zu Übernehmen, muss MongoDB neu gestartet werden.

### Entpacken & Einspielen der Testdaten

Um die Testdaten zu entpacken, wird 7zip installiert. Zuerst wird die DB, falls vorhanden, gelöscht. Danach wird die Datei entpackt und mittels mongoimport in die Datenbank webappdb gespielt.

### Installation Java & Konfiguration Tomcat

Um die Anwendung später auf die virtuelle Maschine deployen zu können, muss Java installiert und Tomcat konfiguriert werden. Dabei wird zuerst per Oracle Java 8 Installer das Java 8 JDK installiert. Danach wird Tomcat 8 heruntergeladen, entpackt und in das Verzeichnis /usr/share/tomcat verschoben. Bevor Tomcat nun gestartet wird, wird noch ein User angelegt, um mittels dieses Users dann deployen zu können.

## Ausführen

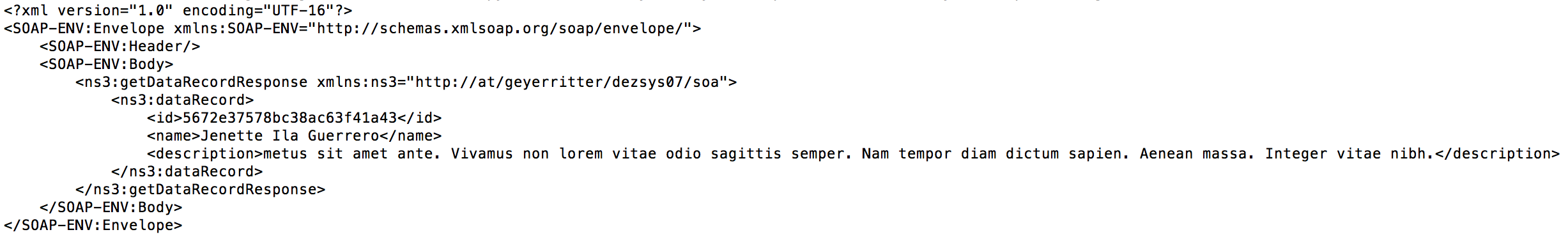
Um die Webapplikation und den SOA-Client auszuführen, muss zuerst eine virtuelle Maschine mittels Vagrant aufgesetzt werden. Dazu muss man VirtualBox, Vagrant und Maven herunterladen und installieren. [3, 4, 5]

1. Vom Root-Verzeichnis des Projekts navigiert man in den Ordner vagrant und führt in der Konsole vagrant up aus.
2. Im Root-Verzeichnis des Projekts führt man in der Konsole mvn clean und mvn install aus.
3. Vom Root-Verzeichnis des Projekts navigiert man in den Ordner web-application und führt mvn tomcat7:deploy aus.
4. Vom Root-Verzeichnis des Projekts navigiert man in saop-client/final und führt die in diesem Ordner enthaltene .jar-Datei in der Konsole mittels java -jar <DATEINAMEN> -h 192.168.10.200 -n <SUCHBEGRIFF> aus.
   1. Ein Output-File kann mittels -o <DATEINAME> angegeben werden
   2. Der Port kann mittels -p <PORTNR> angegeben werden

Bitte beachten: Bei der Suche und der Ansicht auf der Webseite werden immer nur die ersten hundert Treffer angezeigt, da es nicht besonders sinnvoll wäre, tausende von Datensätzen in einem Request zu übertragen bzw. auf der Website darzustellen.

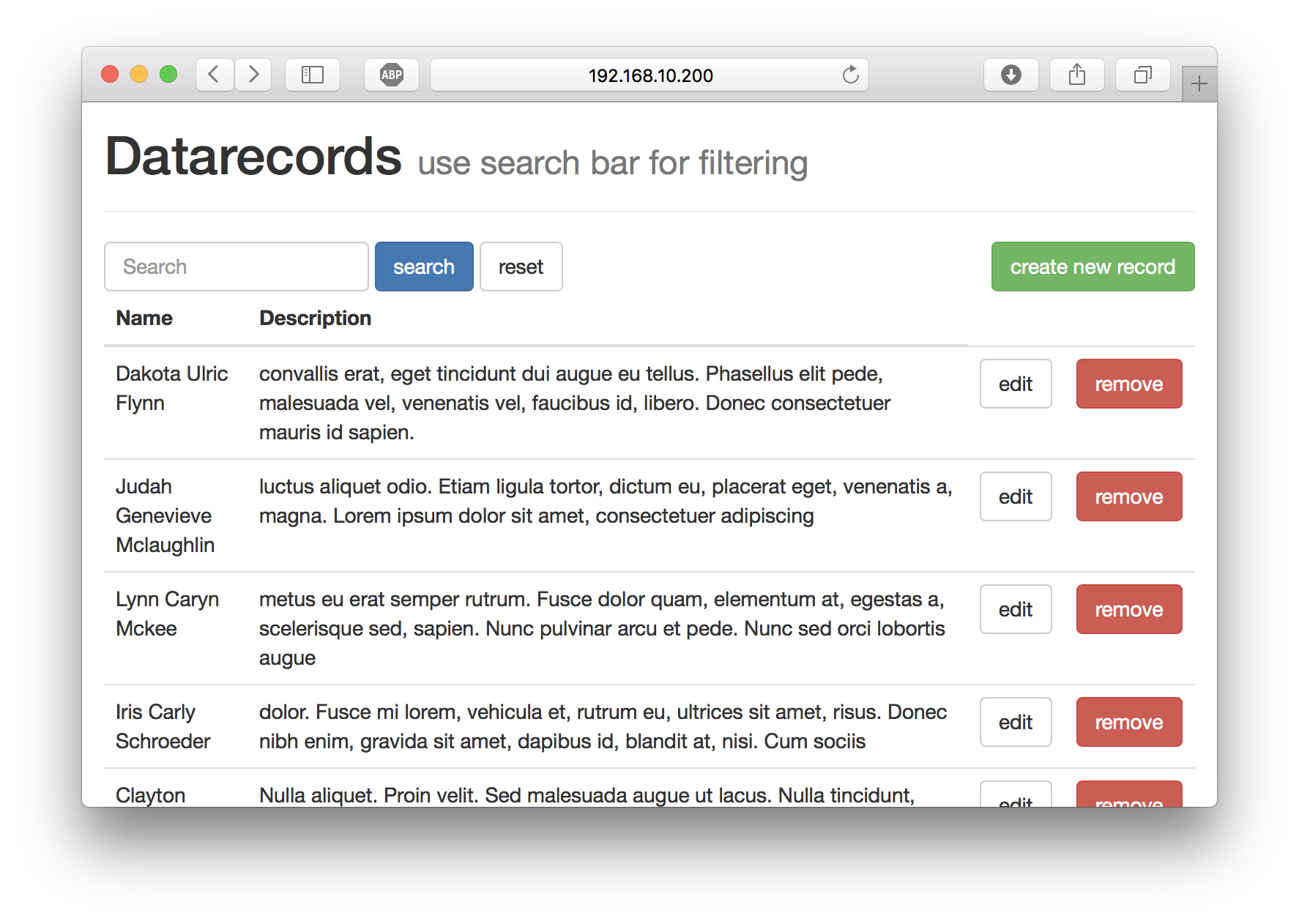
### Gestarteter SOAP-Client

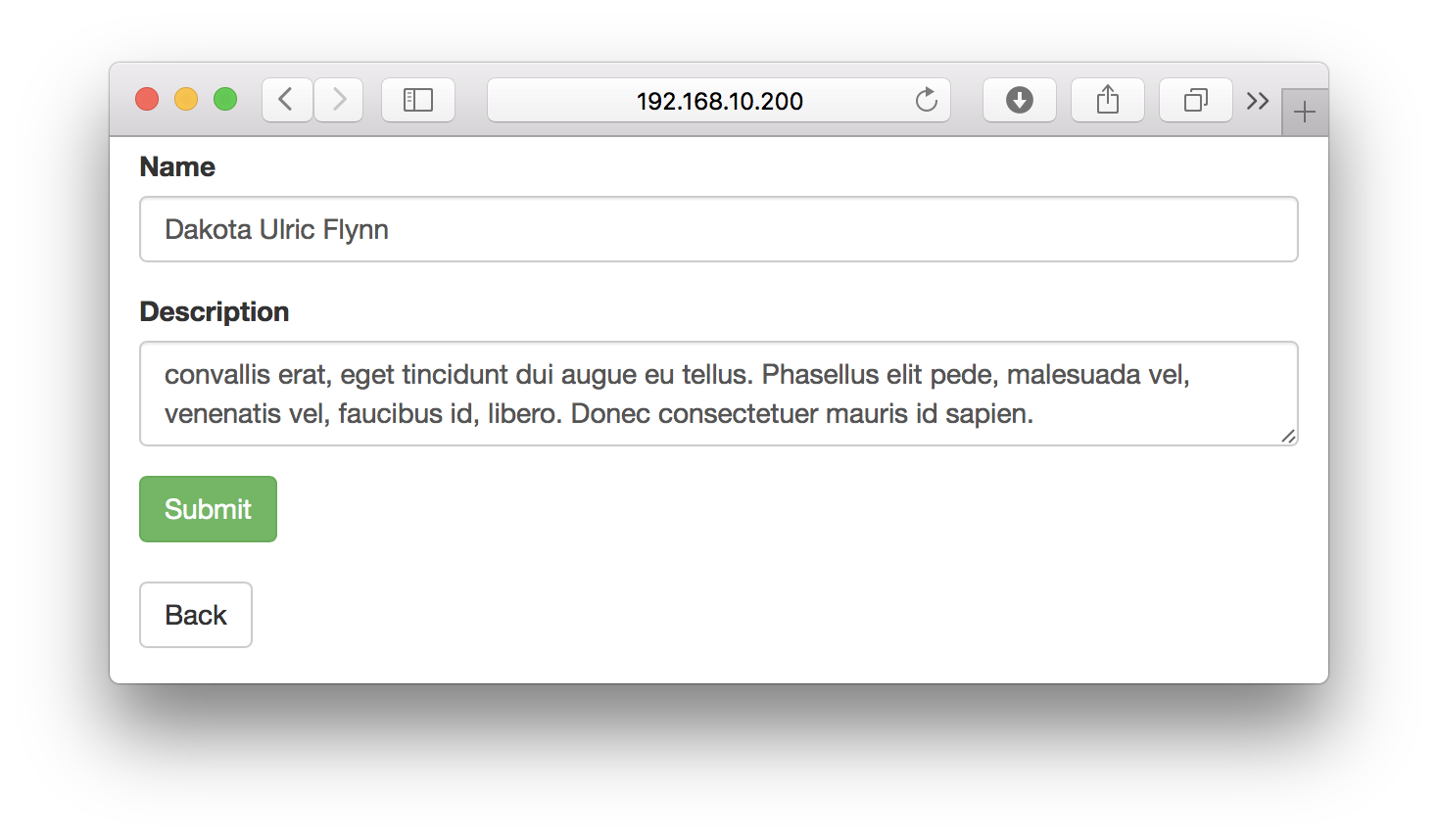
Die Ausgabe des SOAP-Clients sieht beispielsweise folgendermaßen aus:



### Gestartete Web-Anwendung

Die Web-Anwendung wurde bereits nach Schritt 3 auf 192.168.10.200 deployt. Mittels Browser kann man nun unter dieser URL die Webanwendung aufrufen:





# Literaturverzeichnis

[1] Generatedata Daten-Generator, Version 3.2.4 [Online]

Verfügbar unter: <http://www.generatedata.com>

[zuletzt abgerufen am 17.12.2015]

[3] Vagrant, Development environments made easy [Online]

Verfügbar unter: <https://www.vagrantup.com>

[zuletzt abgerufen am 17.12.2015]

[4] VirtualBox, Powerful x86 and AMD64/Intel64 virtualization [Online]

Verfügbar unter: <https://www.virtualbox.org>

[zuletzt abgerufen am 17.12.2015]

[5] Apache Maven Project, software project management and comprehension tool [Online]

Verfügbar unter: <https://maven.apache.org>

[zuletzt abgerufen am 17.12.2015]