Implementierung von REST Services zur Umsetzung der SQLcoach Anwendung

Max Mustermann

Zusammenfassung
Das Abstract ist eine maximal 200 Worte lange Zusammenfassung des Inhalts der Arbeit, so dass sich der Leser vorab ein erstes Bild vom Inhalt machen kann.

Keywords
SQL-Coach, REST, Java, Service

Hochschule Kaiserslautern
Corresponding author: Braun Daniel

Inhaltsverzeichnis

Anforderungen

Entfernen Sie bitte diesen Abschnitt aus Ihrem fertigen Dokument und arbeiten Sie den Inhalt in den Text mit ein. Ihr REST Service soll folgende User Stories abdecken.

1. Der REST-Service liefert auf Anfrage alle Szenarien mit allen Infos zu den Szenarien.
2. Der REST-Service liefert auf Anfrage alle Aufgabengruppen zu einem Szenario mit allen Infos zu den Gruppen.
3. Der REST-Service liefert auf Anfrage alle Aufgaben zu einer Aufgabengruppe mit allen Infos zu den Aufgaben.
4. Der REST-Service erlaubt das Hinzufügen von Szenarien, Aufgabengruppen und Aufgaben.
5. Der REST-Service erlaubt das Ändern der Eigenschafen zu Szenarien, Aufgabengruppen und Aufgaben.
6. Der REST-Service erlaubt das Löschen von Szenarien, Aufgabengruppen und Aufgaben.
7. Der REST-Service erlaubt das Anlegen, Ändern und Löschen von Trainingsdatensätzen mit frei konfigurierbaren Datasources.
8. Der REST-Service erlaubt das Auslesen der Tabelleninformation aller Tabellen in einem Trainingsdatensatz (Tabelleninformation = Tabellennamen, Spaltennamen, Primärschlüssel, Sekundärschlüssel).
9. Der REST-Service erlaubt das ausführung von beliebigen *SELECT* Abfragen auf den Trainingsdatensätzen.
10. Der Personaldatensatz von Prof. Schiefer muss mit eingebunden werden, so dass Ihr Service nach der Installation sofort einsatzbereit ist.

Weitere Anforderungen:

1. Voll dokumentierter Quellcode (jede *PUBLIC* Klasse und Methode hat sinnvollen JavaDoc)
2. Guter Programmierstil (insb. Namensgebung, Funktionen, Objekte, Exceptions, Logging usw.)
3. Zero Warnings bei SonarJava oder Intellij Code Inspection (oder Argumentation warum diese Warnung ignoriert werden kann)
4. Exakte Einhaltung der vorgegebenen REST Schnittstellen
5. Exakte Einhaltung der vorgegebenen maven/ docker Installation
6. Trainingsdatensätze in einer von den Stammdaten unabhängiger Postgres Instanz (innerhalb von Docker).

7. Nachvollziehbare Branches in Ihrem GIT (Namenskonvention: SCS-[NUM]-[feature/ bugfix /doc/ refact/ test]-name)

1. Einleitung

Die Einleitung sollte ausreichenden Hintergrund für den Leser liefern, so dass er sich ohne großes Studium von Sekundärliteratur in das Thema hineindenken kann.

Es sollen hier die grundlegenden fachlichen Begriffe eingeführt werden.

Auch sollte die Motivation für die vorliegende Arbeit dargelegt werden, sowie die Zielsetzung, die man erreichen will. Weiter wird auch das Thema von eventuell verwandten Themen abgegrenzt. Hier kann auch die Literatur [1, 2, 3] vorgestellt werden.

Als Beispiel können Sie hier ein Bild des SQLCoaches einbinden (Abbildung 1).

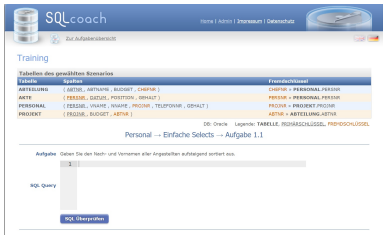

Abbildung 1. Die SQLcoach Anwendung (Quelle `https://sqlcoach.informatik.hs-kl.de/sqlcoach/`).

Folgende Punkte könnten hier angesprochen werden:

- Sinn und Zweck der Anwendung SQLCoach.
- Sinn und Zweck von REST Services.
- Einfüren der wichtigen Begriffe.
- Welche Problemstellung wird hier bearbeitet?

Der SQL-Coach ist eine an der Hochschule Kaiserslautern unter der Leitunf von Dr.Prof. Schiefer entwickelte eLearning Plattform, welche sowohl das Üben als auch das Festigen von SQL-Requests auf einer Trainingsdatenbank ermöglicht. Eine Sql-Request kann beispielsweise so Elemente der Datenbank entfernen oder hinzufügen. Auf der Seite finden sich mehrere Szenarien, Aufgabengruppen und Aufgaben. Eine Aufgabe beinhaltet eine Übersicht der Tabellenstrukturen aller für dieses Szenario relevanter Tabellen eine Aufgabenstellung, ein Feld in welches der Nutzer Query/Request abtippt.(Abbildung2)
Ebenfalls vorhanden sind gewünschte Ausgabe bzw. Musterlösung und die Ausgabe der vom Nutzer definierten Query aufgelistet.(Abbildung3)

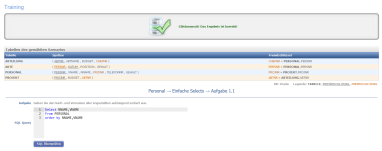

Abbildung 2. Aufgabenstellung und Interface (Quelle `https://sqlcoach.informatik.hs-kl.de/sqlcoach/`).

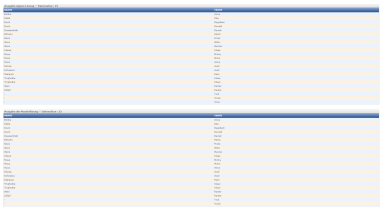

Abbildung 3. Die SQLcoach Anwendung (Quelle `https://sqlcoach.informatik.hs-kl.de/sqlcoach/`).

2. Methoden und Werkzeuge

Kurze Einführung in den Abschnitt. Es wird beschrieben was jetzt kommt, d.h. welche Methoden und Werkzeuge im folgenden vorgestellt werden. Es wird noch nicht darauf eingegangen was im Ganzen damit gebaut wurde. Es werden nur die Details zu den einzelnen Werkzeugen angegeben, so dass im nächsten Kapitel darauf Bezug genommen werden kann.

2.1 JAX-RS und Jersey

JAX-RS wurde im Zuge der Tutorialreihe ”REST Web Services”[1] als Java Programmierschnittstelle kennengelernt. Im Projekt wurden verschiedene Annotationen verwendet, unter anderem:
Pfad Annotationen, welche die Url der Schnittstelle definieren, der Requesttyp, welcher die Art der Anfrage definiert, sowie @Consumes und @Poduces, welche die MIME-Typ der jeweiligen akzeptierten bzw. zurückgegebenen Ressource definieren.

2.2 Docker

Docker ermöglicht die Ausführung von Anwendungen in sogenannten Container/Images, wodurch Portabilität und Kompatibilität garantiert werden. Ironischer Weise ist fehlt diese Kompatibilität bei der Installation von Docker, da Docker auf Linux ausgelegt ist. Um Docker auf Windows laufen zu lassen wird eine Virtualbox oder Hyper-V benötigt.[2] Der Server bzw. die Datenbank läuft in einem Docker Container/Image, welcher mittels ”Docker-compose build bzw.Docker-compose upërstellt Beziehungsweise gestartet wird. Die hierfür notwendigen Informationen finden sich in jewils einer Dockerfile

[1] `https://www.youtube.com/playlist?list=PLqq-6Pq4lTTZh5U8RbdXq0WaYvZBz2rbn`

[2] Hyper-V ist Bestandteil von Microsoft Servern oder Windows 8/10 in der Pro-, Enterprise- und Education-Edition

in docker_postgres und docker_tomcat, sowie in der Dockercompose.yml. Die Dockerfiles besitzen die Syntax :
”FROM postgres:9.6-alpine
Copy SRC_Path DEST_PATH”[3] Kopiert werden hier die Datenbank(en) auf das Host-system.

2.3 Maven

Zwei Sätze zur Einführung der Technologie. Was wurde von dieser Technologie verwendet? Wie wurde von dieser Technologie verwendet? Welche technischen Artefakte (Dateinamen, Konfiguration, Klassen,...) sind entstanden, wie heißen diese und was machen diese.

2.4 Postgres

Postgres wurde als objektrelationales Datenbankmanagementsystem (ORDBMS)”[4] eingeführt und ersetzte somit Oracle SQL Developer, welcher für das Vorbildprojekt genutzt wurde. Die in den sql-Dateien definierte(n) Datenbank(en) operiert mittels Postgres, weshalb auf der Datebank ausgeführte Queries auch als PostgreSQL statement formuliert sind. Die verwalteten Stammdaten befinden sich in der init_database.sql Datei.

2.5 Apache Log4j

Zwei Sätze zur Einführung der Technologie. Was wurde von dieser Technologie verwendet? Wie wurde von dieser Technologie verwendet? Welche technischen Artefakte (Dateinamen, Konfiguration, Klassen,...) sind entstanden, wie heißen diese und was machen diese.

2.6 GIT

Git ist eine Filehosting-Service, der es Nutzern erlaubt hochgeladenen Code oder Code-snippets anzusehen oder diesen herunterzuladen. Auf diese weise ist es Programmierern möglich im Team an Sourcecode(-segmenten) zu arbeiten Zwei Sätze zur Einführung der Technologie. Was wurde von dieser Technologie verwendet? Wie wurde von dieser Technologie verwendet? Welche technischen Artefakte (Dateinamen, Konfiguration, Klassen,...) sind entstanden, wie heißen diese und was machen diese.

3. Ergebnisse

In diesem Abschnitt beschreiben Sie nun das Endergebnis Ihrer Arbeit.

3.1 Architektur

Die beziehen sich hier auf die im Methodenteil vorgestellten Artefakte und beschreiben wie diese zusammengesetzt wurden. Welche Komponenten ruft welche Komponente auf?

Sie müssen hier auf jede der Methoden in Abschnitt 2 Bezug nehmen.

[3] urlhttps://docs.docker.com/engine/reference/commandline/cp/
[4] http://www.postgresql.de/index.whtml

3.2 REST Schnittellen

Implementiert wurden, die im folgenden genauer beschriebenen Methoden, welche das Hinzufügen,Updaten und Löschen eines Szenarios, einer Aufgabengruppe oder einer Aufgabe ermöglichen. Des Weiteren ist es möglich Zusatzinformationen über die implementierten Tabellen im Trainingsdatensatz zu erhalten, oder Queries auf diesen auszuführen.

Get all scenarios
http://localhost:8001/sqlcoachservice/api/v1/catalog
Bei Aufruf der Methode getScenarios() / Aufruf der Url gibt der Restservice eine Übersicht der Szenarien in JSON-Strutur zurück. Im JSON beinhaltet sind die Informationen auf welchem dataset das Szenario operiert, Rang und ID, sowie der Name des Szenarios und dessen Besitzer.

Get all groups for given scenario
http://localhost:8001/sqlcoachservice/api/v1/catalog
/scenarioId
Bei Aufruf der Methode getGroups() / Aufruf der Url gibt der Restservice eine Übersicht der Aufgabengruppen, welche Bestandteil des passenden Szenarios sind, in JSON-Strutur zurück. Durch den Pathparameter catalog/scenarioid ist eindeutig, dass die Aufgabengruppen des korrespondierenden Szenarios ausgegeben werden sollen. Nur die Aufgabengruppen eines Szenarios können zurückgegeben werden, weil die scenarioid Foreign KEY ist. Die JSONS beinhalten die ID der Gruppe und die Id des Parent tables, sowie die Name der Aufgabengruppe.

Get all exercises for given group
http://localhost:8001/sqlcoachservice/api/v1/catalog
/scenarioId/groupId
Bei Aufruf der Methode getExercises() / Aufruf der Url gibt der Restservice eine Übersicht der Aufgaben, welche Bestandteil der passenden Aufgabengruppe sind, in JSON-Strutur zurück. Durch die Pathparameter (”catalog/scenarioId/groupId”) sind die Foreign Keys angegeben, wodurch die Schnittstelle die gewünschten Aufgaben eindeutig identifizieren und ausgeben kann. Die JSONS beinhalten den Primary Key exercise ID, den foreign Key groupID, scenarioId, sowie Aufgabenbeschreibung und Aufgabenlösung.

Add scenario
http://localhost:8001/sqlcoachservice/api/v1/catalog/add
Bei Aufruf der Methode addScenario() / / Ausführen einers ADD-Befehls (z.B. per Postman) auf dieser Url fügt die Rest-Schnittstele, sofern ein geeignetes JSON übergeben wurde, das Szenario der Datenbank hinzu, anschließend führt er getScenario() aus und gibt somit eine Übersicht aus in der das neue Szenario vorhanden ist.

Add group for given scenario
http://localhost:8001/sqlcoachservice/api/v1/catalog
/scenarioId/add

Bei Aufruf der Methode addGroups() / Ausführen einers ADD-Befehls (z.B. per Postman) auf dieser Url fügt die Rest-Schnittstelle, sofern eine geeignete ruppe im JSON-Format übergeben wurde, die Gruppe in das durch die Pathparameter spezifizierte Szenario ein und gibt anschließend eine Übersicht der Aufgabengruppen im Szenario aus.

Add exercise for given group
http://localhost:8001/sqlcoachservice/api/v1/catalog /scenarioId/grouId/add
Bei Aufruf der Methode addExercise() / / Ausführen einers ADD-Befehls (z.B. per Postman) auf dieser Url fügt die Rest-Schnittstelle, sofern eine geeignete Aufgabe im JSON-Format übergeben wurde, diese in die durch die Pathparameter eindeutig bestimmte Gruppe ein und gibt anschließend eine Übersicht der Aufgaben dieser Gruppe zurück.

Update scenario
http://localhost:8001/sqlcoachservice/api/v1/catalog /scenarioId
Bei Aufruf der Methode updateScenario() / Ausführen einers PUT-Befehls (z.B. per Postman) auf dieser Url ändert die Rest-Schnittstelle, sofern ein geeignetes Szenario im JSON-Format übergeben wurde, das durch den Parameter scenarioId definierte Szenario den übergebenen Werten entsprechen ab. Ruft der Nutzer beispielsweise
http://localhost:8001/sqlcoachservice/api/v1/catalog/1 mit der Übergabe folgenden JSONS auf:
{
“scenarioName”: ”Neuer Szenarienname”,
“scenarioOwner”: ßcenarioOwner nach update ”,
“datasetId”: 2
}
wird der scenarioName des Szenarios mit der scenarioId 1 auf Neuer Szenarienname, scenarioOwner auf scenarioOwner nach update und die datasetId auf 2 geupdated/abgeändert.

Update group
http://localhost:8001/sqlcoachservice/api/v1/catalog /scenarioId/groupId
Bei Aufruf der Methode updateGroup() / Ausführen einers PUT-Befehls auf dieser Url ändert die Rest-Schnittstelle, sofern eine geeignete Gruppe im JSON-Format übergeben wurde, die durch die Pathparameter definierte Gruppe den übergebene Werten entsprechen ab.

Update exercise
http://localhost:8001/sqlcoachservice/api/v1/catalog /scenarioId/groupId/scenarioId
Bei Aufruf der Methode updateExercise() / Ausführen einers PUT-Befehls auf dieser Url ändert die Rest-Schnittstelle, sofern eine geeignete Aufgabe im JSON-Format übergeben wurde, die Werte der Aufgabe dem JSON entsprechend ab.

Delete scenario
http://localhost:8001/sqlcoachservice/api/v1/catalog/scenarioId
Bei Aufruf der Methode deleteScenario() / Ausführen einers DELETE-Befehls auf dieser Url wird das durch scenarioId defninierte Szenario gelöscht.

Delete group
http://localhost:8001/sqlcoachservice/api/v1/catalog /scenarioId/groupId
Bei Aufruf der Methode deleteGroup() / Ausführen einers DELETE-Befehls auf dieser Url wird die durch grouId defnierte Aufgabengruppe gelöscht.

Delete exercise
http://localhost:8001/sqlcoachservice/api/v1/catalog scenarioId/groupId/exerciseId
Bei Aufruf der Methode deleteExercise() / Ausführen einers DELETE-Befehls auf dieser Url wird die durch exerciseId defninierte Aufgabe gelöscht.

Get table definitions
http://localhost:8001/sqlcoachservice/api/v1 /dataset/dataId/tables
Das ausführen dieses Befehls gibt alle Relevanten Tabelleninformationen(Tabellen, Spalten,Primary und Foreign Keys) des Datensets mit korrespondierender dataId aus.

Execute query
http://localhost:8001/sqlcoachservice/api/v1 /dataset/dataId/execute?query= *query*
Sofern hier anstelle des kursiven query eine geeignete Query steht, so wird diese auf dem durch dataId definierten Datenset ausgeführt.

3.3 Installation

Geben Sie hier eine Installationsanleitung. Ihre Software muss Sie wie folgt bauen und starten lassen:

```
git clone MYREPO

mvn clean package

docker-compose  build

docker-compose  up
```

4. Diskussion

In diesem Abschnitt werfen Sie einen kritischen Blick auf Ihre Arbeit:

1. Haben Sie alle angeforderten User Stories umgesetzt?
   Alle erforderlichen User-Stories wurden implementiert und wiederholt auf Funktionalität geprüft.

2. Welche Features haben sie zusätzlich umgesetzt?

3. Welche positiven Eigenschaften hat Ihre Implementierung?
   Vorteil der vorgestellten Implementierung von Execute Query ist, dass sie nicht nur Selects zulässt auch Delete Befehle können auf der Datenbank ausgeführt werden, weil diese aber keine Rückgabe haben erhält man eine Fehlermeldung(cause: Die Abfrage lieferte kein Ergebnis., errorCode: 500, errorMessage: Failed to execute Query,).
   Get Table Definitions gibt zwar die gewünschten Informationen in JSON-Format zurück, dieses kann aber von der Erwartung abweichen, weil sowohl die Fremdschlüssel, als auch die Spaltennamen lediglich als Liste innerhalb des JSONS zurückgegeben werden. Vorteil dieser Implementierung ist, dass keine IF-Abfragen oder Switch-Cases den Rückgabetyp bestimmen müssen und ebenfalls keine zusätzlichen Klassen für jede im Nachhinein eingefügte Spalte erstellt werden müssen. Statt dieser Klassen ist nur die eine Klasse Table_shemas implementiert. Die etwas unschöneren JSONS wurden in Kauf genommen, weil man auf diese Weise einen kürzeren Code erhält und eine einfachere Erweiterbarkeit der Datenbank.

4. Welche negativen Eigenschaften hat Ihre Implementierung?

5. Sind die Schnittstellen gut gelungen? Warum Sind das gute Schnittstellen?

6. Wie performant ist Ihre Implementierung?

7. Welche Best Practices haben Sie eingehalten?

8. Wie hoch ist Ihre Test Coverage (JUnit)?

9. Wie ist die Code-Qualität? Berichten Sie hier über das Ergebnis der Code-Inspection (SonarJava oder IntelliJ IDEA Code-Inspection). Die IntelliJ IDEA Code-Inspection gibt mehrere zu ignorierende Fehler aus:
   Unused declaration

Dieser Abschnitt ist nicht leicht zu schreiben, aber dieser Abschnitt ist essentiell für Ihre Benotung. **Sie müssen hier das Positive herauszustellen!** Zudem sollen Sie hier auch auf Unzulänglichkeiten Ihrer Arbeit eingehen. Wenn Sie die Schwachstellen Ihrer Arbeit hier beläuchten und idealerweise Argumente finden warum das so gelöst wurde, dann hat einen sehr positiven Einfluss auf die Benotung.

5. Zusammenfassung

Hier wird nochmal der Inhalt und die Ergebnisse der Arbeit kurz zusammen gefasst. Des Weiteren werden Themen und Aufgabenstellungen genannt, die es lohnt weiter zu untersuchen.

Literatur

[1] David Harel. Statecharts: A visual formalism for complex systems. *Sci. Comput. Program.*, 8(3):231–274, June 1987.

[2] D. Harel, Y. Feldman, and M. Krieger. *Algorithmik*. Springer Berlin Heidelberg, 2006.

[3] Jilles Van Gurp and Jan Bosch. On the implementation of finite state machines. In *in Proceedings of the 3rd Annual IASTED International Conference Software Engineering and Applications, IASTED/Acta*, pages 172–178. Press, 1999.

Erklärung zur Ausarbeitung

Hiermit erkläre ich, Daniel Braun(880335), dass ich die vorliegende Ausarbeitung selbstständig und ohne fremde Hilfe angefertigt habe und keine anderen als in der Abhandlung angegebenen Hilfen benutzt habe; dass ich die Übernahme wörtlicher Zitate aus der Literatur sowie die Verwendung der Gedanken anderer Autoren an den entsprechenden Stellen innerhalb der Arbeit gekennzeichnet habe. Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben kann.

_____

Unterschrift // Notes: abgabe über olat digital gedruckte fassung ebenfalls nötig Begriffe einführen

Behauptungen untermauern keine blümchenwörter Code zählt nicht als text keine website sondern ne webbasierte anwendug wurde erstellt

ziel: moderne Technologien (wie jax rs) visuell ansprechend Zusammenfassugn 200 wörter zusammenfassug der arbeit Einleitung:wann wurde sql coach ersstellt 2007 unter leitung prof. schiefer ist veraltet es wird bsp das mvc Framework apache struts keine website sondern ne webbasierte anwendug wurde erstellt http standartmethoden, get post put delete

Jax rs ist eine java api spezifikation,"welche die Rest architektur bereitstellt und seit Version 6 offiziell bestandteil der Java EE Pakete ist."Jersey(referenz) implementiert die Jax-RS Schnittstelle.

dir in latex mit
dirtree

.1SQL-coa... .2..... .3etc Docker: Anwedunug werden isoliert und laufen in container werden aud f anderen Rechenrn beeitgestellt referenz zu docker

werkzeuge: Scöpfer und baujahr

dockerfile:Dieses Dockerfile spezifiziert den PHP Webserver Apache 7.2.siehe section**??** szenarien gezielt eine topic

vermittelnd ding lehrend. ”Die Informationen zu Szenarien–....datensätzen datasources werden in der sogenannten catalog datenbank abgelegt, Zusätzlich gibt es den begriff der trainingsdatenbank welche datensätze für die ausführnung der sql abfragen enthalten

[(docker compose eigene db 8004:80)] am ende der einleitung: was in wlechem kapitel ist 2:in diesem abschnitt werde die genutzten Methoden und Werkzeuge erklärt.sql coach läuft af max db wurde in postgres überführt.

2:

–yyyxyxyxy