

Go Microservices 1 Assignment Write-up

Submitted by: Tho Yan Bo

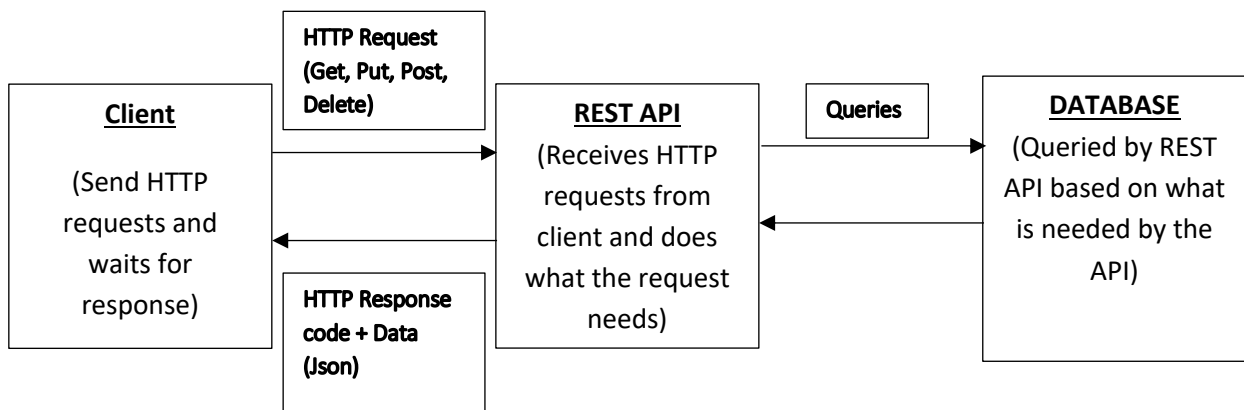
Table of Contents

1. Assignment Overview	3
2. Deploying of application	4
2.1 Deploying of MySQL database.....	4
2.2 Starting the REST API	5
2.3 Starting the client console application.	6
3. Test cases for the console application	6
3.1 Get all courses	6
3.2 Get specific course	6
3.3 Create a new course.....	7
3.4 Update course.....	8
3.5 Delete course.....	9
4. Additional implementations for this assignment.....	9

1. Assignment Overview

For this assignment, a total of three components were implemented to demonstrate the interactions between a 1) a REST API, 2) MySQL database and 3) Client application.

The client console application will take in user input and send a HTTP request in the form of (Get, Post, Put or Delete) to the REST API. The REST API server listens for such HTTP requests and when client HTTP requests are received, it will make subsequent calls to the database to perform any necessary 'CRUD' (Create, Update, Delete or Read) operations. In all cases, a HTTP response and response code will be returned to the client. Additionally, if required, the REST API will return Json data back to client console (in the case of GET request).



The database for this assignment is a MySQL database deployed on docker container. There are prepared scripts within the assignment folder that can be mounted inside the official MySQL docker container. By following the setup guide in the following section, a table (named "CourseInfo") and its prepared data will exist within the running container database (named "goMS1_db").

```

mysql> USE goMS1_db;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * FROM CourseInfo;
+-----+-----+-----+-----+-----+
| Code | Title | Dates | Lecturer | Description |
+-----+-----+-----+-----+-----+
| 1 | Go Basic | 13th - 15th Jan 2021 | Ching Yun Lee | Basic Programming Knowledge about Golang |
| 2 | Go Advanced | 28 Jan - 3 Feb 2021 | Ben Low | Learn advanced concepts in Go programming such as as packages, creation of data structures and error handling mechanism. |
| 3 | Go In Action 1 | 24-26 Feb 2021 | Ben Low | Explore the practical aspects of Go software development in detail. |
| 4 | Go In Action II | 17-19 March 2021 | Ching Yun Lee | A course about Go Security Best Practices |
| 5 | Go MicroService I | 5 - 7 April 2021 | Ching Yun Lee | Introduction to Microservices, APIs, containerization and microservice frameworks |
+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
  
```

2. Deploying of application

From Tho_Yan_Bo_MS1.zip, unzip the files to a local directory.

2.1 Deploying of MySQL database

The MySQL database will be deployed on a docker container. If you do not have docker installed, go to <https://docs.docker.com/docker-for-windows/install/> and download Docker for the OS you are using.

Install docker and check version of docker installed to verify installation.

Getting Version of Docker

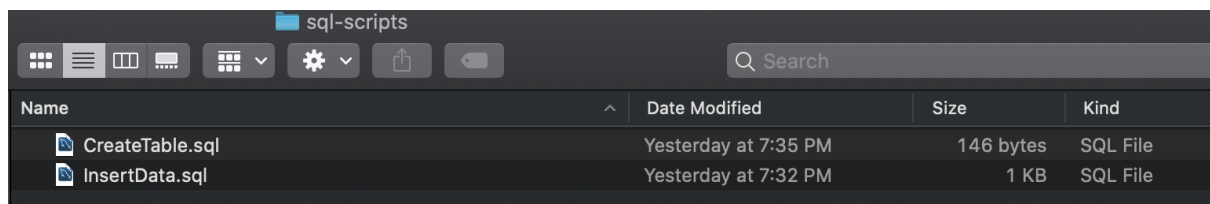
```
$ docker version
Client: Docker Engine - Community
Version: 19.03.8
API version: 1.40
```

We will be using SQL scripts to contain the SQL statements which we want to implement on the container database. Create a directory in which we will work. Also, create a subdirectory to store our .sql scripts using the following command:

```
mkdir -p ~/my-mysql/sql-scripts
```

```
$ mkdir -p ~/my-mysql/sql-scripts
```

Move the two files **CreateTable.sql** and **InsertData.sql** into this newly created directory.



Name	Date Modified	Size	Kind
CreateTable.sql	Yesterday at 7:35 PM	146 bytes	SQL File
InsertData.sql	Yesterday at 7:32 PM	1 KB	SQL File

These files will help to load the table (CourseInfo) and course datas into the docker database later.

Next, in terminal let us create a MySQL container with the following command:

```
docker run -d -p 52364:3306 --name goMS1-mysql -v ~/my-mysql/sql-scripts:/docker-entrypoint-initdb.d/ -e MYSQL_ROOT_PASSWORD=password -e MYSQL_DATABASE=goMS1_db mysql:latest
```

The above command will deploy a mySQL container that is exposing its internal port 3306 (but mapped externally to 52364). It also sets the environment variable MYSQL_ROOT_PASSWORD to “password” and sets MYSQL_DATABASE to goMS1_db. The scripts to create the table and insert data are mounted into the docker container through “-v ~/my-mysql/sql-scripts:/docker-entrypoint-initdb.d/”. Take note to point to the correct directory before running the above docker run command.

Type the following command in Terminal:

```
docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
4378908ccc62	mysql:latest	"docker-entrypoint.s..."	13 minutes ago	Up 13 minutes	33060/tcp, 0.0.0.0:52364->3306/tcp	goMS1-mysql

Connecting to the MySQL Container using a Local MySQL Client, the following section assumes you already have the MySQL client installed on your local computer.

You can download and install MySQL from: <https://dev.mysql.com/downloads/mysql/>

Type the following command in Terminal:

```
mysql -P 52364 --protocol=tcp -u root -p
```

You will be prompted to enter password to login, the previously set environment password is "password".

Type the following in mysql client prompts:

```
USE goMS1_db;
```

```
SELECT * FROM CourseInfo;
```

```
mysql> USE goMS1_db;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SELECT * FROM CourseInfo;
+-----+-----+-----+-----+-----+
| Code | Title           | Dates           | Lecturer | Description |
+-----+-----+-----+-----+-----+
| 1    | Go Basic        | 13th - 15th Jan 2021 | Ching Yun Lee | Basic Programming Knowledge about Golang |
| 2    | Go Advanced     | 28 Jan - 3 Feb 2021 | Ben Low    | Learn advanced concepts in Go programming such as as packages, creation of data structu |
| 3    | Go In Action 1  | 24-26 Feb 2021   | Ben Low    | Explore the practical aspects of Go software development in detail. |
| 4    | Go In Action II | 17-19 March 2021  | Ching Yun Lee | A course about Go Security Best Practices |
| 5    | Go MicroService I | 5 - 7 April 2021 | Ching Yun Lee | Introduction to Microservices, APIs, containerization and microservice frameworks |
+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)

mysql>
```

Seeing the 5 rows of courses above shows that the MySQL docker container has been successfully deployed, and contains a database with 5 pre-loaded courses. Type "quit" into mysql client to exit the client.

Note: The container goMS1-mysql needs to be kept running in order for the REST API to query this database. Any changes to this database will be persisted until the container is removed.

2.2 Starting the REST API

While the docker mySQL container is still open and running, we can start the REST API.

On the terminal ,navigate to the folder **/REST** on your local directory and enter the command :

```
go run main.go
```

```
[thoyanbo@Yans-MacBook-Pro goMS1Assignment % cd REST
[thoyanbo@Yans-MacBook-Pro REST % go run main.go
Database opened
Listening at port 5000
```

The REST API is now connected to the docker database and listening for requests through localhost port 5000.

2.3 Starting the client console application.

Now, we can start the client console application.

On the terminal, navigate to the folder `/console` and enter the command:

```
go run main.go
```

```
[thoyanbo@Yans-MacBook-Pro console % go run main.go
=====
Welcome to API client console
=====
Please select 1 of the following URL queries:
c - Create
r - Read
u - Update
d - Delete
e - Exit Console
Please make a selection and hit enter.
```

On seeing the above console interface in terminal, our simple client console is ready to make http requests to the API of CRUD (create, read, update and delete) methods.

3. Test cases for the console application

3.1 Get all courses

While in console, enter `"r"` to read courses.

Then, enter `"1"` to read all courses.

```
200
List of all courses [{"Code":1,"Title":"Go Basic","Dates":"13th - 15th Jan 2021","Lecturer":"Ching Yun Lee","Description":"Basic Programming Knowledge about Golang"}, {"Code":2,"Title":"Go Advanced","Dates":"28 Jan - 3 Feb 2021","Lecturer":"Ben Low","Description":"Learn advanced concepts in Go programming such as as packages, creation o f data structures and error handling mechanism. Go's rich support for concurrency will also be discussed. "}, {"Code":3,"Title":"Go In Action 1","Dates":"24-26 Feb 2021","Lecturer":"Ben Low","Description":"Explore the practical aspects of Go software development in detail."}, {"Code":4,"Title":"Go In Action II","Dates":"17-19 March 2021","Lecturer":"Ching Yun Lee","Description":"A course about Go Security Best Practices"}, {"Code":5,"Title":"Go MicroService I","Dates":"5 - 7 April 2021","Lecturer":"Ching Yun Lee","Description":"Introduction to Microservices, APIs, containerization and microservice frameworks"}]
```

Test case : Pass

3.2 Get specific course

While in console, enter `"r"` to read courses.

Then, enter `"2"` to retrieve specific course.

Next, enter `"3"` to retrieve course 3.

```
Please enter course code:
3
200
{"Code":3,"Title":"Go In Action 1","Dates":"24-26 Feb 2021","Lecturer":"Ben Low","Description":"Explore the practical aspects of Go software development in detail."}
```

Test case : **Pass**

Alternatively, enter "7" to retrieve non-existent course 7.

```
Please enter course code:
7
404
404 - No course found
```

We will get a response with error code and error details logged to the console. The errors will also be logged in the console log and API log.

Test case : **Fail**

3.3 Create a new course.

While in console, enter "c" to read courses.

Then, enter "6" to create new course with course code 6.

Next, enter "Go Microservices II" for course title.

Next, enter "21 – 23 April" for course dates.

Next, enter "Ben Low" for course lecturer.

Next, enter "The final module for Go School. Testing and monitoring, CICD." For course description.

We will get a response if 201 if course is successfully added.

```
Please make a selection and hit enter.
c
Please enter the following details:
Please enter course code
6
Please enter course title
Go Microservices II
Please enter course dates
21 - 23 April
Please enter lecturer name
Ben Low
Please enter course description
The final module for Go School. Testing and monitoring, CICD.
201
201 - Course added: 6
```

Test case : **Pass**

Alternatively, enter "3" to create new course with course code 3 to attempt override of existing course 3.

Next, enter "Go Microservices II" for course title.

Next, enter "21 – 23 April" for course dates.

Next, enter "Ben Low" for course lecturer.

Next, enter "The final module for Go School. Testing and monitoring, CICD." For course description.

```

Please make a selection and hit enter.
c
Please enter the following details:
Please enter course code
3
Please enter course title
Go Microservices II
Please enter course dates
21 - 23 April
Please enter lecturer name
Ben Low
Please enter course description
The final module for Go School. Testing and monitoring, CICD.
409
409 - Duplicate course ID

```

We will get a response of 409 for duplicate course.

Test case : **Fail**

3.4 Update course

While in console, enter “u” to update courses.

Then, enter “3” to create new course with course code 3.

Next, press enter for course title. (we leave it blank so that it will not be updated)

Next, press enter for course dates. (we leave it blank so that it will not be updated)

Next, enter “Kheng Hian Ben Low” for course lecturer. (original data is “Ben Low”)

Next, press enter for course dates. (we leave it blank so that it will not be updated)

```

Please make a selection and hit enter.
u
Please enter the following details:
Please enter course code
3
Please enter course title. Press enter if no change.

Please enter course dates. Press enter if no change.

Please enter lecturer name. Press enter if no change.
Kheng Hian Ben Low
Please enter course description. Press enter if no change.

202
202 - Course updated: 3

```

We can check course 3 is updated correctly using read option.

Enter ‘r’ for read course. Enter ‘2’ for specific course. Enter ‘3’ to read course 3.

```

Please make a selection and hit enter.
r
Please select from the following
1. Get all courses.
2. Get specific course.
2
Please enter course code:
3
200
{"Code":3,"Title":"Go In Action 1","Dates":"24-26 Feb 2021","Lecturer":"Kheng Hian Ben Low","Description":"Explore the practical aspects of Go software development in detail 1."}

```

Lecturer field has been updated from “Ben Low” to “Kheng Hian Ben Low”.

Test case: **Pass**

3.5 Delete course

While in console, enter “d” to delete courses. Then, enter “3” to delete course 3.

```
Please make a selection and hit enter.
d
Please enter course code to delete.
3
202
202 - Course deleted: 3
```

We will get a response of 202 for successful course deleted.

Test case: **Pass**

If we repeat the above to delete course 3 again, we will get an error for no course found.

```
Please make a selection and hit enter.
d
Please enter course code to delete.
3
404
404 - No course found
```

Test case: **Fail**

Finally, enter ‘e’ to exit the console program.

Type the following command into terminal to stop and remove the MySQL container:

```
docker rm -f goMS1-mysql
```

4. Additional implementations for this assignment

In addition to the core requirements for this assignment, the following features/considerations have also been factored into the implementation of 3 core features.

- Security considerations:
 - Input validation and input sanitization (console and REST API)
 - Usage of API access keys to access database
 - The API access keys are stored in environment variables and not shown in source code
 - Potentially sensitive information such as database name, password and port are also hidden in environment variable.
 - Use of HTTPS for request and response between console and API (to simulate communication protection)
 - Use of prepared statements on API source code to prevent potential SQL injection
- Logging of errors and warnings on console and API server.