Connecting to a database server from a Docker environment

Deadline: 23:59 25.2.2024 for all parts of the assignment (AIS, Github and reporting the link to the Docker image via the form)

Overview

The task is aimed at creating the first version of a RESTful application server with an endpoint that returns the required values when called. This application server is then distributed as a Docker image (the Docker image is the input for the automated DBS tester you will use during the semester). Assignment 1 serves as the basis for the other assignments that will need to be developed during the semester.

Within the scope of the assignment, a functional RESTful API needs to be implemented. It is possible to use any programming language so that the required functionality is met. The solution needs to be published within your assigned private repository. The GitHub repository must contain a created Docker image that, when started, listens on TCP port 8000 and takes as input the environment variables (case sensitive) described in the table 1, which are used to configure the database connection. An example of such a solution is described in the FIIT-Databases/dbs-python-example repository. The repository also includes a CI/CD configuration (GitHub Actions) for automatic Docker image creation, which you can adapt to your solution with simple modifications.

Environment variable	Description	Example
DATABASE_HOST	IP address of database server	127.0.0.1
DATABASE_PORT	Database server port	5432
DATABASE_NAME	Database name	dbs
DATABASE_USER	Database user	xstudent
DATABASE_PASSWORD	Password for database user	SuperTajneHeslo

Table 1: Environment variables

The application server reads the variables from the table 1 and creates a connection to the database server based on them. The database server is PostgreSQL.

Aplikačný server implementuje RESTful volanie **GET /v1/status**, ktoré po zavolaní vráti JSON objekt, ktorý obsahuje verziu databázového servera. Pre získanie verzie databázového servera je potrebné zavolať dopyt (query) na databázu. Vrátený parameter je potrebné následne vrátiť v JSON objekte s nasledujúcou štruktúrou:

```json
{
  "version": "information returned by the database server"
}
```

In practice, a JSON object can look like this:

```
{
  "version": "PostgreSQL 15.1 on aarch64-apple-darwin21.6.0, compiled by Apple clang version
   ↪  14.0.0 (clang-1400.0.29.102), 64-bit"
}
```

To test your own implementation, you can use the cURL tool to make a call to the server. An example of a call to the http://127.0.0.1:8000 server with a call from a job might look like this:

```
curl http://127.0.0.1:8000/v1/status
```

To verify the correctness of your solution for an already created docker image, it is possible to use the tester-dbs.fiit.stuba.sk test tool. This tool will also be used for the final verification of the functionality of the assignment. You can you the AIS credentials to for log in. Then you just choose the assignment you would like to test and provide the link to the Docker image from you private repository.

A link to a Docker image in a GitHub repository might look like this: ghcr.io/fiit-databases/dbs-example:1.0.0

As an example, you can use the FIIT-Databases/dbs-python-example repository. The contents are a Python REST server implementation, a 'Dockerfile' and a GitHub Actions configuration for publishing a Docker image. You can manipulate the content of the repository under the MIT license and use it in your assignments. Adaptation for different programming languages and technologies should be easy.

Instructions for submission

Each assignment must be published as a Github Release. To automatically create a docker image, you need to use a CI/CD script that will automatically create the image.

A github workflow .github/workflows/publish.yml is created within the sample repository, which automatically creates an image for the master, main and semantic versioning x.x.x versions.

You do not need to create new releases in order to test a assignment. The tester also accepts images for 'master' or 'main'.

Your final release for an assignment needs to be marked according to semantic versioning, so for assignment 1 it is marked as 1.x.x. A sample example release is given in the example repository.

In order to successfully submit the assignment, all the points listed below must be met:

- The source code (so that the project can be compiled and run) along with the commit hash (which represents the final version of your assignment 1.) has been uploaded to **AIS**.
- A private repository is maintained on **GitHub** within the GitHub Classroom in the following format.
  - subject_and_year-surname_of_lector-schedule-surname_of_student
  - e.g. dbs24-bencel-stv9-mrkvicka-jan
  - Github Classroom is distributed separately for each exercise
- The address of your Docker Image has been reported using **Google Forms** https://forms.gle/Rq5CeUX25qzNy9Kt7

An assignment is considered submited if all parts (AIS, Github, Link to docker image) are submitted by the deadline. In case of late submission, the assignment is considered as not submitted.