eCommerce Back-End Documentation

Author: Toader Eric-Stefan

1. Overview

The eCommerce database was set up using MySQL 8.0.31. It contains the following tables:

* USER: stores the verified user accounts of the eCommerce website
* PRODUCT: stores all of the products sold by the eCommerce website

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| COLUMN | id | name | desc | price | available | image | rating |
| TYPE | INT | VARCHAR | VARCHAR | FLOAT | BOOL | LONGBLOB | FLOAT |
| PRIMARY | YES | NO | NO | NO | NO | NO | NO |
| NULLABLE | NO | NO | NO | NO | NO | YES | YES |
| DEFAULT | N/A | None | None | None | FALSE | None | 0 |

The table was populated using an existing dataset of electronic devices. In its initial state, the dataset’s price column is incomplete and is missing the ‘available’, ‘image’ and ‘rating’ columns.

To populate each product's ‘image’ attribute, I wrote a Python3 script that fetches the first available Bing Images picture it can find for each product name. The pictures are then stored locally and are inserted into the database with the help of an SQL Dump Generator program which I wrote in C.

The SQL Dump Generator takes the incomplete dataset and the absolute folder path of the downloaded images as inputs and creates an SQL Dump file containing the table creation of PRODUCT and the insertion of the processed, complete dataset.

Note: Compiling/Running the SQL Dump Generator on non-UNIX-based systems, such as Microsoft Windows, will result in an error. The program was written with POSIX system calls and libraries and the provided executable was compiled on macOS 13.0

Additionally, the executable was compiled on a machine running on the ARM processor architecture. Thus, it may be unsuitable for x86 systems. Recompilation on your local machine is advised.

1. Setup on local machine

To be able to run the web application on your local machine, it needs to access the Back End, which is not stored remotely using Cloud Services, but rather stored locally on your device.

In this section, the steps required to install and access the Back End on your local machine will be described in detail.

# Installing the database

The Back End uses a MySQL Database which needs to be installed and powered on for the application to be able to access its contents and display the products’ information for example.

Hence, you need to download and install MySQL Community Server via the link <https://dev.mysql.com/downloads/mysql/> where you can select the installation package that suits your system, by selecting your operating system type and version.

Graphical user interface, application

Description automatically generated

Once installed, make sure MySQL has been added to your PATH variable (<https://www.tutorialspoint.com/adding-mysql-to-windows-path>)

To check if it has been added correctly open a Terminal/ Command Prompt/ Windows Powershell and run the following command:

> mysql -- version

The expected output should be the version of your MySQL Server.

To access the database tables and be able to perform operations, the local server needs to be powered on. To achieve that, the following command needs to be run:

> mysqld -u root -p

If executed correctly and the password you entered matches the username ‘root’ , which was set up during the installation process, the server will now be up and running and you will enter the MySQL Command Line Interface within your terminal. While here, you will need to perform a few operations to set up the environment.

Firstly, let’s create the database schema on which we will operate. Run the following commands:

> CREATE DATABASE eCommerce;

> USE eCommerce;

Now, we need to create an administrator account for this database and give it all of the permissions available:

> CREATE USER ‘eComAdmin’@’localhost’ IDENTIFIED BY ‘admin’;

> GRANT ALL PRIVILEGES ON \*.\* TO ‘eComAdmin’@’localhost’ WITH GRANT OPTION;

# Populating the database

You will need Linux or macOS for this section.

If you already have the dump file configured, skip to the last paragraph of this section.

In this section, we will focus on uploading data onto the previously created database, for the web application to be able to perform operations on it.

The process itself has been streamlined drastically thanks to the auxiliary dump generator program which I wrote in C. Any number of rows can be inserted simultaneously into the database by executing only executing 2 commands within the terminal.

To get started, you have to identify where the ‘dumpGenerator’ program resides inside the resource package provided alongside this documentation.

Once you found the ‘dumpGenerator’ folder, enter the ‘src’ folder within it. Afterwards, right-click on the File Explorer/ Finder window and open another instance of Terminal and execute the following command:

> gcc main.c generator.c -o main

Now that we have compiled the source code, we will run the program by passing it the path to the product images (identify the folder containing folders of images labelled from 1 to 300 and copy its absolute path)

> ./main [ABSOLUTE PATH GOES HERE]

If the path provided was valid, the program will output a file with a .sql extension which will be used to populate the database with titles, descriptions, prices and pictures for 300 products. The file is called eCompDump.sql and is placed in the ‘out’ directory of ‘dumpGenerator’.

Now let’s use this generated file to populate the database. In another instance of Powershell/ Terminal, run the following command:

> mysql -u eComAdmin -p eCommerce < [ABSOLUTE PATH TO DUMP FILE GOES HERE]

# Starting up the Apache server

The last step in order for the Back End to be accesible by the web application is to start up the Apache server that exposes the dataset available locally using RESTful web services.

For this, simply open the eCommerce Back End IntelliJ project and run the ‘EcommerceApplication’.

If done successfully, the Front End will now be able to consume data exposed by the Back End and the web application will run optimally.

1. API Requests
2. Important notes
3. References

Initial dataset

<https://github.com/etano/productner/blob/master/Product%20Dataset.csv>

Spring Initialzr

## <https://start.spring.io>

Java Spring Boot documentation

<https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/>

MySQL documentation

<https://docs.oracle.com/en-us/iaas/mysql-database/doc/getting-started.html>