Hemran Akhtari

Mrs. Winnie van Schilt & Mr. Rob Smit

Data Processing

27 May 2021

Documentation

**1. Introduction**

Starting the project, I picked the World database from [dev.mysql.com](http://dev.mysql.com) as my dataset with MySQL (using XAMPP) as my database language.

My API language of choice is Node.JS and for the visualisation part I used Python Flask. My goal was to create an API with Node.JS that accesses the MySQL database to provide a dataset in JSON and XML to my Python application without connecting directly to the MySQL database which gives you serious benefits such as high security and low latency applications.

**2. API (Node.JS)**

**2.1 Prerequisites**

The first step for creating the API is to download Node.JS from [nodejs.org](http://nodejs.org). After successful installation you need to install the following packages with the “npm install” command in your console of choice (I used Git Bash):

* Express
* Mysql
* Express-xml-bodyparser
* Ajv
* Xmllint
* Morgan
* Cookie-Parser
* Http-Errors

**2.2 Execution**

The next step is to import those installed packages with the “require” method in the index.js. After initiating the packages, I defined the base routes to my nodes and defined a port the node.JS server is running. Furthermore, I created a connection to my database.

After successfully creating the index.js file, I went on creating the route file which in my case is routes/world.js. Inside this file I created routes to controllers for either (C)reating, (R)eading, (U)pdating or (D)eleting rows in the database. The controllers on the other hand are fetching and processing the data to finally send them as a response to the request.

**3. Visualisation (Flask)**

**3.1 Prerequisites**

These are the steps that I have taken to run flask:

1. Install Python
2. Create project folder
3. Crerate virtual environmen: python venv venv
4. Command: venv\scripts\activate
5. Install flask: pip install flask

**3.2 Execution**

After installing the necessary packages, I started by creating the \_\_init\_\_.py file which imports flask, the routes and runs the app. Next, I created the routes.py file which defines the web routes and request data from the API (either XML or JSON depending on URL), parses the data and passes to the views which are the templates. Inside the templates I processed the data by visualizing it via charts, icons etc. Furthermore, the application is able to switch between XML and JSON easily by a click of a button.

**4. Why did I choose those languages?**

I chose Node.JS for its known high performance (uses Googles JSv8 Engine which compiles JS directly into machine code), easy scalability (horizontal scaling: adding nodes to existing system, vertical scaling: adding extra resources to existing nodes) and caching (gets cached in the application memory, so developers don’t have to re-execute the code). Python Flask on the other hand, since it has a lot of function for data manipulation, processing and parsing. Additionally, Flask has a high scalability, simple routing, lightweight API request integration and small core which increases processing performance drastically. Lastly, Python Flask is modular which makes it perfect for Data Processing since it is dynamic in its functionalities and is able to process a variety of data structures.