Documentation  
Hylke Westerdijk  
Winnie van Schilt and Rob Smit  
17th of January 2020  
  
Introduction

The dataset that was chosen by me was also created by me. The database was made using MySQL in combination with xampp and PhpMyAdmin. My API was created with NodeJS and was my visualization was done with Python. The goal of this project was to create an API that took data from my MySQL database, convert the data to XML and JSON and then create a Python script that could consume the XML and JSON into graphs without my script ever having to connect to the database directly.

API (NodeJS)

**Prerequisites**To be able to create an API in NodeJS there are a few steps you must follow. The first step is downloading and installing NodeJS from nodejs.org after the installation you’ll need to install the following libraries by using the command “npm install <name\_of\_module>” in a console such as the CMD.  
  
- express  
- body-parser  
- mysql  
- js2xmlparser  
- express-xml-bodyparser

**Execution**Once you’ve installed all the packages, you’ll need to start by making a new .js file and importing the libraries by creating constants or variables. After doing such you need to create a default route as well as a connection to your database with a port of your choice.  
Once you’ve taken all those steps it’s time to make all your requests including: get, post, put and delete, these will be used to Select, Insert, Update and Delete data from the database through the API. Using “body-parser” or “xml-bodyparser” I was able to parse the data from the database into either a JSON or XML file. Finally, to convert the JSON into XML I used the “js2xmlparser” library.

Visualization (Python)

**Prerequisites**To be able to create a script in Python to consume an API there are a few steps you must follow. The first step is downloading and installing Python from python.org after the installation you’ll need to install the following libraries by using the command “pip install <name\_of\_module>” in a console such as the CMD.  
  
- json  
- requests  
- numpy  
- pyplot  
- jsonschema  
- xml.etree  
- xml.etree.ElementTree  
- lxml

**Execution**Once you’ve installed all the libraries, you’ll need to start by making a new .py file and importing the packages by using “import <library\_name>”. After having created your imports you’ll need to make a request to your localhost to retrieve the data in either JSON or XML format. Then for JSON you create a variable with what part of the JSON you want in my case “[‘data’]”. Then you validate your JSON using jsonschema’s validate, if this returns None then you move on to making the graph. In the case of the XML you must first convert the XML to a string and then decode that String in the UTF-8 format and then specify that the XML being read is a String, after doing all of this you can validate the XML by comparing it to my own created .xsd schema. The validate method is created in a separate class called validate.py, in laymen terms it validates the XML compared to the .xsd file. After all of this you can finally create a graph, but to do so you first need to run a loop that grabs all the data from the XML file and puts each countries individual data into a map and then puts each country map into an array so you can use the data in a graph.

Argumentation

I could have used any programming language, but I decided to use NodeJS for my API, my reasoning is: NodeJS is an efficient, lightweight and fast programming language due to being asynchronous(non-blocking) rather than synchronous(blocking). NodeJS can also read JSON natively and read XML easily with the addition of a library. For the reasons I stated previously I would certainly recommend NodeJS to create a JSON/XML API. I also decided to use Python for my visualization, I did this because, Python is compatible with many languages. It can read JSON extremely easily with only a few lines of code. Many libraries and quite simple. The only issue I had with Python with its compatibility with XML, I spent a lot of time trying to figure out how to be able to first of all read XML let alone validate it and for this reason I would not recommend Python if you plan to use it with XML.