Rami MAzloum

17 January 2020

1.0

Data processing

[Document subtitle]

Contents

[Introduction 2](#_Toc30169058)

[Building API 2](#_Toc30169059)

[Error Handling 4](#_Toc30169060)

[Handling request 5](#_Toc30169061)

[Database linkage 6](#_Toc30169062)

[Data visualization 6](#_Toc30169063)

[Running the code 6](#_Toc30169064)

# Introduction

This project has a visualization of alcohol compassion for Eupen countries passed on the user choose represented by a pie chart. Also, visualization of the reelection between alcohol usage level, freedom level and the number of LGBT people in each EU country.

This project is using nod.je to create the REST API which is linked into database has all the data are needed to give a valuable data. In ideation to JSON XML has been used, therefor there are two methods to provide data.

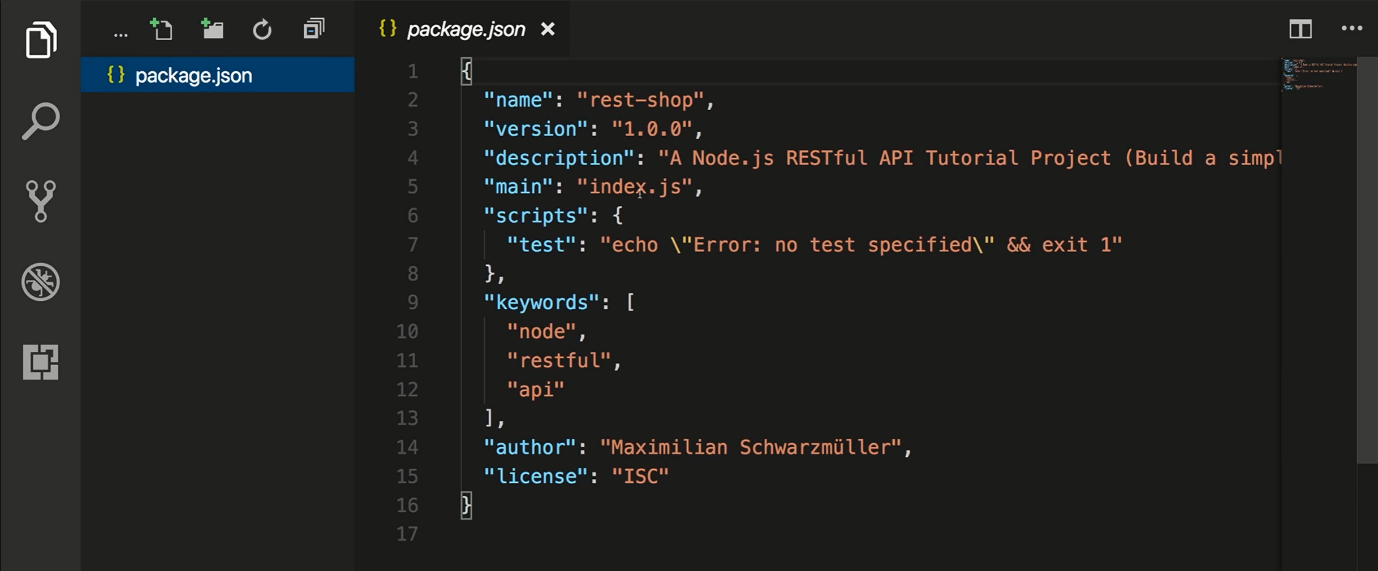
In order to visuals the data which has been created by json or xml, Python has been used to consume this data and visuals it in pie chart and bubble chart.

In addition, the REST API proved a functionality of POST, delete and UPDETE the database which is linked to.

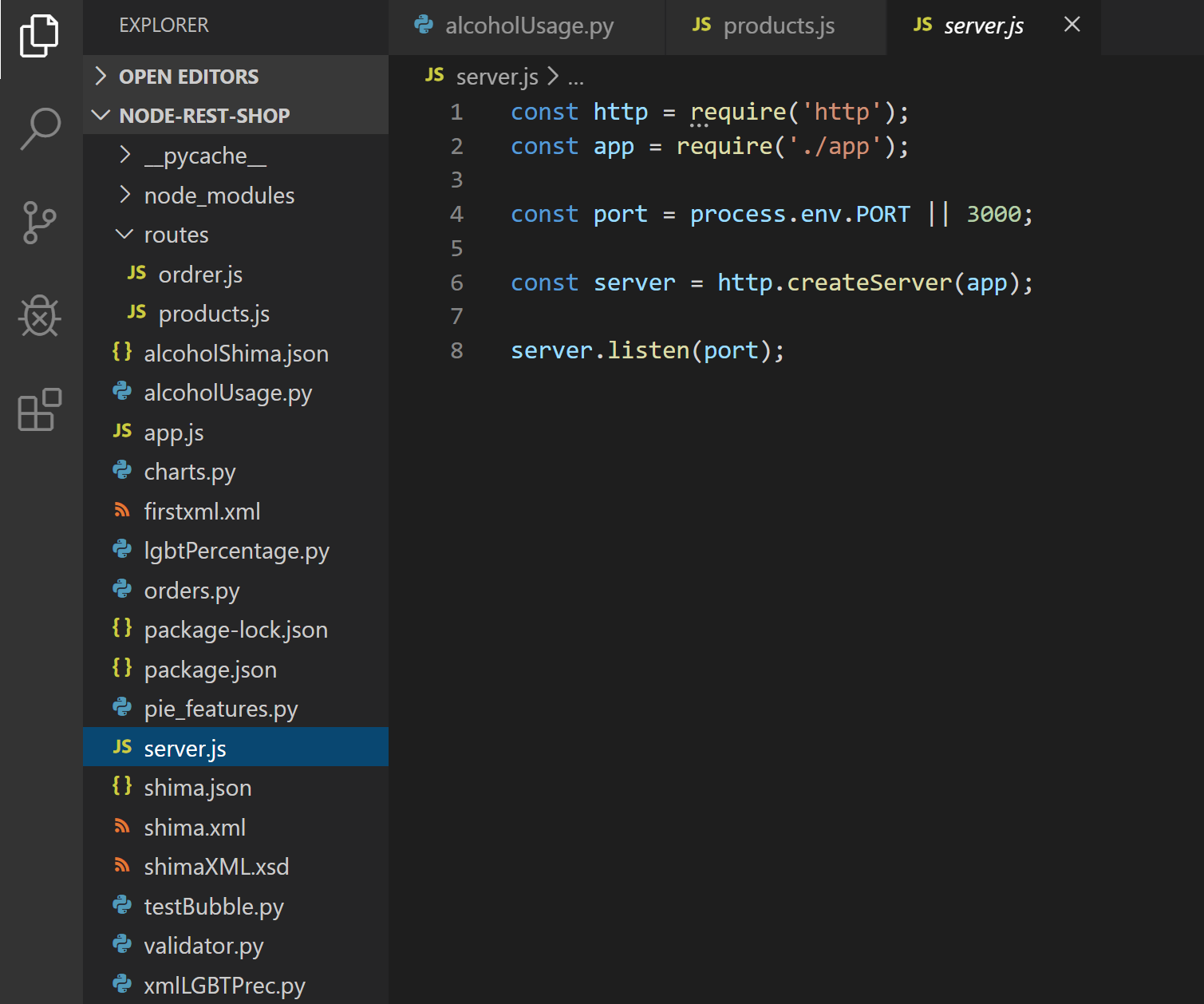
# Building API

In order to build the API, you need to install Nodjs form [hier](https://nodejs.org/en/), download the last version. While its installing creates a new folder where you make the project.

In your command Prompt enter the file you creeted by inter the command: mkdir filename. Install Nodjs by inserting: npm init. Then give it a package name, description, and then keep pressing enter until it reaches Is this OK? Type yes. Now chose a editor and open the file you created, I’m using Visual Studio Code. Check the example below.



In your editor open the terminal and install express package. npm install --save express.It’s a framework for Nodejs to make bulging the API easier.

create a new file to run the API server. First variable Import a constant from Nodjs to run the server on the browser const http = require('http'). The second variable is for handling where the port will be run, if its injected or default 3000 const app = require('./app');. const server = http.createServer(app) is a Function it response every time there is a request . (Figur1)

(Figur1)

The file app.js will run the express package to make handling request easier. const express = require('express’) will call the express package and the const app = express() will execute it as a function. app.use use is a method works as a middleware, an incoming request should go through the it. Using Postman app to test the API by using <http://localhost:3000/>.

In the movie.js file its handling the GET, POST, UPDATE and DELET. Again we will need use const express = express() plus const router = express.Router() will be able to handle different routes(GET, POST, UPDATE, DELET) with different http verbs.

Keep in mind that every time you make some change on the server you need to restart the server, to prevent that I’m installing Nodemon. In the VSC terminal type: npm install –save-dev bodemon. This will restart the server every time you change anything in the server automatically. To make it works you need to go to package.json and add "start": "nodemon server.js" (Figur2) so when you type in terminal npm start it will start the server and restart it in case you make any changes.

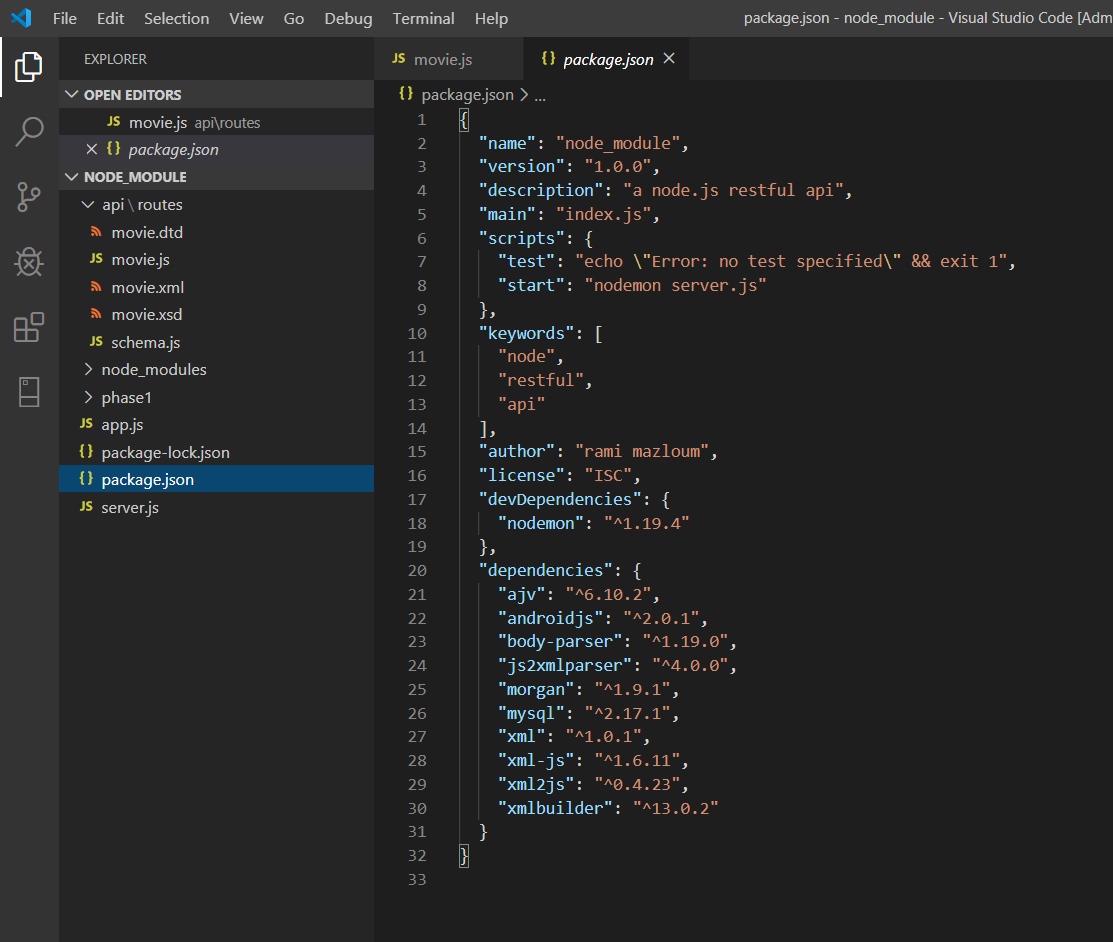
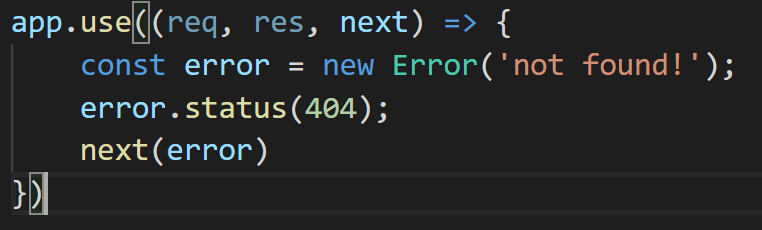
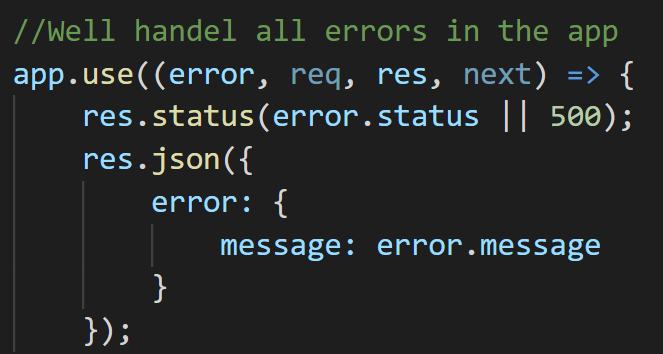


Figure 2

# Error Handling

So far, the server we have build has no error handling, in the chapter we will add some error handlers. In the app.js project file I’m making a middleware to throw error, this middleware should be after the routs handler.

Error object is available by default. next(error) will pass the error.

Another middleware error handler will handle all kinds of error for example Jons errors.

# Handling request

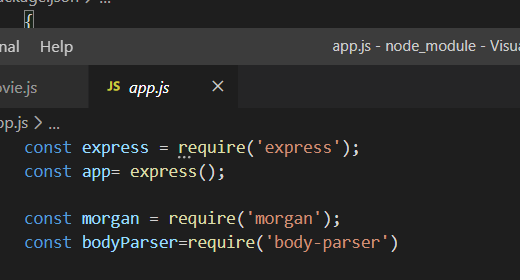
To make the server we build able to handle requests from GET,POST,DELET I’m installing a parser package, this parser handle url

. In the terminal insert npm install --save body-parser. In app.js project file I’m doing the following.

const bodyParser = require('body-parser') is to import the parser.

app.use(bodyParser.urlencoded({ extended: false })) is used apply it. False is for simple data and True is for rich data.

app.use(bodyParser.json()) for handling json data.



movie.js file project has GET, POST, delete and UPDETE functionalities

To send GET request I don’t need to provide any kind of information

* To send a delete request I’m providing budgetId only
* To send a POST request I’m proving the following data:
  + Budget of the movie
  + The original language
* To send a PATCH (update) request I’m providing following data:
  + Budget Id
  + Budget of the movie
  + The original language

# Database linkage

Import movie.sql to database using xamPP Control Panel. Inside movie.js I’m importing the database (var mysql = require('mysql')) and inside each method(GET,POST,DELET, and PATCH) I’m creating a connection and then I’m closing it for security preps.

Now by using Postman I can test my GET, POST,DELET and PATCH.

# Data visualization

After I’m done with building and testing the API and the concoction to the database, I’m using JavaScript for visualize the data I’m getting from API by using JOSN and XML.

# Running the code

1. Open xaMPP files and move the project node\_module file inside htdocs file.
2. Stat xaMPP and run the virtual server.
3. Use visual Studio Code to open the project.
4. In visual Studio Code terminal type npm start to start the server.
5. move the project visualize file inside htdocs file
6. Open <http://127.0.0.1/visualize/index.php> in you browser .

Now you see the results.