Zama project report : Web API (Node.js, Python)

Adrien JEANNEROT

November 10, 2021

1 Implementation

I decided to develop this web API using NodeJs and Express. I think these are very flexible and user-friendly techno to use for this kind of project. At the beginning, I hesitated with a .Net API connected to a python service because I was more familiar with this type of architecture but Node.js finally got me. I connected my API to mondodb atlas which is a free to use cloud database. It was the simplest and quickest way I found to have a db in order to allow user registration and connection.

2 Libraries and packages

One of the challenges was to connect my API to a ML service. The first idea that came to my mind was to use a python service with a flask API to query it. I was not confident enough with Flask, and considering the time I had, I decided to simply call a python script directly from my Node.js controller. I used the PythonShell package to communicate with my python script.

This script is using the common tensorflow and keras packages. I used a dataset I knew, the famous Iris dataset. It is a relatively small dataset, composed of entries of three different flower types, with their own characteristics. The inputs values are 4 float numbers representing sepal and petal length and width. The goal is to determine the flower type using the neural network. I had to decide how to store my model after the training phase. I needed to load an already trained model to avoid a training phase at each new prediction query from the API. I decided to store it locally after a training phase I did manually. The inputs values are sent in a JSON file and the output flower type is sent back to the API and then returned as a JSON too.

The API has 4 routes :

- GET: /api/index/test
  This route is used as a test route to check if the web API is reachable. Originally, I wanted to use this route in a CI/CD gitlab pipeline.
- POST: /api/auth/signupUser
  This route is the user endpoint to register to the API. The user db schema needs a nickname, mail and password.
- POST: /api/auth/signinUser
  This route is the used endpoint to log into the API. It needs an email and a password existing in the db. A JSON containing the authentification token (JSON web token) is sent back.
- POST: /api/prediction/predict This route is the inference endpoint to query the neural network. A request must contain a JSON file with the 4 flower characteristics (seplength, sepwidth, petlength, petwidth). The user token obtained from the signIn route must be passed in the authorization header field.

A JSON file "ProjectZama.postman-collection" in the root folder can be imported in postman to directly try these routes.

3 Perspectives

If I had more time on this project, I would change and enhance few things.

- I really wanted to incorporate a swagger documentation into my API, also to directly and easily try the routes without using postman. This is probably the first thing I would do if I had to work on this project again.

- Another thing I wanted to improve is the model storage. In this case, I had to train it manually and to store it locally. This is directly related to the fact that I didn't use python as a service with Flask for example, but as a simple script. I would find a solution to change that and to store my model in the cloud probably.
- To containerized my API, I used an existing and free to use docker image with node and python in it. With more time, I would try (again) to create my own image and to use it to deploy my container.
- Last but not least, I would probably comment my code a little bit more, adding documentation, and clean it, to make it easily reusable.