***Practice’s Week 5th & 6 th***

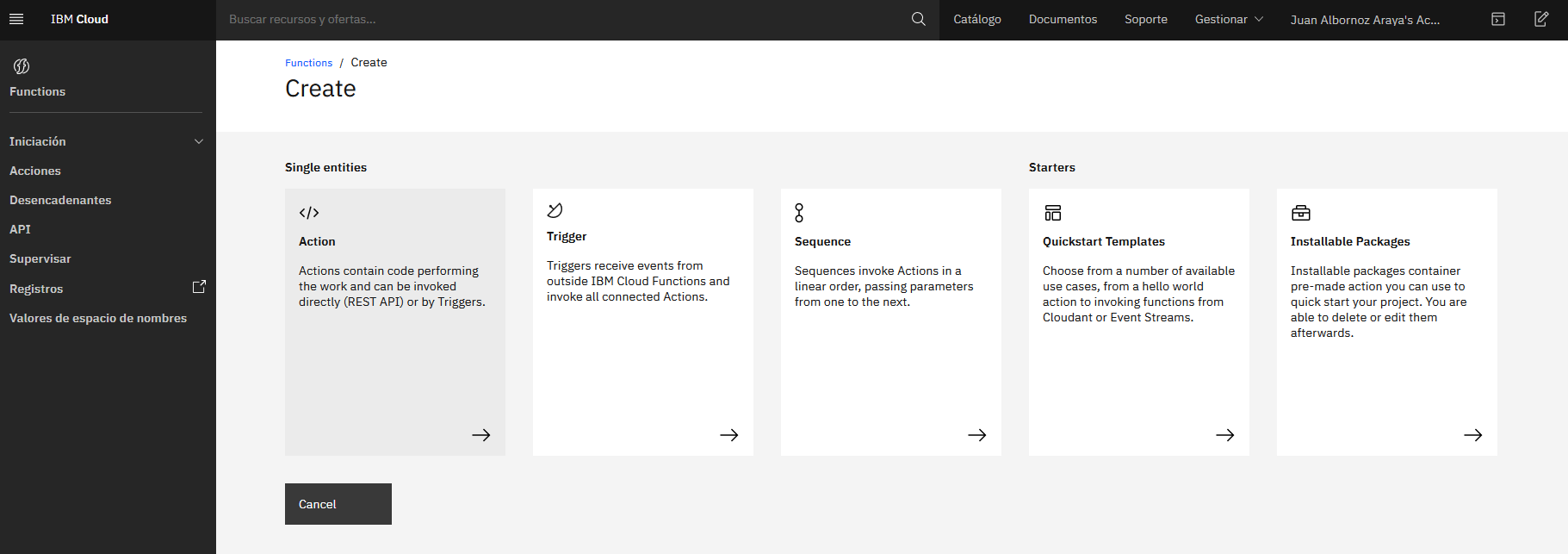
*Into IBM & Azure Cloud Services*

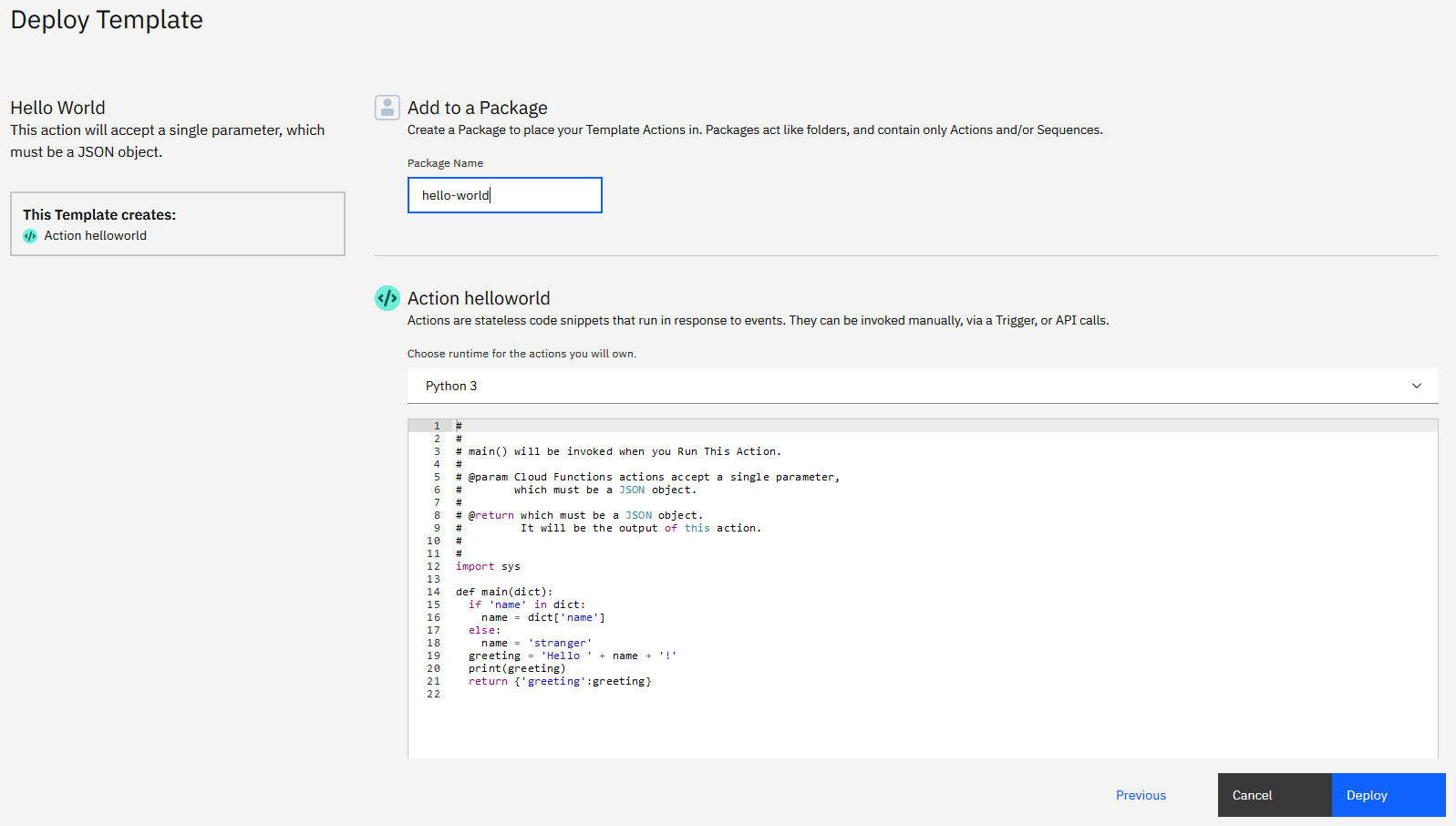
Cloud Services (Introduction)

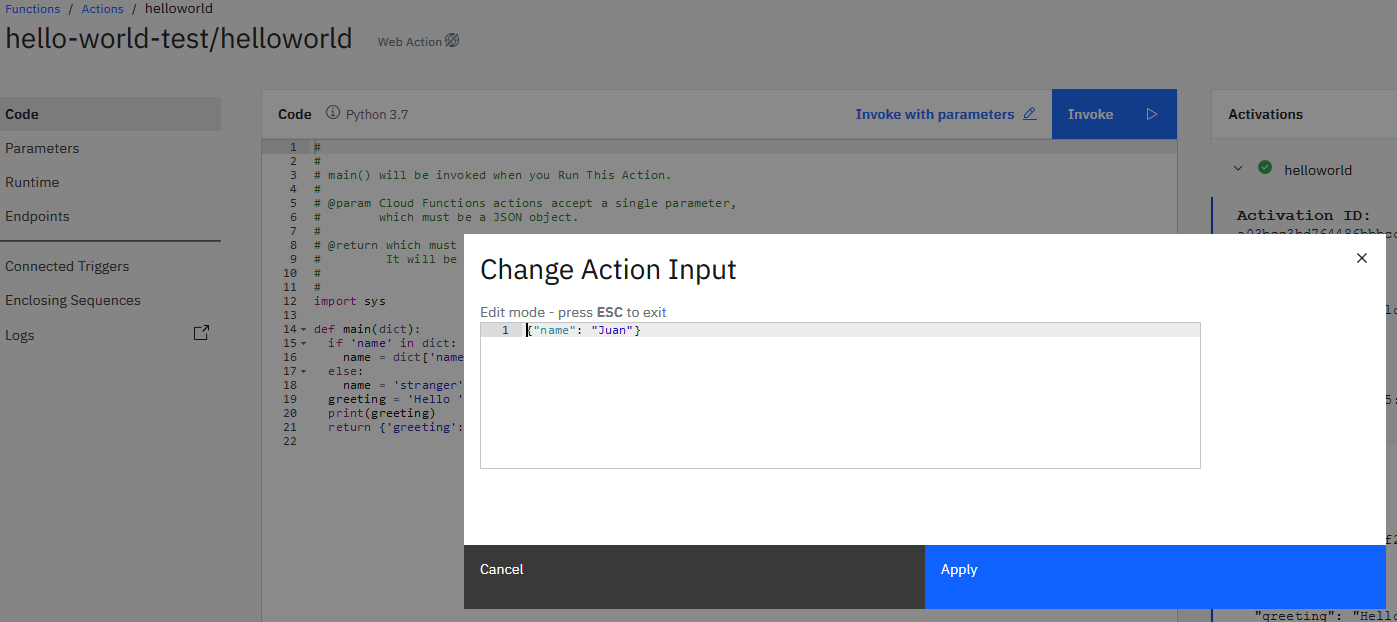
Teacher: Mr. Zheng Li

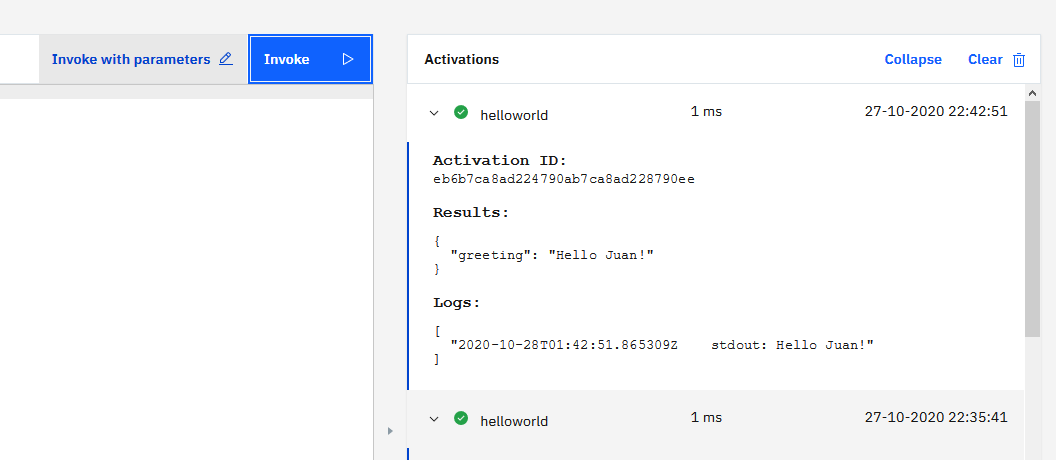
Student: Juan Albornoz

# Task 1: Fibonacci calculation via IBM Cloud (5th Practice)

* 1. Getting into the platform there is a Functions Service available for creating actions, triggers, sequences and some quickstart templates. I’m going to use a quickstart template for the deployment of the Hello world function:
  2. In the deployment’s window I’ll choose Python 3 runtime. It is well known that these functions receive and return JSON objects in the same way Amazon Web Services does. Then, deploy the function:



* 1. The function can be invoked passing the parameter in the JSON format:
  2. The function is invoked and the activation’s logs are shown in the left side of the screen:



* 1. Modified the function for calculating the 35th Fibonacci number. The main() function receives a dictionary and I passed the num parameter to the Fib() function for executing it and measuring the latency:

import sys

import json

import time

def fib(num):

if num<=1:

return num

return fib(num - 1) + fib(num - 2)

def main(dict):

start=time.time()

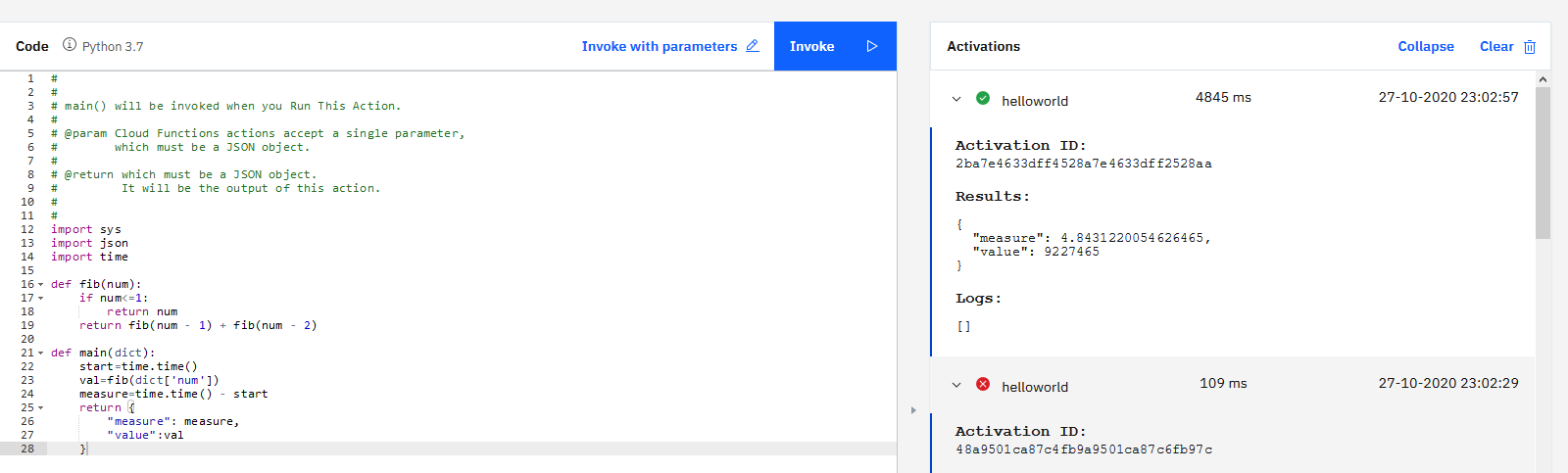
val=fib(dict['num'])

measure=time.time() - start

return {

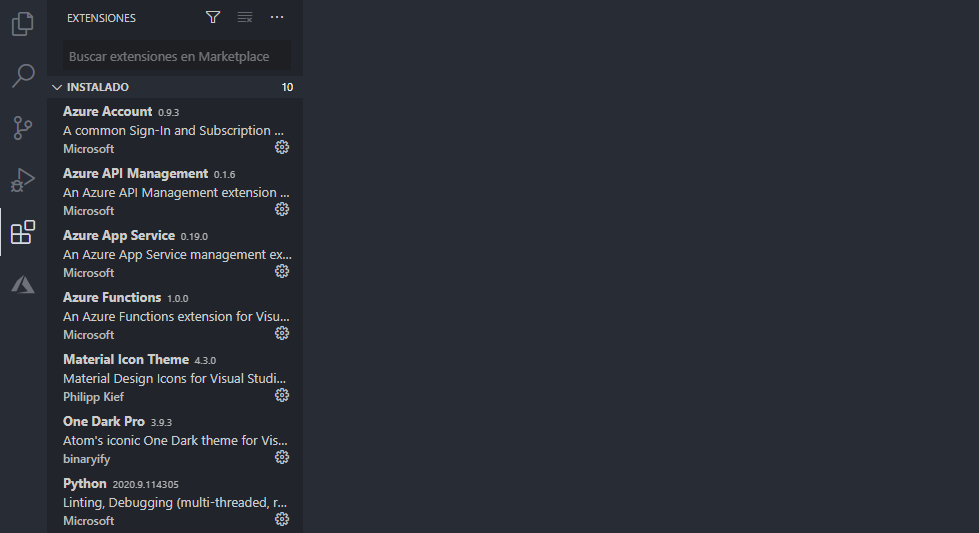
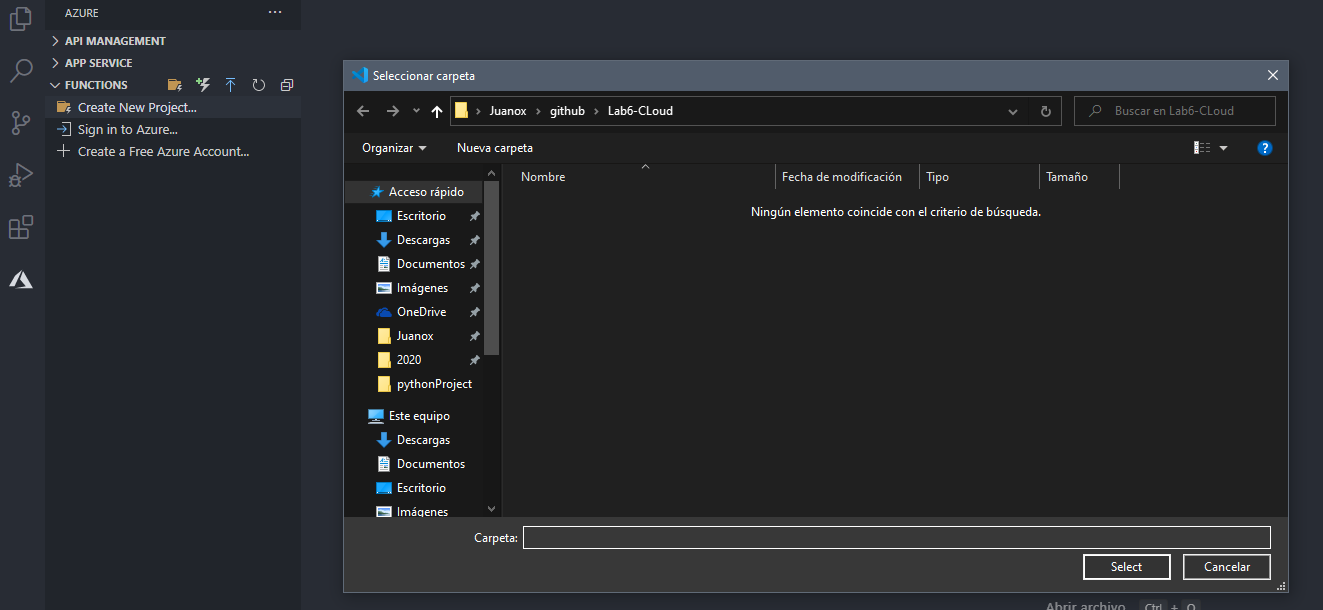
"measure": measure,

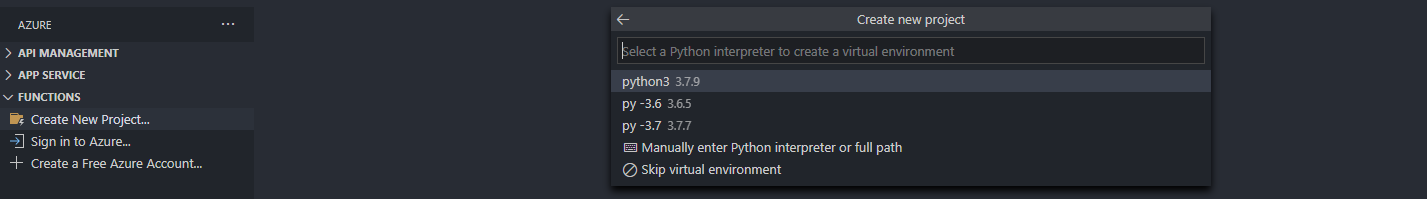
"value":val

 }

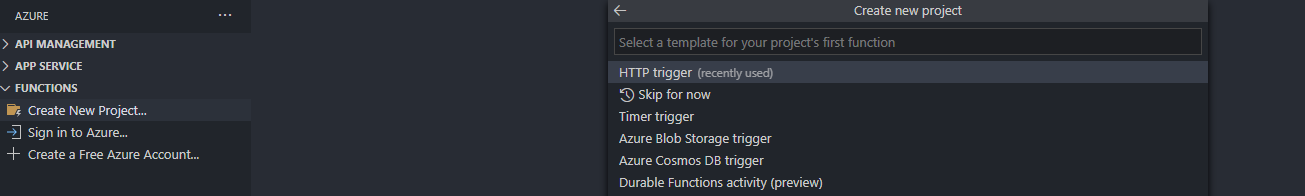
* Latency: 4.8431 sec.
* Value: 9.227.465

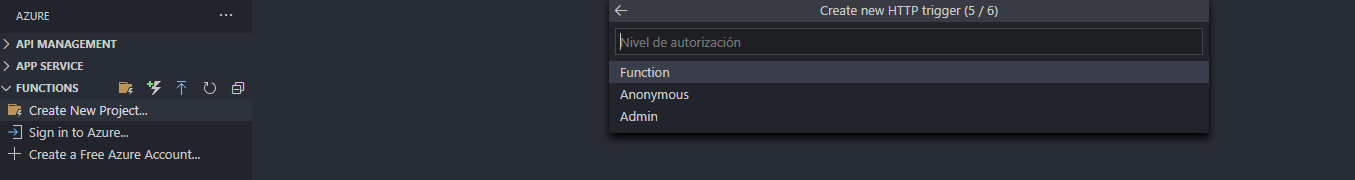
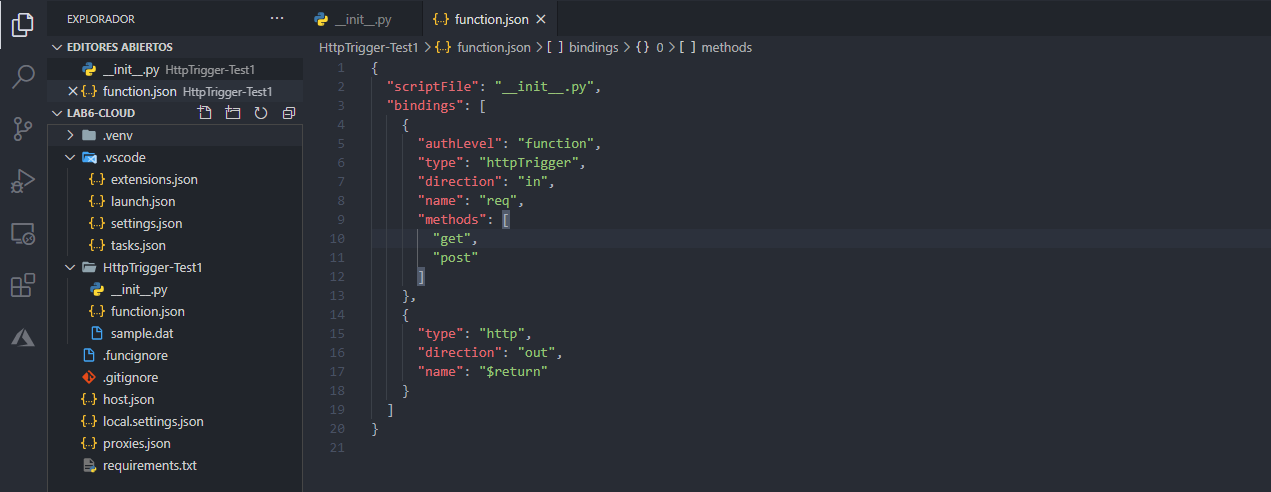
# Task 1: Explore a (PaaS or IaaS or FaaS) service from Microsoft Azure

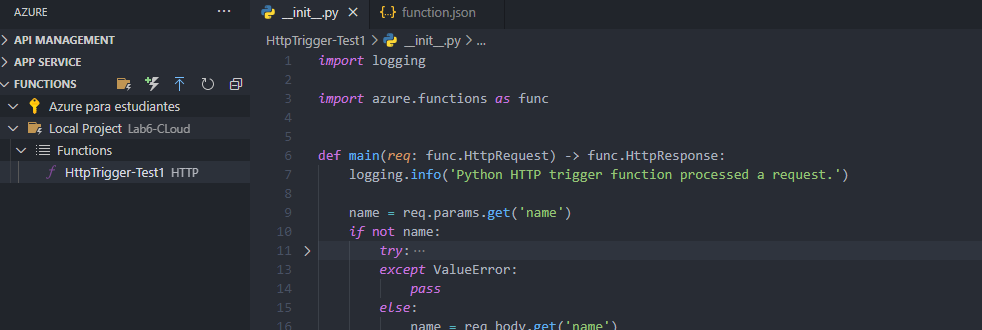
* 1. Installed Visual Studio Code for developing with Azure services. Installed the next extensions:
* Azure Account
* Azure Functions
* & Python support
  1. Create a new project in the Azure functions apart:
  2. Select a Python Interpreter to create the virtual environment. Selected Python 3:



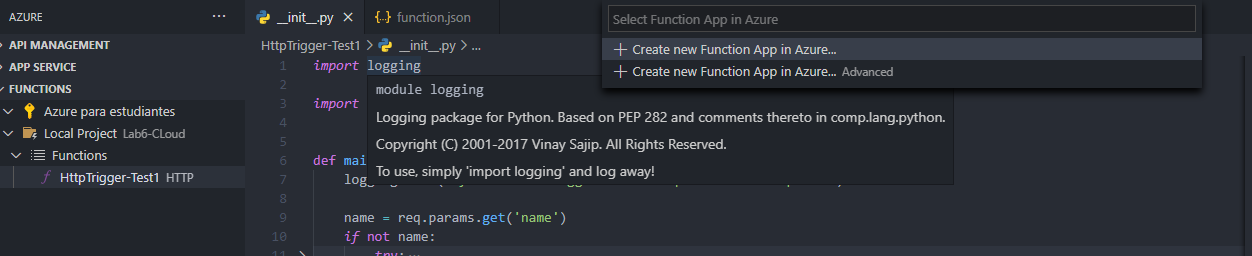
* 1. Select the template type for the function. Selected Http trigger:



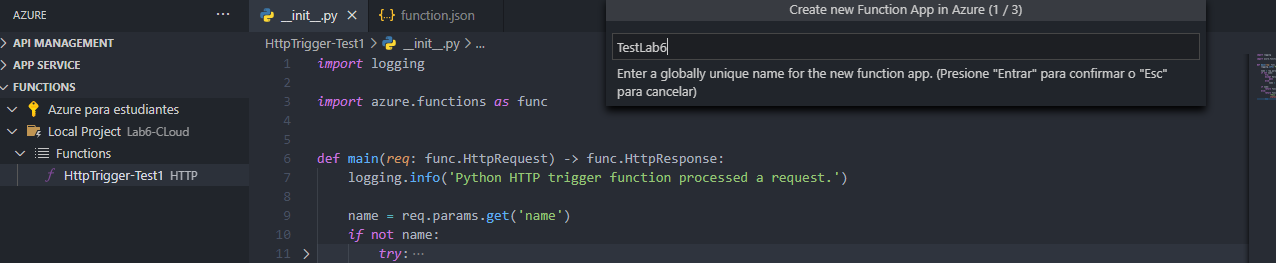
* 1. Selected the authorization level as “Function”. Then the Project is created:
  2. The function’s Project is created and there are two main files in it:
* **\_\_init\_\_.py**: It has the code for the serverless function.
* **function.json**: Contains the configuration of the function like the script file name and also determines wich Http actions to accept, GET, POST, etc.
  1. Then I’m going to deploy the function app to the Azure Students Services via VSCode in the deployment apart:



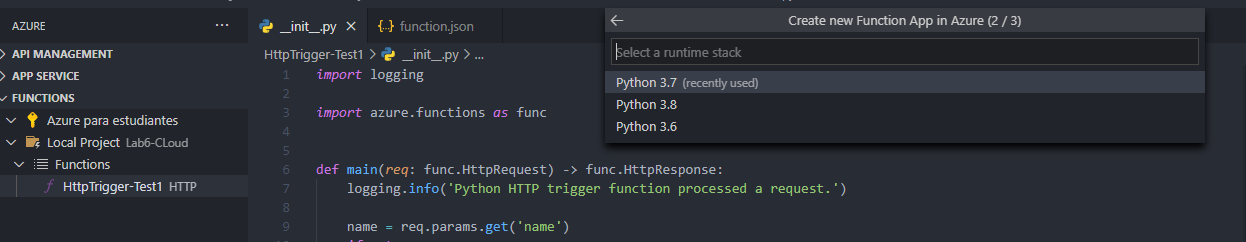
* 1. Create a new function app in Azure:

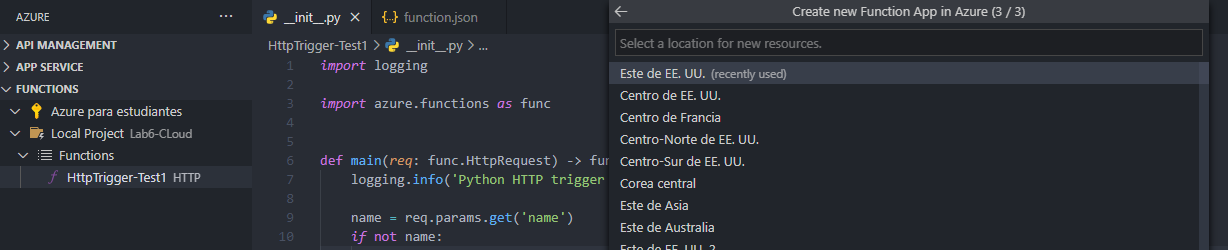
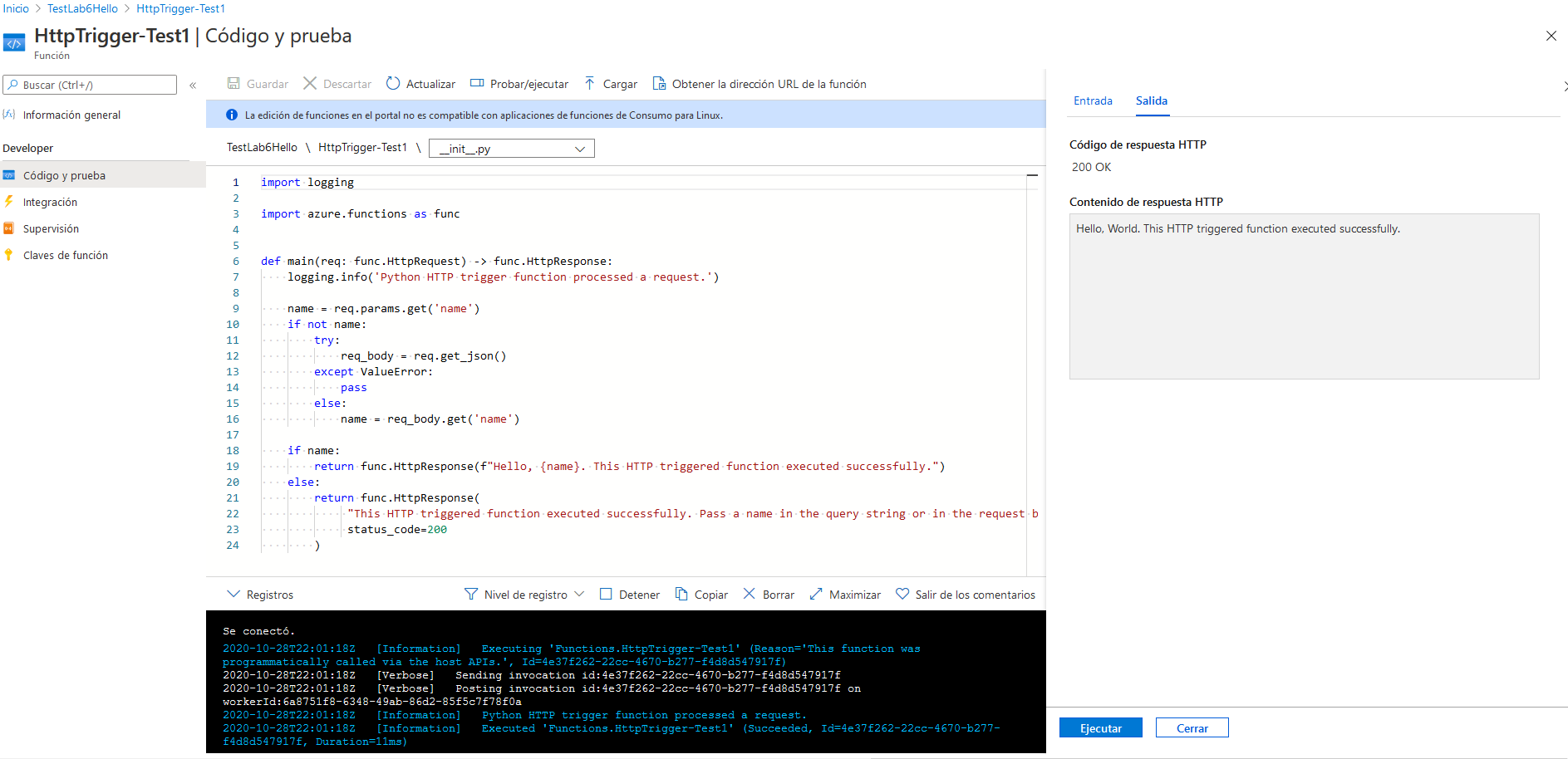
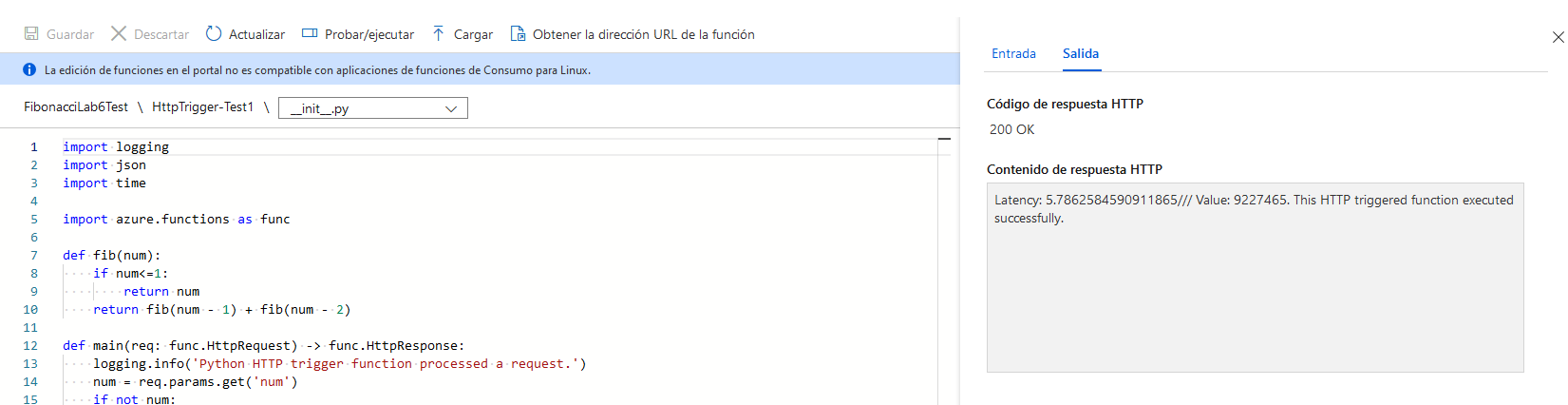


* 1. Give it a global name to the function:



* 1. Selected Python 3.7 as the runtime stack:



* 1. Selected the location for the resources:
  2. The deployment in Azure service is done succesfully and it can be tested in the Azure resources website:
  3. Changed the function to the Fibonacci 35th calculation and deploying it in the same way as before, then it gave me the next results:
* Latency: 5.7862 sec.
* Value: 9227465

# Task 2: Performance evaluation of cloud services:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Latencies by Service (seconds) | EC2 AWS | IBM Cloud | Lambda AWS | Microsoft Azure |
| Execution #1 | Pendent-- | **4.5596** | **8.9203** | **5.7862** |
| Execution #2 | -- | **4.4715** | **8.9360** | **5.7810** |
| Execution #3 | -- | **4.5035** | **8.9175** | **5.7899** |
| Execution #4 | -- | **4.4677** | **8.9282** | **5.8059** |
| Execution #5 | -- | **4.4573** | **8.9877** | **6.7309** |
| Execution #6 | -- | **4.4665** | **8.9250** | **6.5989** |
| Execution #7 | -- | **4.4667** | **8.9974** | **6.6084** |
| Execution #8 | -- | **4.4819** | **8.9860** | **6.6299** |
| Execution #9 | -- | **4.4470** | **8.9158** | **6.6688** |
| Execution #10 | -- | **4.4458** | **8.9656** | **6.7235** |

* Averages:
  + IBM Cloud: **4.4768 secs.**
  + Lambda AWS: **8.9479 secs.**
  + Microsoft Azure: **6.3123 secs.**
  + **EC2 AWS: Pendent**
* Standard Deviations:
  + IBM Cloud: **0.0318 secs.**
  + Lambda AWS: **0.0309 secs.**
  + Microsoft Azure: **0.427 secs.**
  + **EC2 AWS: Pendent**

## Summarizing:

* If I would have to choose one of the services in terms of intuitiveness of the interface and the plus of using the VSCode for deploying the function in the cloud service, it’s definitely Microsoft Azure Cloud Services. But I dont really know if the functions support Json formats because the Hello World example came with another input and output formats.
* In terms of latency the best service it’s shown in the results and is for sure IBM Cloud, with the lowest latency and also the lowest standard deviation for the set of results.