Cloud Computing and Big Data

LIS 4102 SECTION 1

Dr. Genoveva Vargas-Solar

HO1: Understanding externalized settings on the cloud

Melina Escobedo Zárate

ID 164094

February 20th ‘22

Cloud computing is responsible for offering services through the connectivity and large scale of the Internet. Cloud computing democratizes access to world-class software resources, as it is a software application that serves diverse customers.

Therefore, nowadays there are three models due to the increase in use:

- The SaaS - Software as a Service - model focuses on providing access to the software application for the user via a browser or program interface. With this model, the underlying network, operational system and resources run behind the scenes.
- The PaaS - Platform as a Service - model can leverage the benefits of cloud computing while maintaining the freedom to develop custom software applications. Users can access PaaS in the same way as SaaS. The provider is responsible for maintaining the operational system, network, servers and security.
- The IaaS model - Infrastructure as a Service - goes a step further in abstraction, providing organizations with the ability to leverage raw server resources while the rest of the platform and software management is the responsibility of the enterprise. This allows for greater capacity without the need to worry about hardware requirements.

The main objective of this practice is to get familiar with the use of Infrastructure as a Service (IaaS) and Platform as a Service (PaaS) cloud services for building and deploying applications on the cloud, with the use and help of different tool and technologies like Azure, Google (Server, Drive), Ngrok, and more.

In the Diagram 1 we are able to understood how does this cloud services are involved all around in an App. The functional architecture diagram explain or represents how does we prepared the environment. What we did first, was installing Azure CLI in Google server, then we connected our Azure account. After this little steps we retrieved the code from ngrok-stable-linux-amd.64.zip and executed ngrok.yml with a unique token. Finally we cloned Hands-on exercise at GitHub repository to Google Drive and after this, installed (code) in Google Server.

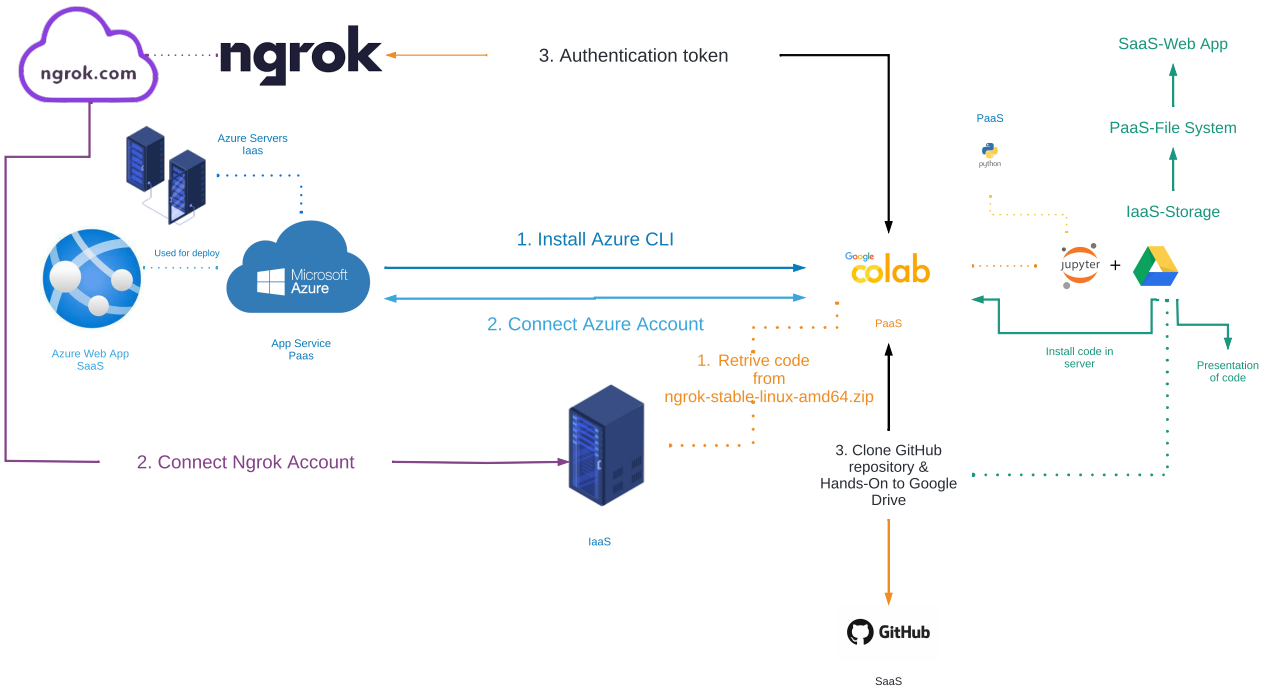

Diagram 1: Functional Architecture

The process that summarized the main steps and sub-steps of the excessive above are: Prepare the environment to work with, connect to GitHub to retrieve Hands-On exercise/practice, open Hands-On in Google Collab, after that is the installation of Azure CLI and then configured, also install grok and validate the account.

By connecting the account, we need a copy of our token on the notebook, then expose a web server (running local on our machine) on internet.

Clone the sample from GitHub repository and install Python dependencies. Next navigate into the application folder and install all the requirements, at that point we need to install !pip install flask-ngrok.

Subsequently run the sample, to be able to load and execute the sample in Google Collab, and then follow the grok URL to see the output.

Afterwards is to deploy the sample, at that moment we needed to deploy the code in our local folder pyhton-docs-hello-world using the az web app up command.

Finally brows on the app.

During the Lab Session we follow a tutorial in which we made use of some services, like:

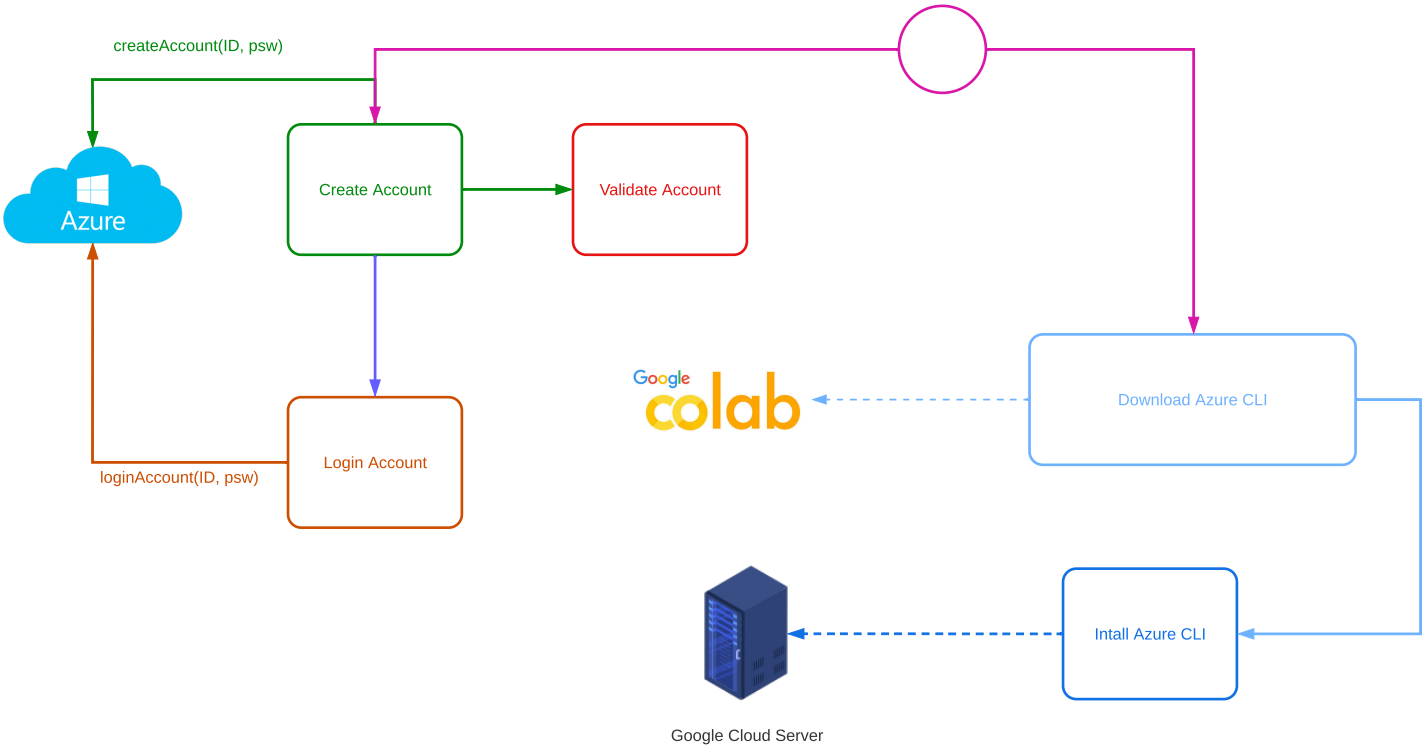

Diagram 2: Azure workflow

- The Azure command-line interface (Azure CLI) is a set of commands used to create and manage Azure resources. The Azure CLI is available across Azure services and is designed to get you working quickly with Azure, with an emphasis on automation. The Command-Line Interface is a cross-platform Command-Line tool designed to work with different Azure resources.

Ngrok authentication was made like in the diagram below::

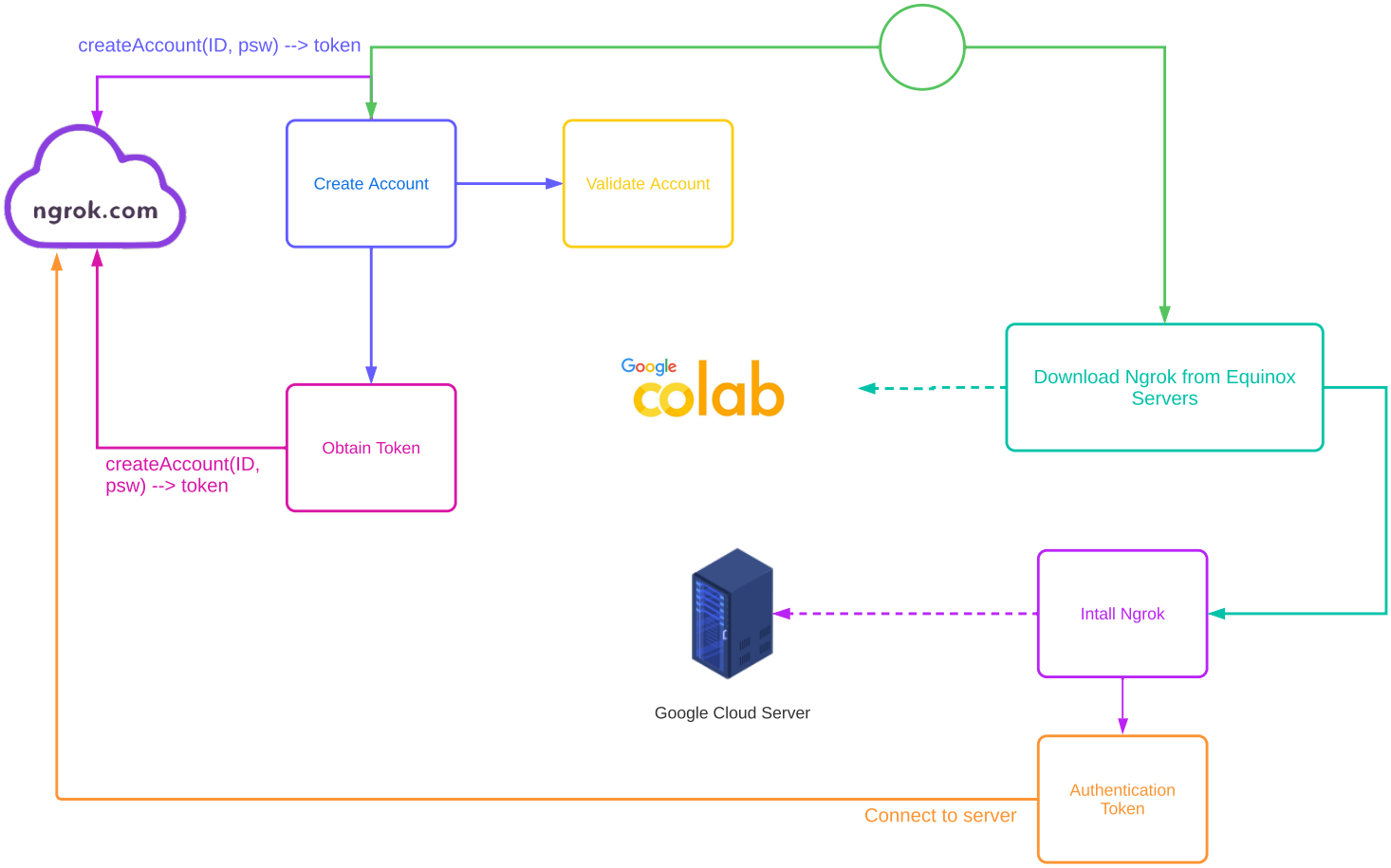

Diagram 3: Ngrok workflow

- Ngrok is a service that we use to create a kind of tunnel or cable between a running web server and a port on a local machine. This service provides us with a real-time web interface where we can check all HTTP traffic running through each of the tunnels. To give a specific user access to all the functions provided by Ngrok, an authentication token (authtoken) must be generated as it is required and is established through the configuration file or by means of the ngrok authtoken command.

Th most important thing of Diagram 3 is the authentication, because is one of the most crucial steps to be able to continue for the correct deployment for the final App.

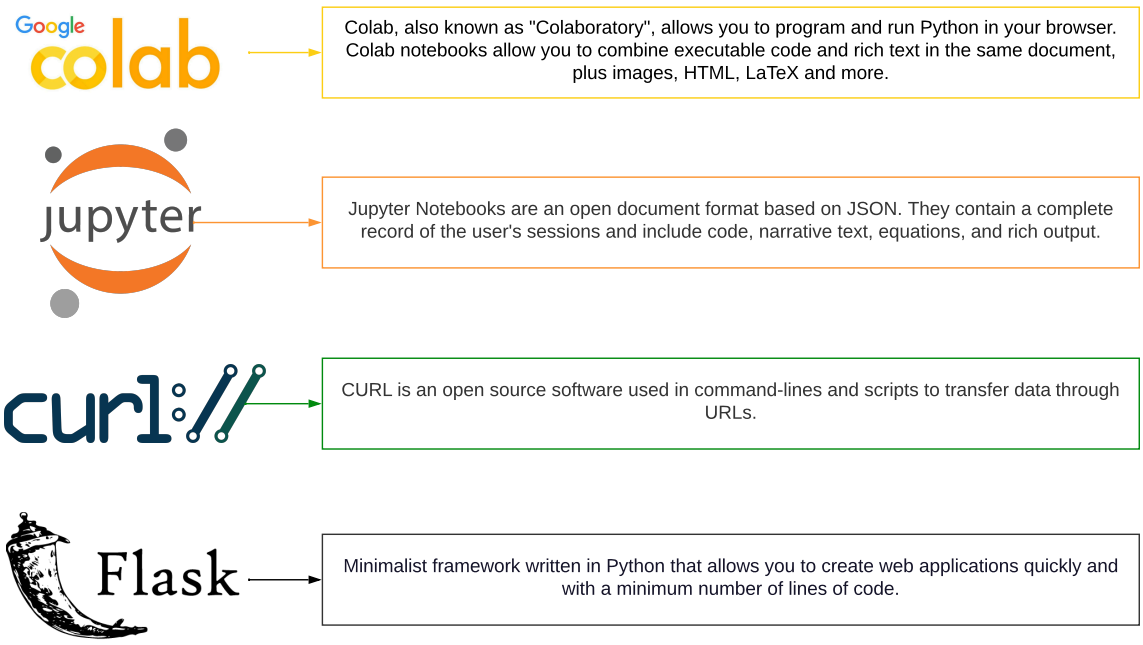

Diagram 4: Services providers

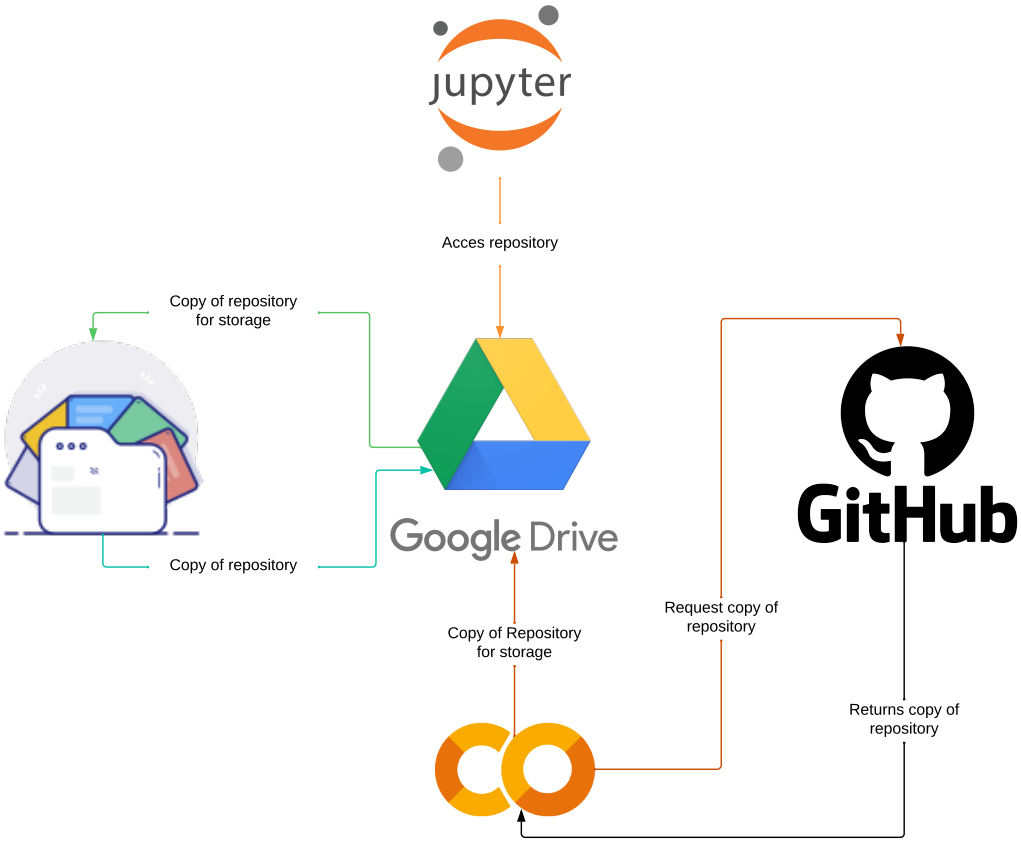

Diagram 5: Download process

With all the information understood is how we conducted the lab session where we performed the following series of steps, which allowed us to corroborate the actual operation of some of the services previously mentioned:

1. Create a Google Colab account and retrieve the data located in the following repository: https://gist.github.com/javieraespinosa/8d9e93cff61c675d74bd862975a84d5e
2. Install Azure CLI in Google Collab and verify that it has been installed correctly by verifying the version
3. Sign in to the Microsoft Azure account
4. Install ngrok on Google Colab and enter the authtoken
5. Clone the code example given in the tutorial
6. Install the code requirements
7. Run the sample code and launch it through a web application. Get the URL of the application and check that it works correctly
8. Check the log queue for recent requests
9. Remove resources from the application

If we put together all the diagrams and analyze each of the steps, we can see in a "visual" way how was the complete development of this practice, however we had reference to the tutorial that Azure shows on its official website, which is why when performing this series of steps compared to the tutorial provided directly by Azure there are some notable changes, below is a comparison between the Lab Session and Azure Tutorial to analyze if there are many differences and what they are. We need to understand that Infrastructure as a Service (IaaS)and Platform as a Service (PaaS) are great cloud tools that lead us build and deploy applications in an easy way. This cloud services has been more sophisticated and advanced to improve the time to give life to any App.

Comparison Azure Tutorial vs. Lab implementation

To host our application in Azure we created an Azure App Service web app in Azure, in Acure Tutorial they give us 3 different options (Azure portal, VS code, Azure CLI) to create the app, while in the Lab Session we just got one (Azure CLI).

Azure App services supports multiple methods to deploy our application code to Azure, in Azure Tutorial they focused on how to deploy the code from our local workstation to Azure, and in the Lab Session instead of having it in our local machine, that’s on a server provided by Google.

In the Lab Session we made use of Ngrok (install, configure, connect) contrary in Azure Tutorial we deploy directly the application in the web server. But in both of them we stream logs, because Azure App Service captures all messages output to the console to assist us in diagnosing issues with the application. print() is included to demonstrate this capability.

For the Lab Session we did some extra thing to explore a little bit Azure and its component. After all of this, we finished with the sample app, and that means we were able to remove all of the resources for the app from Azure, just to ensure we don't incur additional charges and keep our Azure subscription uncluttered. By removing the resource group we also removed all resources in the resource group and is the fastest way to remove all Azure resources for the app. For this in Azure Tutorial to make the Clean up resources is following a couple of steps inside the App, inversely in Lab Session we just used a simple command.

At the end we know both exercises/practices are closely because they do the same thing, or have the same purpose that is to deploy a Python web app to Azure App Services.

In a more general point, I could said that the Lab implementation has a preparing environment section where we prepared Google Collab by installing and configuring Azure CLI and Ngrok. These two things are completely different from Azure Tutorial because instead of installing on the local machine is on Google Server. But in both of them the Github repository is cloned and installed the python dependencies, but in Azure Tutorial a virtual environment for the App is created. For the test (running) in the Lab implementation we did it in Google Server, so we loaded and executed the same in Google Collab by configuring the sample for running Google Collab using the library flask_ngrok with the function run_with_ngrok(app), in the other hand, for Azure Tutorial the running is on the local machine. Another big difference was the creation of the app, cause in the Lab implementation we didn't do what is used for Azure Tutorial, that is made by creating the Web App in Azure to host the app using its own portal.

And for the Deployment is equally different. Since in the Lab implementation we deployed the code in a local folder using a command, meanwhile in Azure Tutorial was by using VS code and extensions pack provides by azure tools. For the last things, like the browse, cleanup was completely the same. Just the Streams Logs are a little bit different, but both made use of the command to do it. Just that on Azure Tutorial before been able to execute, we needed to configure Azure App Service to output logs to its Filesystem with a command. Another thing is that in the Lab implementation we have the redeploy section where can be used by !az webapp up command.

Glossary HO-1 Services as a unit of construction

Ngrok: Exposes local servers behind NATs and firewalls to the public internet over secure tunnels. It has 3 main steps (**1.-** Download and run to provide the port of a network service, **2.-** Connect to the Ngrok cloud service which accepts traffic on a public address, **3.-** Access from anywhere because traffic is relayed through to the Ngrok process running on the machine and then on to the local address you specified.)

Flask: Minimalist framework written in Python that allows you to create web applications quickly and with a minimum number of lines of code. It is based on the Werkzeug WSGI specification and the Jinja2 template engine and is licensed under the BSD license.

CLI: It stands for command line interface. It is a program that allows users to write text commands instructing the computer to perform specific tasks. Similar tasks can be performed as in the GUI but with minimal resources. A specific command can be used to target specific destinations with ease. You have full control over the system.

Notebook: Allow you to combine executable code and rich text in a single document, along with images, HTML, LaTeX and more. They are Jupyter notebooks hosted in Colab. It is a widely used tool in the machine learning community.

CURL: Is a software project consisting of a library and a file transfer-oriented command interpreter. It supports FTP, FTPS, HTTP, HTTPS, TFTP, SCP, SFTP, Telnet, DICT, FILE, LDAP and other protocols.

IaaS: Infrastructure as a Service is a bit of a departure from on-premises infrastructure. It is a pay-as-you-go service, where a third party provides you with infrastructure services, such as storage and virtualization, when you need them, through the cloud or the Internet.

PaaS: Platform as a service moves a little further away from full on-premises infrastructure management. In this case, the provider hosts the hardware and software on its own infrastructure and offers the platform to the user as an integrated solution, solution stack, or service over the Internet.

SaaS: Software as a Service also known as cloud application services, is the most comprehensive of the cloud computing options. It offers a comprehensive application that is managed by the provider, through a web browser.

References

Azure. *What is IaaS?* Retrieved from: https://azure.microsoft.com/es-mx/overview/what-is-saas/

Azure. *What is PaaS?* Retrieved from: https://azure.microsoft.com/es-mx/overview/what-is-saas/

Azure. *What is SaaS?* Retrieved from: https://azure.microsoft.com/es-mx/overview/what-is-saas/

MCCOY, L. *Microsoft Azure Explained: What It Is and Why It Matters*. CCB Technology. Retrieved from https://ccbtechnology.com/what-microsoft-azure-is-and-why-it-matters/

ngrok - secure introspectable tunnels to localhost. Ngrok.com. Retrieved from https://ngrok.com/product.

Vargas-Solar, G. *HO-1 Services as a unit of construction - Cloud Computing and Big Data*. Cloud Computing and Big Data. Retrieved from http://vargas-solar.com/cloud-bigdata/ho-1-services-as-a-unit-of-construction/