# Cloud and Big Data Project

Lecturer :
Dr. Boris Teabe
Prof. Daniel Hagimont

October 2024

**Students :**
Vu Trung Dung - ICT.2402008
Pham Gia Phuc - M23.ICT.010
Nguyen Tu Tung - M23.ICT.012

## Part I

# Achievement

## 1 VM Provisioning with Terraform

The first stage of the project focuses on employing Terraform to launch virtual machines. It starts by verifying if Terraform has been initialized before. If not, Terraform is initialized to set up the infrastructure management environment. Next, based on the provided $N\_SLAVES$ parameter, we provision the designated number of worker nodes, and one other virtual machine (VM) for the master node is deployed. After a successful VM start, the IP addresses of the machines are retrieved, and an Ansible inventory file is created, mapping the worker and master nodes appropriately (the master IP address is the first, and the remaining addresses are slave IP addresses). By doing this step, you can be sure that the recently built infrastructure is ready for additional orchestration and configuration.

## 2 Cluster Installation with Ansible

Ansible is used in the second section[1] to set up the virtual machines as a Hadoop/Spark cluster. We install Hadoop, Spark, and other required software using the prepared inventory file. We also configure the node names on the master and worker nodes to ensure that Hadoop and Spark are operational. All

of this is done by running an Ansible playbook. Ansible guarantees that the cluster is configured consistently and that all necessary services are deployed automatically. In addition, a connection to the nodes for configuration is made using the SSH private key.

## 3 Running the Spark Job

Running a Spark job is the third step of the project after the cluster is configured. Before anything else, we make sure there are no data conflicts by cleaning up any outdated output from prior runs. Subsequently, a further Ansible playbook is utilized to execute the Spark job, passing as parameters the paths to the JAR file, data file, and main class. This reduces the need to manually configure each job and enables the flexible execution of various jobs across the cluster. The job's output is saved in the output directory then it will be copied to the native server.

## 4 Stopping the VMs

Making sure that all resources are freed once the assignment is completed is the last step. We destroy the previously provisioned virtual machines (VMs) using Terraform once more, shutting down the infrastructure to prevent needless expenses. Effective resource management requires this cleanup phase, particularly when utilizing cloud or virtualized systems. In order to provide you an idea of how long the job will take to complete, we also compute the overall time at the end of the work.

# Part II
# Tools

## 5 How to execute the work

### 5.1 Step 1: Log in to Your User Ubuntu

Run the following command to switch to the ‘ubuntu‘ user:

```
su - ubuntu
```

### 5.2 Step 2: Clone the Code from GitHub

Make sure you have Git installed, and then run this command to clone the repository to your directory:

```
git clone https://github.com/Lib3Rt9/CloudAndBigData.git
```

### 5.3 Step 3: Navigate to the CloudAndBigData Folder

After cloning the repository, navigate to the project folder:

```
cd CloudAndBigData
```

### 5.4 Step 4: Run the Code

Run the following shell script to start the process:

```
sh begin.sh
```

**Explanation of the Command in begin.sh**

Inside ‘begin.sh‘, the command specifies the number of slaves, the location of the ‘.jar‘ file, and the location of the ‘datafile.txt‘. Here is the command:

```
bash start.sh 2 ./examples/wc.jar ./examples/filesample.txt WordCount
```

This command starts a process with 2 slaves, using the WordCount example from the specified ‘.jar‘ and ‘filesample.txt‘.

For your further info, please click **here** to see the demo.