LAB 1 - WEEK 1 AND 2
SETTING UP THE DEV ENVIRONMENT - DOCKER

Setup Dockerized BDT Lab Environment on your machine

1. Machine Pre-requisites

- A machine with 64 bit OS and more than 8 GB RAM (Total of 16GB+ RAM is preferred)
- At least 30-40 GB free space on hard disk
- Virtualization (VTx) parameter must be enabled in the BIOS settings of your machine otherwise Docker will not start!

2. Docker Installation

Install Docker Desktop from the official *Docker website*. Download the version suitable for your machine.

Giving you detailed instructions for Docker installation is not part of this lab.

Note: *The course lecture videos use the Cloudera QuickStart VM for lab demos. You may use the same VM by following the instructions from another document. However, since it is based on CentOS 6.7 (now EOL), you may face issues installing new software during the course Project. Also, the Cloudera QuickStart VM can be difficult to run on newer MacBooks.*

3. Configuring Resource Limits for Windows

Unlike the Cloudera QuickStart VM where you manually assign RAM in VirtualBox or VMware, WSL2 (Windows Subsystem for Linux) that Docker uses, shares resources dynamically with your OS. If left unconfigured, the lab environment may consume a large portion of your system memory, causing your IDE, browser or the OS itself to become unresponsive.

To ensure a smooth experience, you should manually limit the resources available to WSL2.

***1.* Create a Configuration File**

1. Open your Windows User Folder (e.g., C:\Users\YourName).
2. Create a new text file named `.wslconfig` (ensure there is a dot at the beginning and no .txt extension at the end).
3. Open the file in Notepad and paste the following configuration:

```
[wsl2]
# Recommended minimum: 8GB RAM and 2 CPU cores
# If your laptop has total 32GB RAM, you can increase this to 12GB.
memory=8GB
processors=2

# This prevents the WSL VM from keeping memory it no longer needs
autoMemoryReclaim=gradual
```

2. Apply the Changes

1. For these settings to take effect, you must restart the WSL subsystem:
2. Open Windows **PowerShell** as an Administrator.
3. Run the command: `wsl --shutdown`
4. Restart Docker Desktop.

3. Monitoring Resource Usage

Once the lab Docker containers are running, you can monitor how much memory the Java daemons (Hadoop, Hive, HBase) are using by running the jps command inside the container terminal.

***Note for Mac Users:** You do not need a .wslconfig file. Instead, go to Docker Desktop Settings > Resources and adjust the sliders for CPUs and Memory to at least 8GB and 2 Cores.*

4. Compose Hadoop Image

1. Create a new folder *cs523-bdt* anywhere on your machine where there's enough space available. Open Terminal in this folder.
2. Get the required files by cloning the git repository:

   `git clone https://github.com/himrudula/cs523-bdt.git`
3. Create an empty subfolder named `my_code` inside the newly created folder cs523-bdt for keeping your work to access from the container.
   (All the program jars that you would like to execute on the YARN cluster should be inside this `my_code` folder.)
4. Open the yml file. You need to put the password for the custom image.
   a. Go to the service named *cs523bdt-lab* in the .yml file.
   b. Paste the following inside the *environment* tag.

      ```
      - CLASS_PASS=DEcs523bdt_Feb2026
      ```
5. Make sure Docker Desktop is running. Then run `docker-compose up -d` from the newly created folder cs523-bdt.

   This command downloads the necessary images and starts a pseudo-distributed Hadoop cluster in the background (-d means detached mode). Be patient as it'll take some time to download all the images. You'll have all the HDFS and YARN services running once the containers start.
6. You can check that the containers are running with `docker ps`.
7. Stop the Zookeeper, Hive and Kafka containers as we won't be using those for the MapReduce labs. Stopping them will save you some RAM. You can start them when we study those tools and need them for labs.

5. Verify Hadoop Services

You can access the web UI for various services as shown below. Make sure to bookmark these links in your browser so that you can access them easily for future labs:

1. HDFS Name Node: http://localhost:9870
2. YARN Resource Manager: http://localhost:8088/
3. HBase Master UI: http://localhost:16010
4. Spark UI: http://localhost:4040

Note that HBase master UI will not work as we've not started that service in the container. But still bookmark the link. We'll see how to start that service when we'll learn HBase.

Also, note that Spark UI will only work while you are running Spark jobs as we are not using Spark History Server to save our machine resources.

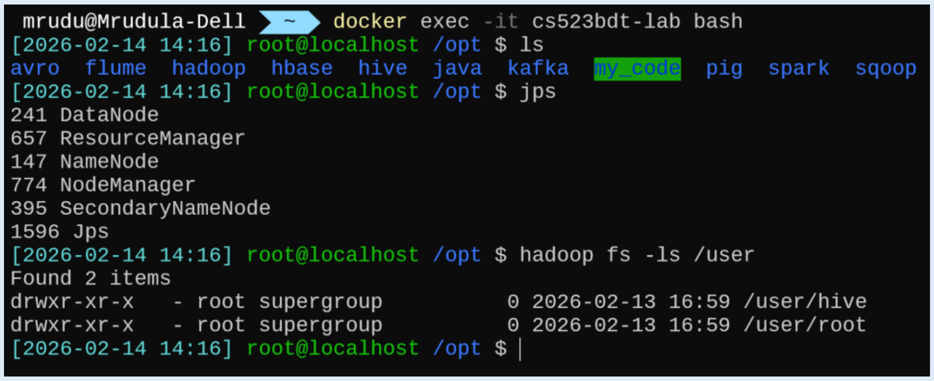
6. Access the Shell of the Hadoop Container

1. To enter the running container's shell, run the following command:

   docker exec -it cs523bdt-lab bash

   Note: *cs523bdt-lab* is the name of the container. You can alternatively use container-id. Use the name that matches your container.

2. You should be now inside the container where we've setup the pseudo-distributed mode for Hadoop. You are required to run all the commands from this prompt.



```
 mrudu@Mrudula-Dell  ~  docker exec -it cs523bdt-lab bash
[2026-02-14 14:16] root@localhost /opt $ ls
avro  flume  hadoop  hbase  hive  java  kafka  my_code  pig  spark  sqoop
[2026-02-14 14:16] root@localhost /opt $ jps
241 DataNode
657 ResourceManager
147 NameNode
774 NodeManager
395 SecondaryNameNode
1596 Jps
[2026-02-14 14:16] root@localhost /opt $ hadoop fs -ls /user
Found 2 items
drwxr-xr-x   - root supergroup          0 2026-02-13 16:59 /user/hive
drwxr-xr-x   - root supergroup          0 2026-02-13 16:59 /user/root
[2026-02-14 14:16] root@localhost /opt $ |
```

3. You can type "exit" at the prompt to come out of the container shell.
4. Once you are done with your work, you can run the following command to properly stop all the Hadoop services: *docker-compose down*
5. If you want to remove all the created volumes as well then run the command:

   docker-compose down -v

6. When you need the pseudo-distributed single node Hadoop Docker environment again, you can simply run *docker-compose up -d* and it'll simply start the containers again.
7. When you are inside the container shell, type `jps` command to see the running services.

> This custom Docker image provides a preconfigured environment with all core Big Data services required for the course. This is a single-node, pseudo-distributed setup, which does not reflect the performance or scalability of a full cluster but is sufficient for learning and completing course labs.

If you do not plan to use a VM or Docker then you'll need to install Hadoop and other tools on your own. There may be compatibility issues with different versions of different tools, so refer this page to see which version of which tool works well together.

Homework Questions

Questions

Answer and submit the following questions once the above setup is done.
Numbers in square brackets indicate the points for each question.

1) [2] **In the newly set up Hadoop Docker container, what does the $HOME environment variable represent?**

2) [2] **What does sudo mean in a Unix/Linux environment, and when is it typically used?**

3) [6] **Write and briefly explain any 6 Unix/Linux commands.**
Execute each command in the new Docker container and paste **screenshots of the outputs**.
You may choose from the following commands:
cd, touch, mkdir, rm, clear, ls, pwd, mv, cat, tail, echo

These questions are intended to help you become familiar with basic Unix/Linux concepts, as Linux is the primary operating system used throughout this course.