Assisted Practice 12.5: Abstraction

Problem Scenario: Write a program to demonstrate abstraction using classes, objects, and methods.

Objective: In this demonstration, we will learn how to perform abstraction.

Expected Output:

Ingredients: tomato onion cottage cheese

Taste: Good

Ingredients: chicken meat beef

Taste: Good too

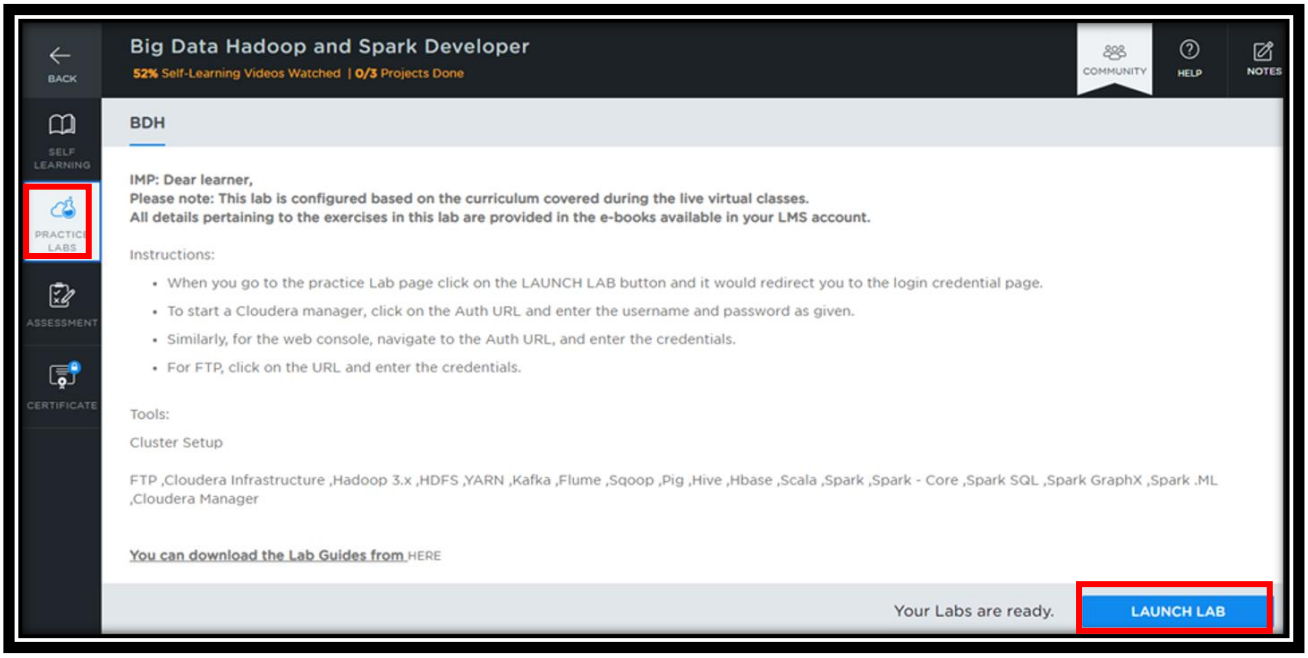
Steps to Perform:

Step 1: Log in to your LMS account.

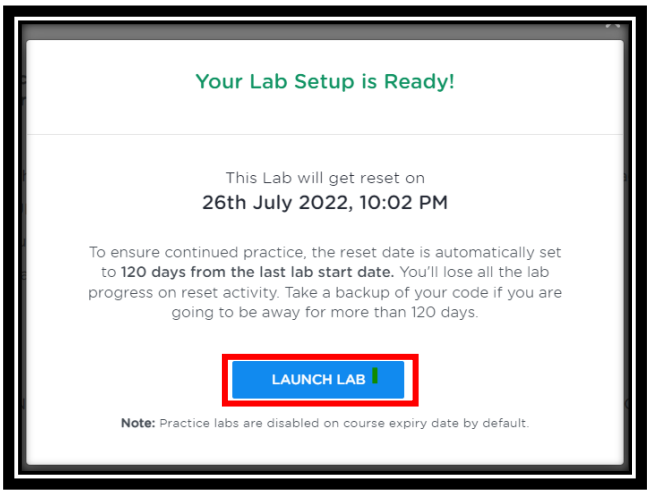
Step 2: Open the course "**Big data Hadoop and Spark developer**".

(Note: The course name reflects depending on the program purchased)

Step 3: On the left side click on the "**PRACTICE LABS**" tab and click on the "**LAUNCH LAB**" button.



Step 4: Again, click on the "**LAUNCH LAB**" button.



STEP 5: Click on "**Webconsole**" and click on the "**Auth Url**".

Step 6: Copy the "**Username**" and the "**Password**" provided to log in to the Web console.

Step 7: Paste the "**Username**" and the "**Password**" on the console and click on enter.

Note: The password will not be visible when pasted on the console

Step 8: Create a python file**.**

Command:

vi abstractAP.py

The below screen appears:

Step 9: Perform the tasks.

9.1 Import the necessary libraries.

Command:

```
from abc import ABCMeta,abstractmethod
```

9.2 Create a base class with the name Food that contains abstract methods.

Command:

```
class Food():
 __metaclass__ = ABCMeta
 @abstractmethod
 def ingredients(self):
  pass
 def taste(self):
  pass
```

9.3 Create two derived classes.

Command:

```
class Veg(Food):
```

```
    def ingredients(self):
      print("tomato","onion","cottage cheese")
    def taste(self):
      print("Good")
class Nonveg(Food):
    def ingredients(self):
      print("chicken","meat","beef")
    def taste(self):
      print("Good too")
```

9.4 Create objects for the derived class and call the non-abstract methods.

```
obj = Veg()
obj.ingredients()
obj.taste()

obj2 = Nonveg()
obj2.ingredients()
obj2.taste()
```

```
from abc import ABCMeta,abstractmethod
class Food():
  __metaclass__ = ABCMeta
  @abstractmethod
  def ingredients(self):
   pass
  def taste(self):
   pass

class Veg(Food):
  def ingredients(self):
    print("tomato","onion","cottage cheese")
  def taste(self):
    print("Good")

class Nonveg(Food):
  def ingredients(self):
    print("chicken","meat","beef")
  def taste(self):
    print("Good too")

obj = Veg()
obj.ingredients()
obj.taste()

obj2 = Nonveg()
obj2.ingredients()
obj2.taste()
~
```

9.5 Run the code.

Command:

python3 abstractAP.py

```
[testdemomay1301mailinator@bdh-cluster2-edgenode10 ~]$ python3 abstractAP.py
tomato onion cottage cheese
Good
chicken meat beef
Good too
```