
System Requirements Specification Index

For

Tumor analysis Usecase

Brain tumor analysis using MI L3

1.0

Step to access the work environment

Step 1 use the URL to login provide the username and password

PySpark-Data Analytics-Employee-L1-VIR-Template

180 Mins

1 Sections

1 Skills

1 Questions

2 Total Attempts

0

60% | Cut Off

100%

System Requirements

- Recommended Browser (Chrome, Safari, Etc)
- Javascript should be enabled in the browser

Link Validity and Cut-off Details

- Link Validity Start Date and Time - 6/9/2024, 5:47 PM
- Cut Off Date and Time - 30/9/2025, 2:50 PM

YAKSHA

Registration Details


First Name *

Last Name *

Email *

Phone (Optional)

☐ I'm not a robot


reCAPTCHA
Privacy - Terms

Start

Description

PySpark-Data Analytics-Employee-L1-VIR-Template

Step 2 Click on the launch assessment Environment

Speed Test Avg: 4.40Mbps Live: 4.40Mbps Time Remaining: 00:00:00 Final Submit

Question

Data Analytics-Employee-VIR-

Instructions

Introduction
This is a project-based assessment, in which you will be provided with a template code to work. You will be provided with a pre-configured Virtual Machine (VM) to develop the case study.

Launch
You can launch VM by clicking "Launch Assessment Environment". It will take around 5-10 mins to launch the VM.

Configuration
Once launched, if you see any configuration screen, skip it.

Cloning
Once VM is up, it will take another 30 sec-1 min to clone your project template on desktop of your VM. Please wait till then.

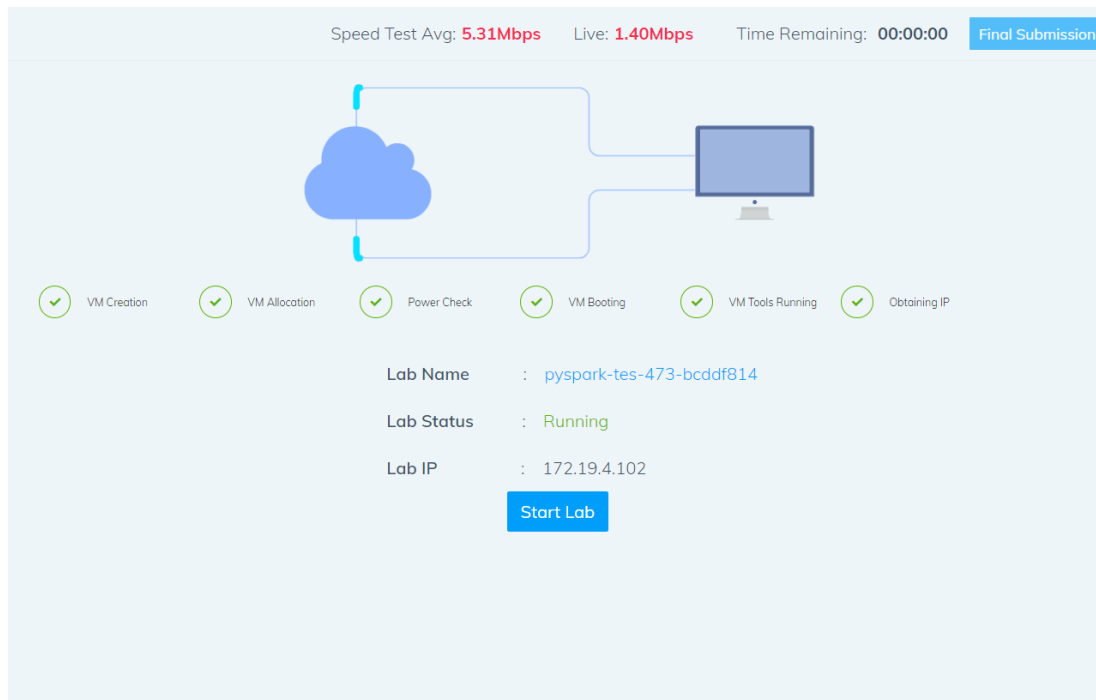
Document
The project will be cloned in a folder, named same as your email ID. The folder contains template code and case study document. You are required to open the case study document and thoroughly go through it to understand project and mandatory process.

Development
To develop use case all IDEs, Database required are available in VM. There is a README file on desktop containing any credentials you need.

Submission
Before you do final submission of your code there are 2 mandatory things you have to follow, else your evaluation result would be affected

1. RUN TEST CASE (AS MENTIONED IN DOCUMENT)
2. PUSH CODE TO GIT (AS MENTIONED IN DOCUMENT)

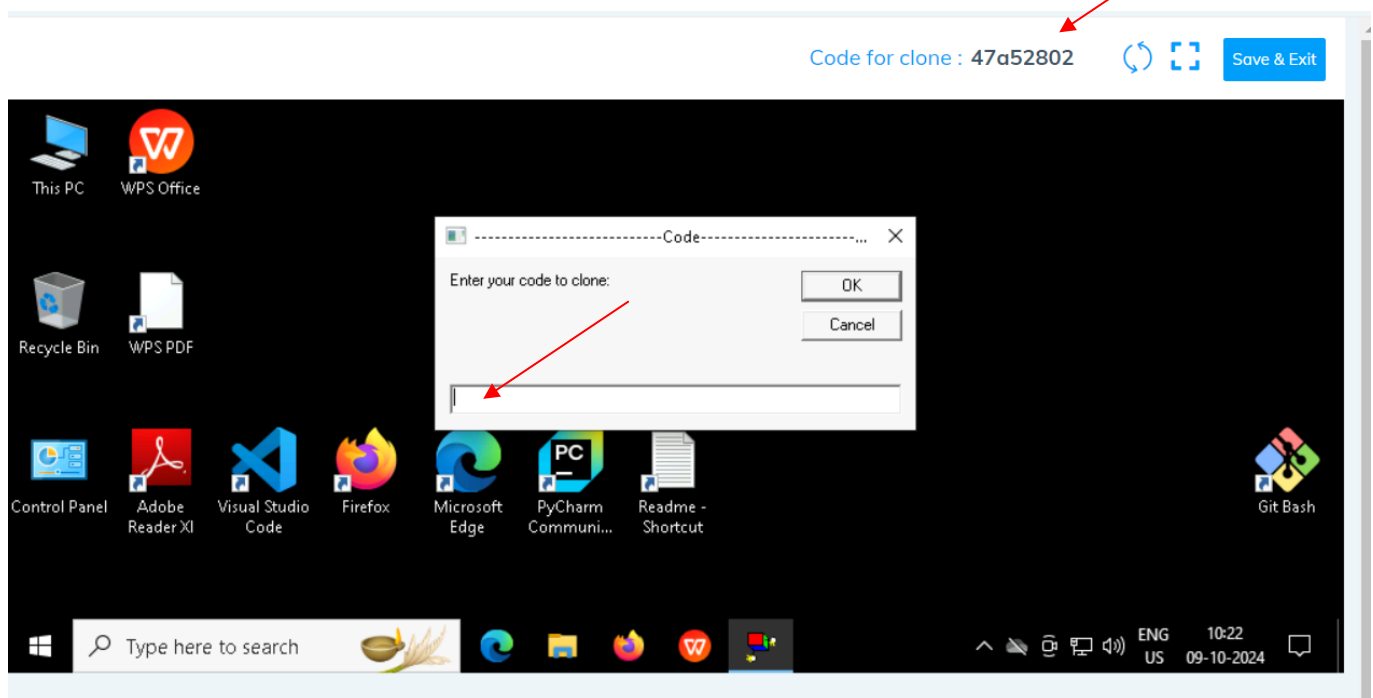
Launch Assessment Environment



Step 3 Click on the start lab button

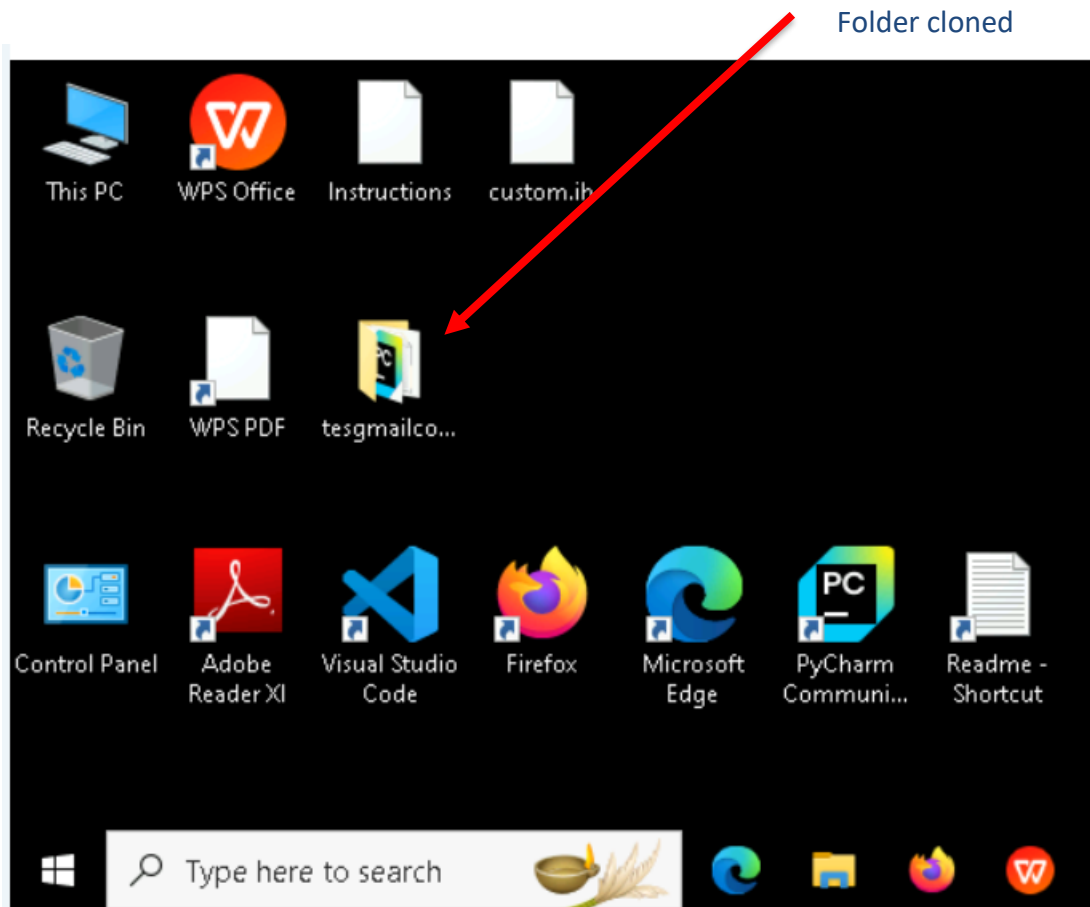
Step 4 you will get a window you need to type the code from that top corner

- You need to type the code in the window . It will take few minutes to start the window

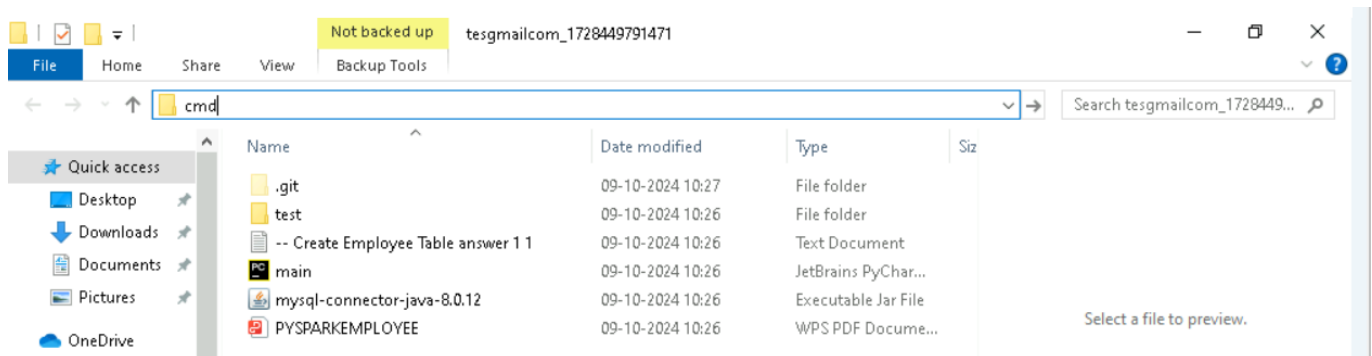


Click on ok

Step 5 after few seconds we can see that the your folder is cloned in the desktop .

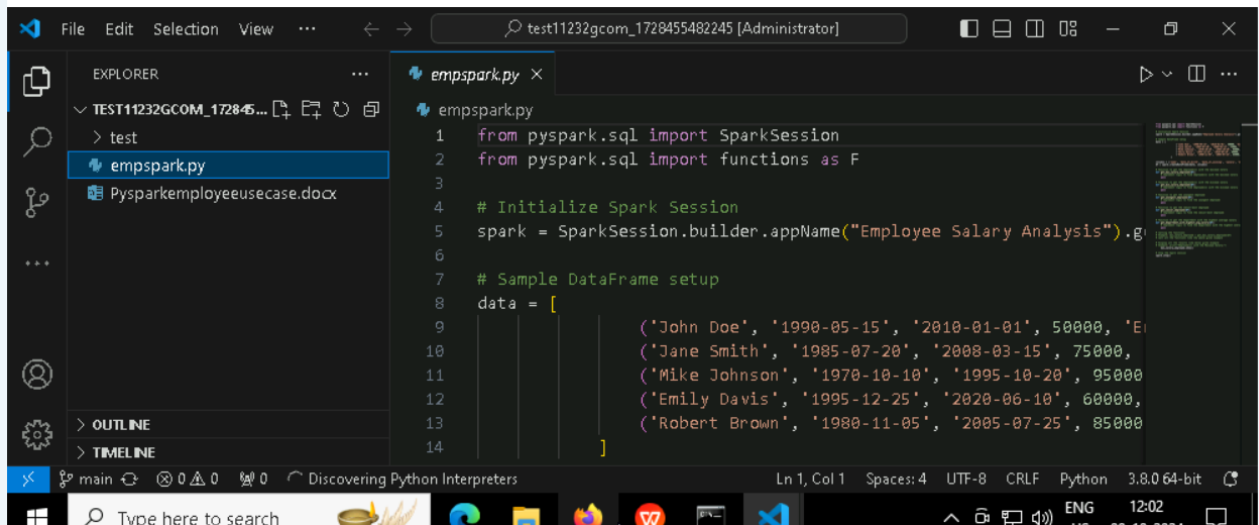


Step 6 go inside the folder type cmd in the top of the file explorer



- Type **code**. And hit enter you can see that workspace is opened in the visual code





- You can see that workspace is ready to code

Note Please only work with visual code not with any other IDE

- In the folder cloned you will have all the project files needed .

Problem Statement : **Brain tumor analysis**

Description : Use relevant methods operations to perform specified activities which are given in the instructions.

Medical imaging, such as MRI scans, provides detailed images of internal body structures, making it a critical tool in diagnosing brain tumors. However, manual analysis of these images is time-consuming and prone to human error. AI-powered systems can analyze thousands of images in seconds with remarkable accuracy, making them invaluable in medical diagnostics.

The program leverages:

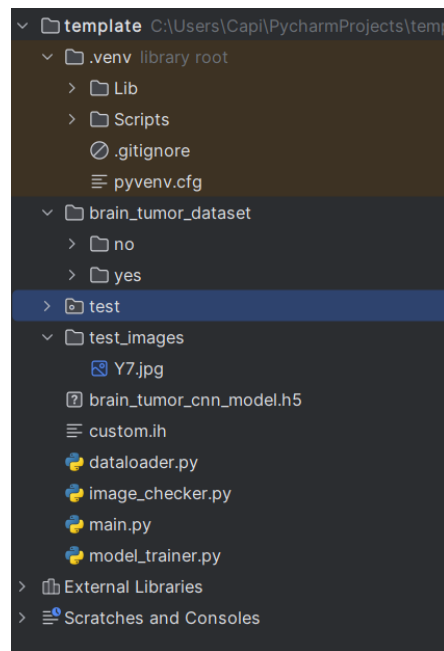
1. MRI Scans: Grayscale images of brain scans.
2. Convolutional Neural Networks (CNN): A deep learning model designed to process image data effectively.
3. Python Programming: Frameworks like TensorFlow and Keras for model development, training, and evaluation.

This program utilizes a Convolutional Neural Network (CNN) model to classify brain MRI images into two categories:

- Tumor: Indicates the presence of a brain tumor.
- No Tumor: Indicates the absence of a brain tumor.

The goal is to automate the process of brain tumor detection using image processing and deep learning techniques, ultimately assisting medical professionals in their diagnosis.

Project structure



data_loader.py: Handles loading and splitting the dataset into training and testing sets and provides a data summary.

image_checker.py: Contains image preprocessing and prediction functions for a pre-trained model.

main.py: Orchestrates the overall workflow, including loading data, training the model, evaluating it, and using it for predictions.

model_trainer.py: Contains the implementation for creating, training, and evaluating a CNN model.

Note

- The dataset folder contains the images datasets
- You need to save create the model with the same name .
- You need to complete all the code in the existing file .

Warning

- To no change any file names or images names

Write the python code to complete the steps

1. Define constants for paths:
2. `DATA_PATH = "brain_tumor_dataset"`
3. `MODEL_PATH = "brain_tumor_cnn_model.h5"`
4. `TEST_FOLDER = "test_images"`
5. Implement the main steps:
6. Load dataset using `load_data(DATA_PATH)`.
7. Split dataset with `split_data()`.
8. Print a summary using `data_summary()`.
9. Create model with `create_cnn_model()` and train using `train_model()`.

10. Save the trained model to MODEL_PATH.
11. Evaluate the model using evaluate_model().
12. Count total images in the dataset using get_image_count(DATA_PATH).
13. Predict images in test_image using predict_test_folder(MODEL_PATH, TEST_FOLDER).

Questions to be answered using the code

1. Write the code to detect the expected count of training samples in the dataset .
- 2 Write the code to detect the expected count of testing samples in the dataset.
- 3 Write the code to verify the expected number of training epochs during model training.
4. Write the code to check if the model file is successfully created after training.
5. Write the code to classify the test image Y7.jpg and verify the expected result.

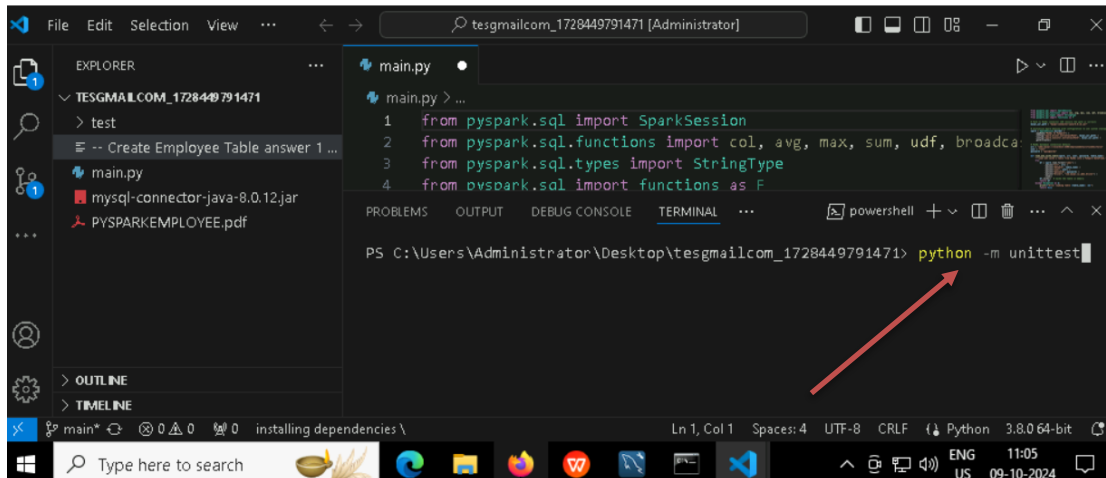
Execution Steps to Follow:

1. All actions like build, compile, running application, running test cases will be through Command Terminal.
2. To open the command terminal the test takers, need to go to Application menu (Three horizontal lines at left top) -> Terminal -> New Terminal
3. This editor Auto Saves the code
4. If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B-command compulsorily on code IDE. This will push or save the updated contents in the
5. internal git/repository. Else the code will not be available in the next login
6. These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.
7. To launch application:
Python main.py
To run Test cases:
python -m unittest

Screen shot to run the program

To run the application with all the libraries

- `python -m pip install --upgrade pip`
- Python install tensor flow
- Python main.py



To run the testcase

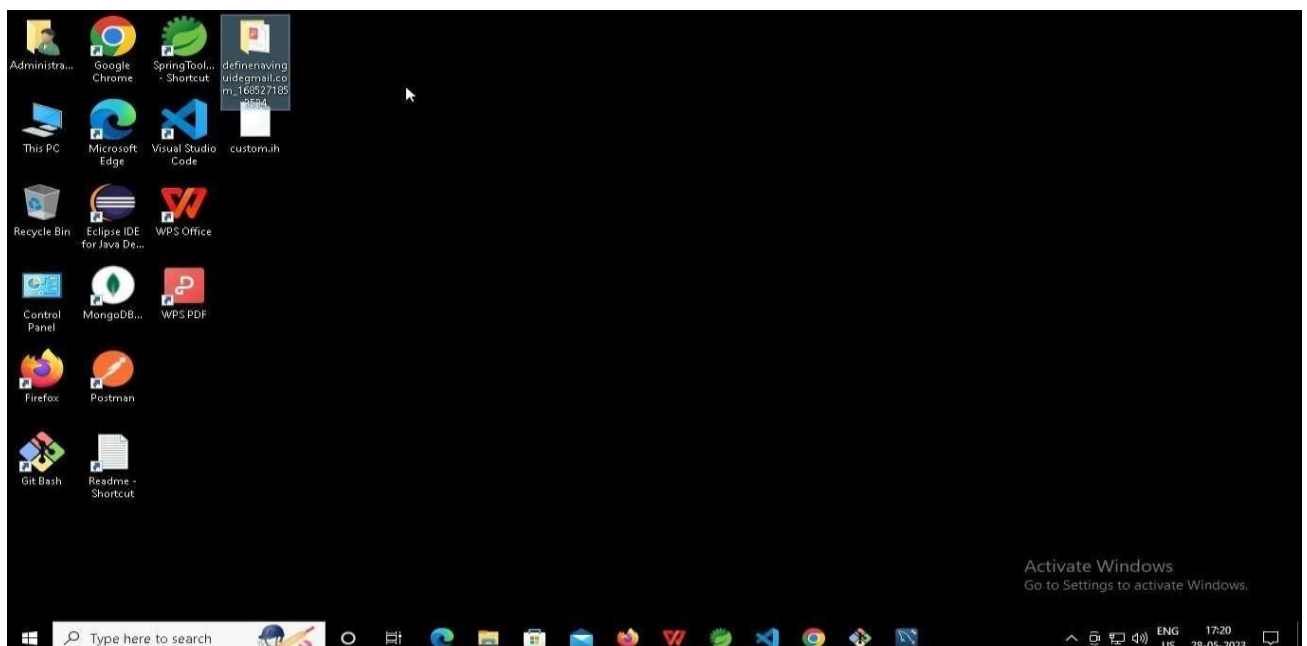
- `Python -m unittest`

Screenshot to push the application to github

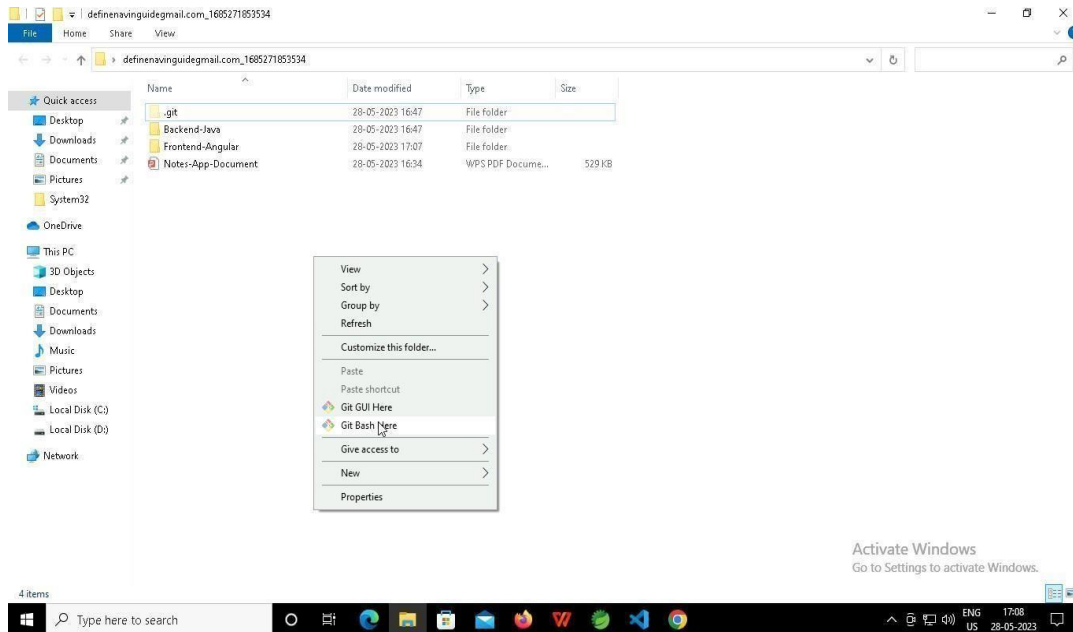
-----X-----

You can run test cases as many numbers of times and at any stage of Development, to check howmany test cases are passed/failed and accordingly refactor your code.

1. **Make sure before final submission you commit all changes to git.** For that open theproject folder available on desktop



a. Right click in folder and open Git Bash



b. In Git bash terminal, run following commands

c. git status

```
Administrator@2a5ee7ad258f58c MINGW64 ~/Desktop/tesgmailcom_1728449791471 (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        deleted:    templateespark.py

no changes added to commit (use "git add" and/or "git commit -a")
Administrator@2a5ee7ad258f58c MINGW64 ~/Desktop/tesgmailcom_1728449791471 (main)
$
```

d. git add .

```
Administrator@2a5ee7ad258f58c MINGW64 ~/Desktop/tesgmailcom_1728449791471 (main)
$ git add .
```

e. git commit -m "First commit"

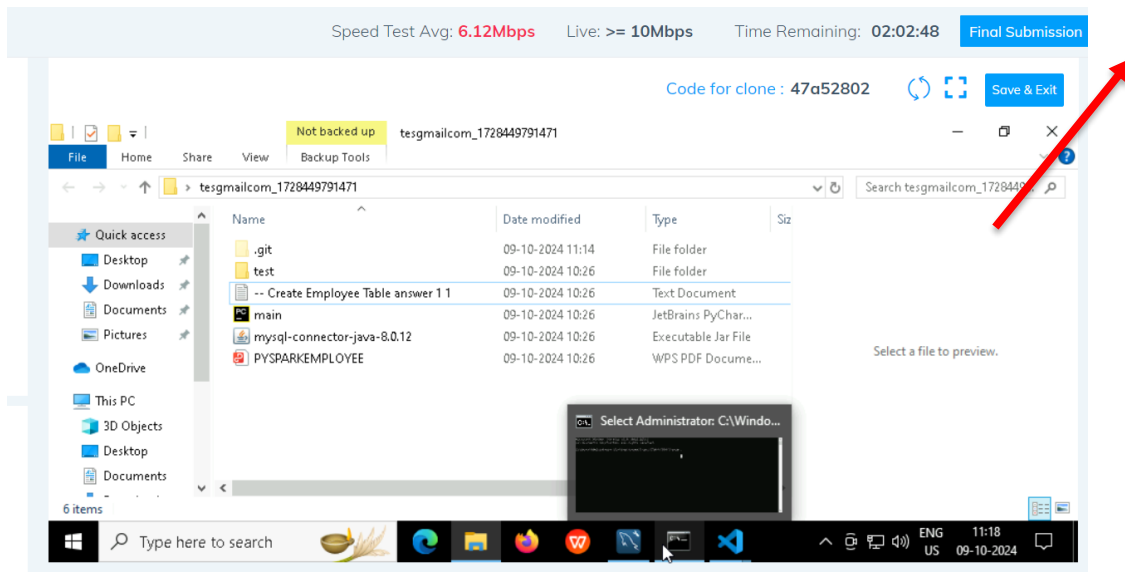
(You can provide any message every time you commit)

```
Administrator@2a5ee7ad258f58c MINGW64 ~/Desktop/tesgmailcom_1728449791471 (main)
$ git commit -m "first commit"
[main f97ce24] first commit
1 file changed, 91 deletions(-)
delete mode 100644 templatespark.py
```

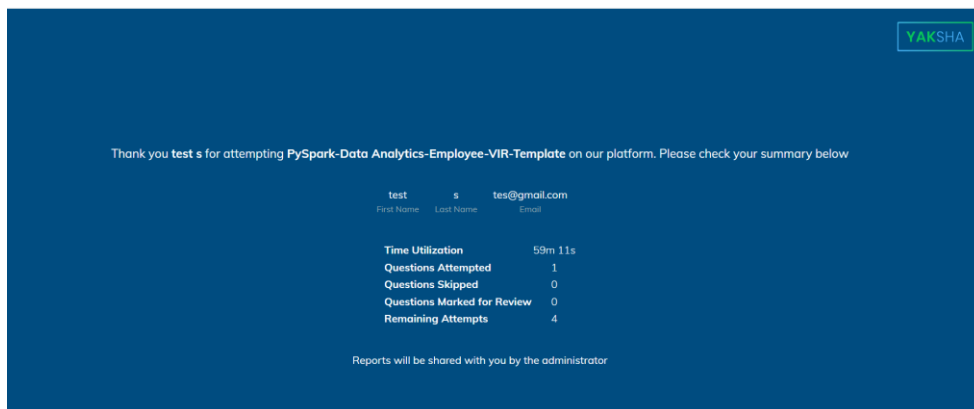
f. git push

```
Administrator@2a5ee7ad258f58c MINGW64 ~/Desktop/tesgmailcom_1728449791471 (main)
$ git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 212 bytes | 212.00 KiB/s, done.
Total 2 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/IIHTDevelopers/tesgmailcom_1728449791471.git
   a1c1905..f97ce24  main -> main
```

After you have pushed your code Finally click on the final submission button



You should see a screen like this you will have to wait for the results . after getting this page you can leave the system



-----X-----