
System Requirements Specification Index

For

Machine learning Usecase

Bank fraud detection with ML L2

1.0

Step to access the work environment

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

PySpark-Data Analytics-Employee-VIR-Template

180 Mins

1 Sections

1 Skills

1 Questions

5 Total Attempts

70% | -CutOff

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 - 16/9/2024, 7:03 PM
- Cut Off Date and Time - 30/9/2025, 4:05 PM

Registration Details

First Name *

Last Name *

First Name

Last Name


Email *

Phone (Optional)

Email

Phone

☐ I'm not a robot


reCAPTCHA
Privacy - Terms

Start

Description

PySpark-Data Analytics-Employee-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

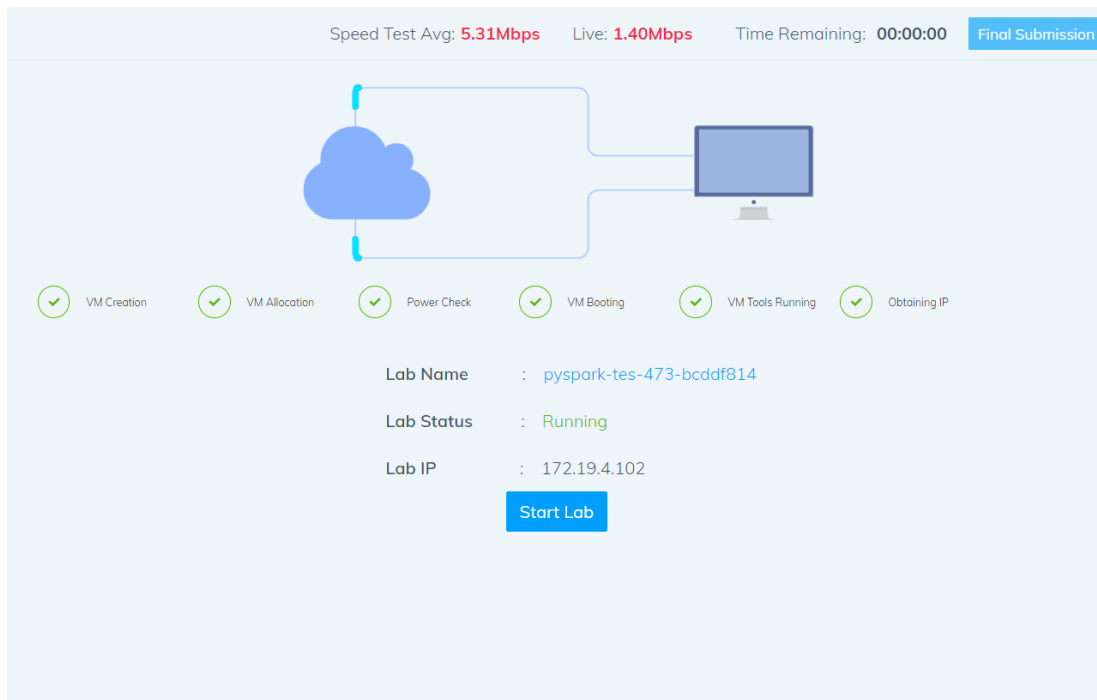
To develop use case all IDEs, Database required are available in VM. There is a README file on desktop containing any credentials you need.

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)

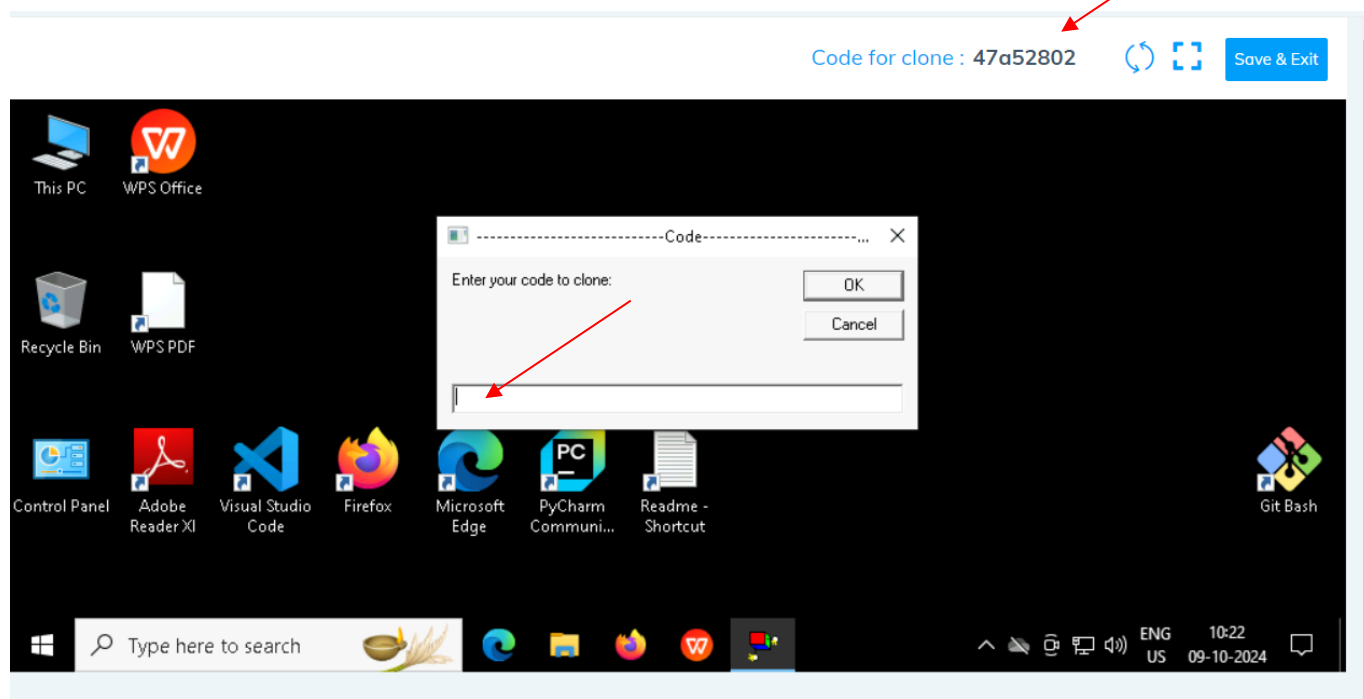
Launch Assessment Environment



Step 3 Click on the start lab button

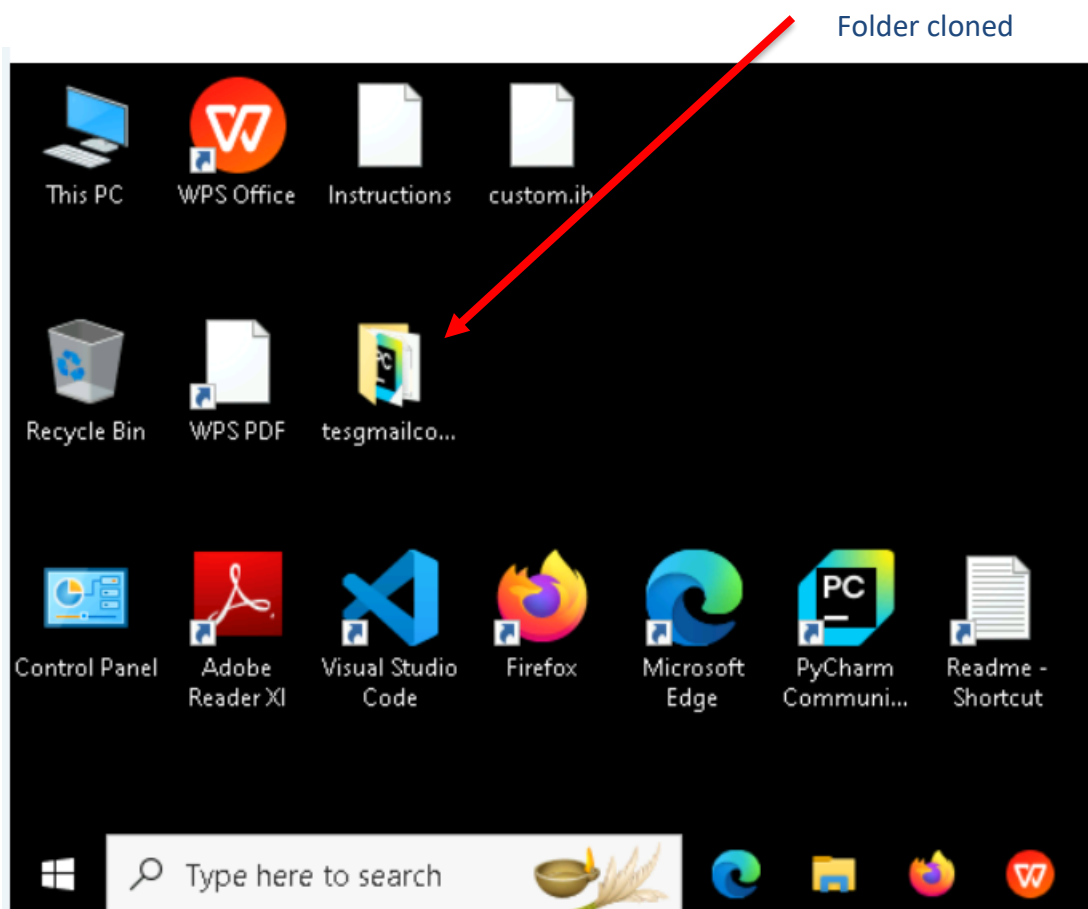
Step 4 you will get a window you need to type the code from that top corner

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

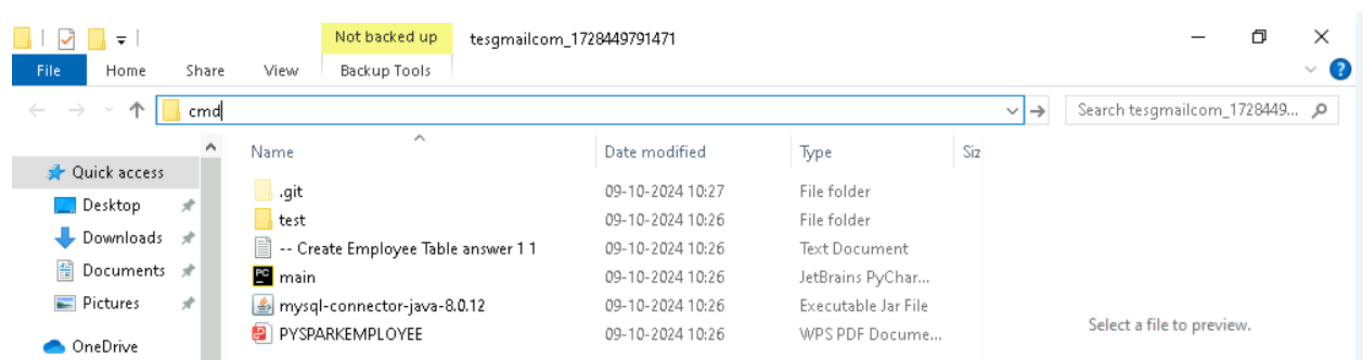


Click on ok

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

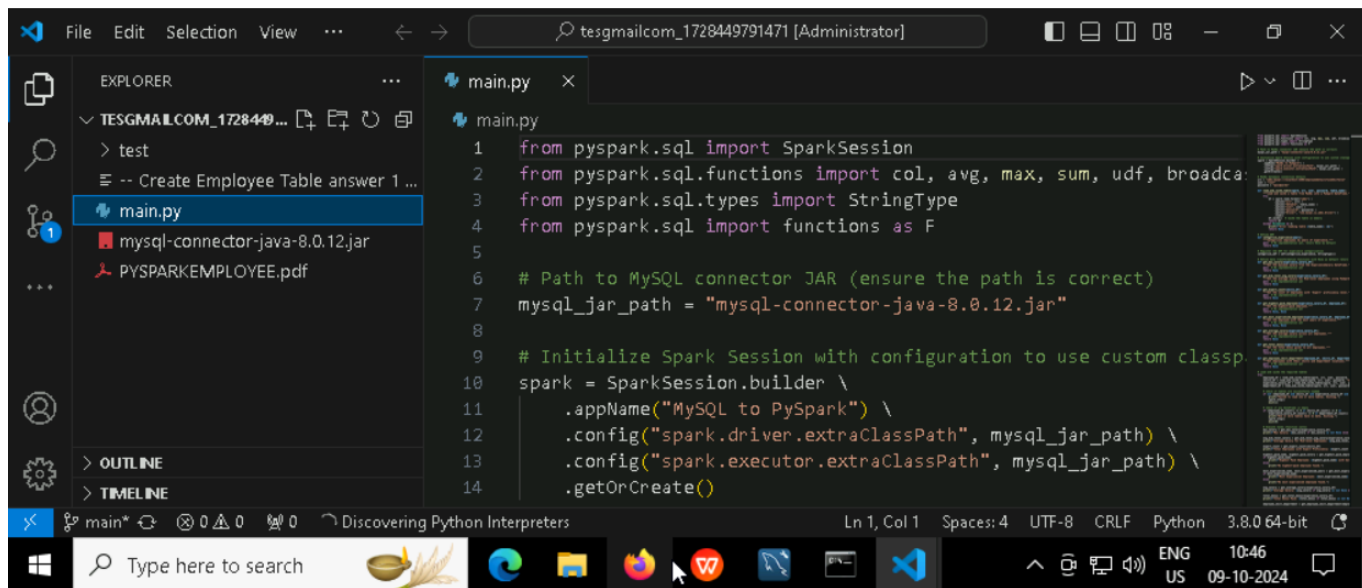


Step 6 go inside the folder type cmd in the top of the file explorer



- 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 : **Bank fraud detection**

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

Introduction

A large retail bank with millions of customers experiences a rising number of fraud cases involving unauthorized transactions, phishing attacks, and account takeovers. To mitigate these risks, the bank deploys an AI-powered Fraud Detection System integrated with transaction monitoring and behavioral analysis.

They begin their journey with a dataset containing transaction details, carefully preprocessing it to ensure accuracy and reliability. By encoding categorical features and scaling numerical data, they prepare the foundation for robust model training.

Next, they address the critical issue of class imbalance, where fraudulent transactions represent only a small fraction of the data. Using SMOTE, the scientist balances the dataset, ensuring the model can detect fraud effectively. With the preprocessed and balanced data, they train an XGBoost classifier, fine-tuning it to identify subtle patterns indicative of fraudulent behavior. After training, the model is evaluated, revealing its ability to distinguish fraud with high precision and recall, backed by a strong ROC-AUC score.

Objective

To detect fraudulent transactions using machine learning, preprocess data for model training, and perform analytical reporting based on fraud data.

preconditions

1. Dataset available in CSV format.
2. Python environment set up with the required libraries installed:
 - pandas
 - numpy
 - sklearn
 - xgboost
 - imbalanced-learn
 - joblib

Postconditions

- Trained XGBoost model saved as a file.
- Analytical insights on fraudulent transactions available

Dataset Information

The dataset used for this use case is a fictionalized credit card transaction dataset designed for fraud detection. It contains the following key attributes:

- **TransactionID:** A unique identifier for each transaction.
- **AccountID:** A unique identifier for the account involved in the transaction.
- **TransactionType:** The type of transaction (e.g., "Online", "POS").
- **Location:** The geographical location where the transaction occurred.
- **Merchant:** The merchant involved in the transaction.
- **TransactionAmount:** The monetary value of the transaction.
- **IsFraud:** A binary flag indicating whether the transaction is fraudulent (1) or not (0).

Steps to follow

1. Load and Preprocess Data
 - Load the dataset from a CSV file.
 - Encode categorical features and standardize numerical features.
2. Balance Data
 - Use SMOTE to address class imbalance.
3. Train Model
 - Train an XGBoost classifier on the balanced dataset.
4. Evaluate Model
 - Generate performance metrics: Confusion Matrix, Classification Report, and ROC-AUC score.
5. Save Model
 - Save the trained model using joblib for future use.
6. Perform Analytics
 - Analyze the dataset for key insights such as high-risk locations and transaction types.

Solve these Questions

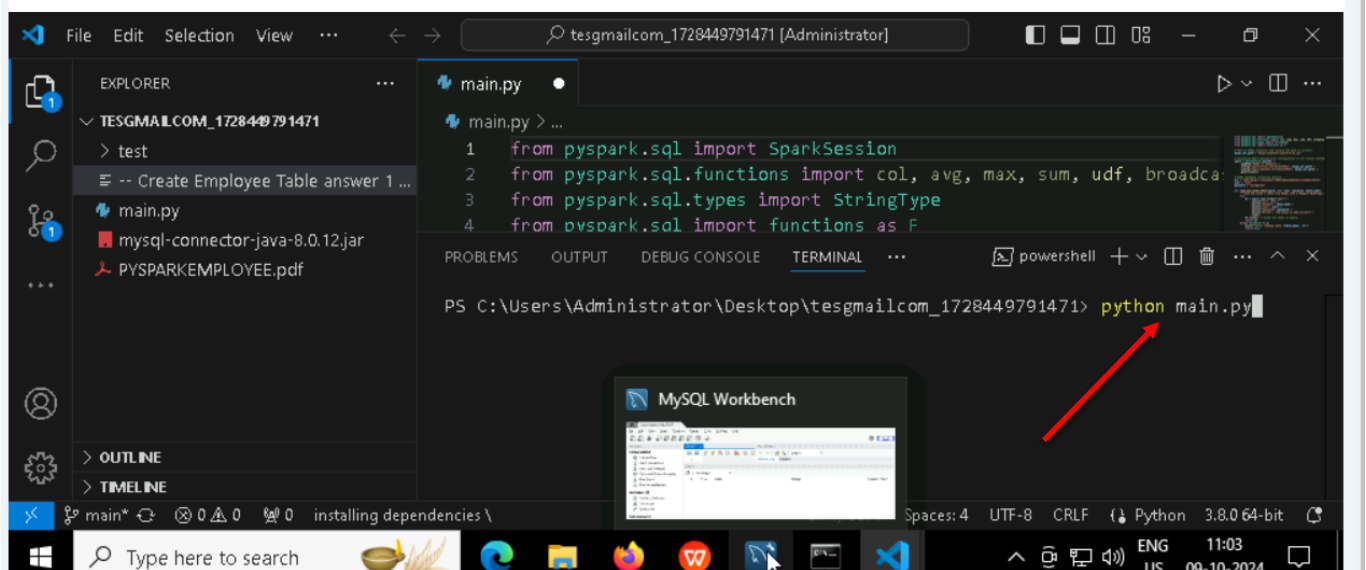
1. How many unique accounts are associated with fraudulent transactions?
2. What is the total number of fraudulent transactions in the dataset?
3. Which location has the highest number of fraudulent transactions?
4. What is the most common transaction type involved in fraud?
5. Which merchant has the highest number of fraudulent transactions?
6. What is the average transaction amount for fraudulent activities?
7. What is the total monetary amount of all fraudulent transactions?

8. How many unique fraudulent transactions are present in the dataset?

Execution Steps to Follow:

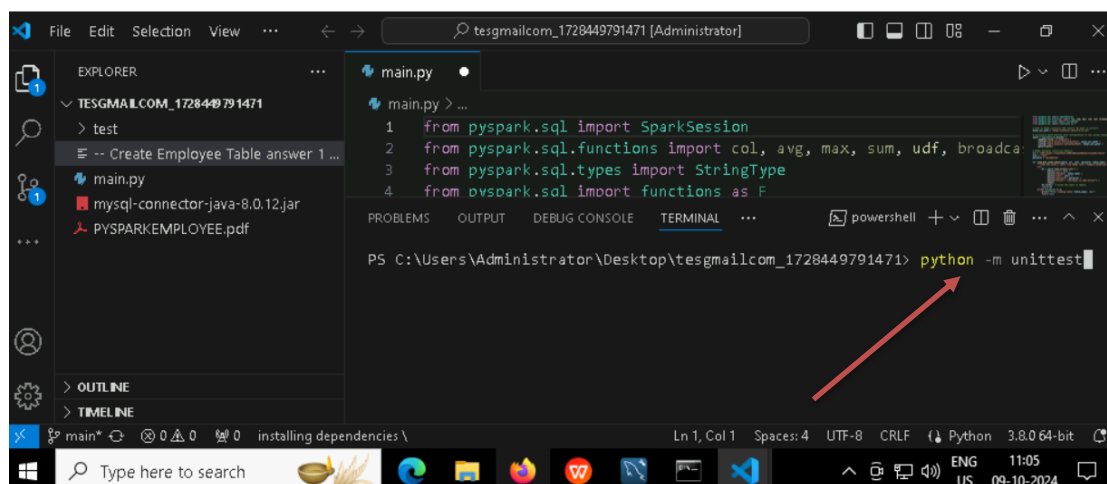
1. All actions like build, compile, running application, running test cases will be through Command Terminal.
2. To open the command terminal the test takers, need to go to Application menu (Three horizontal lines at left top) -> Terminal -> New Terminal
3. This editor Auto Saves the code
4. If you want 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 internal git/repository. Else the code will not be available in the next login.
5. 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.
6. To setup environment:
You can run the application without importing any packages
7. To launch application:
Python banktemplate.py
8. To run Test cases:
python -m unittest
Before Final Submission also, you need to use CTRL+Shift+B-command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository for code

Screen shot to run the program



To run the application

- Python templatebank.py



To run the testcase

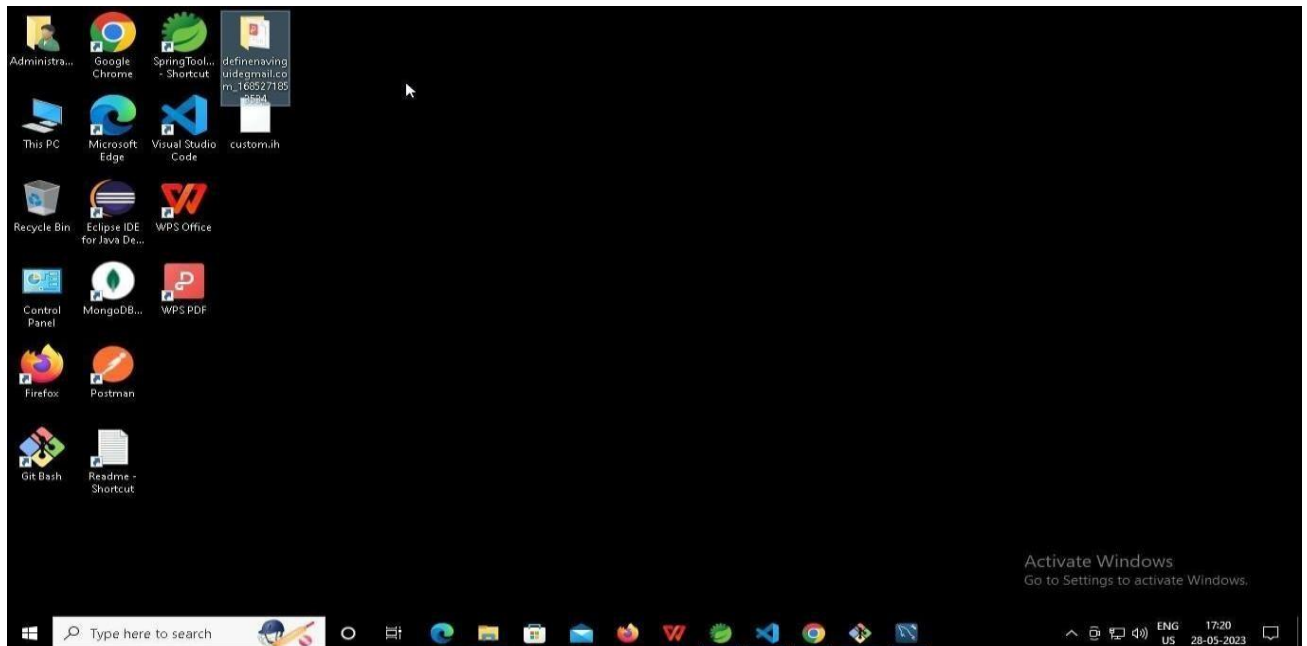
- Python -m unittest

Screenshot to push the application to github

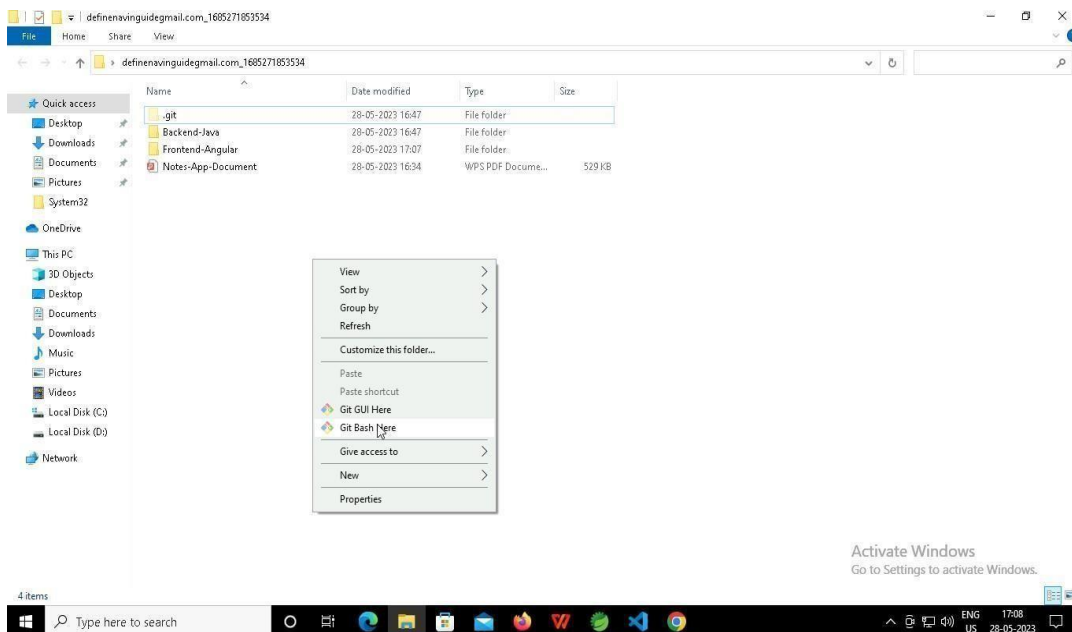
-----X-----

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/tesgmailto...
$ 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/tesgmailto...
$
```

d. `git add .`

```
Administrator@2a5ee7ad258f58c MINGW64 ~/Desktop/tesgmailto...
$ git add .
```

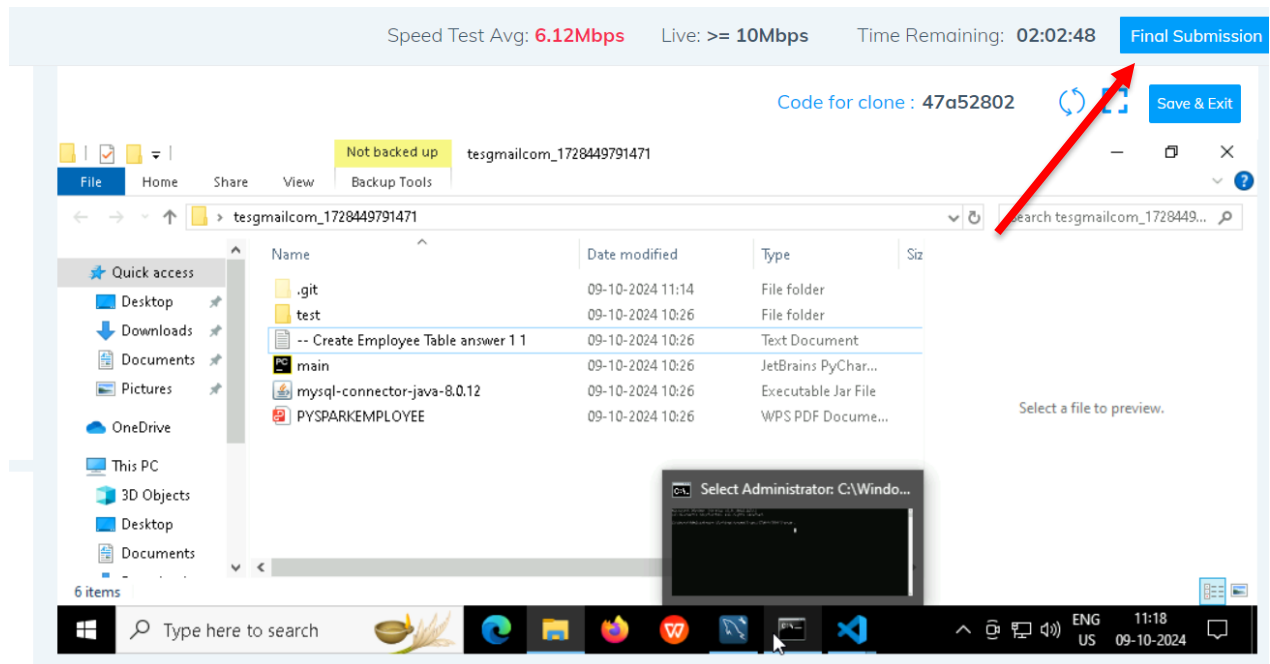
e. `git commit -m "First commit"`
(You can provide any message every time you commit)

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

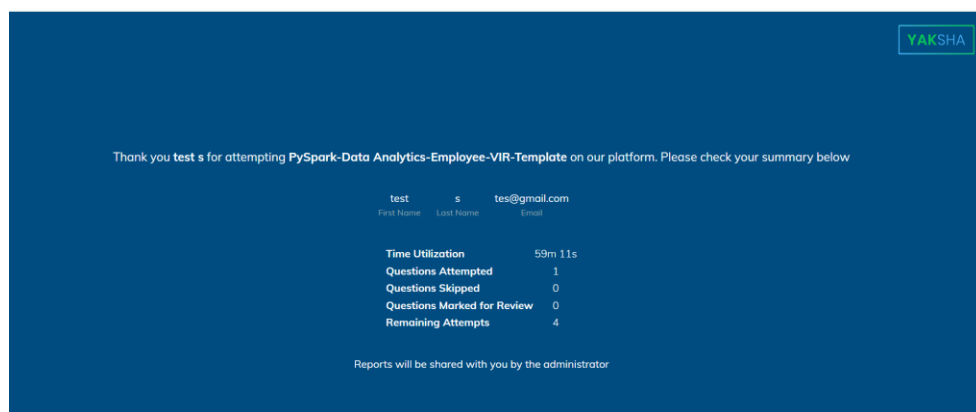
f. `git push`

```
Administrator@2a5ee7ad258f58c MINGW64 ~/Desktop/tesgmailto...
$ 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/tesgmailto...
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-----