

---

# System Requirements Specification Index

For

Python ML

Diabetes Prediction System 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

YAKSHA

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

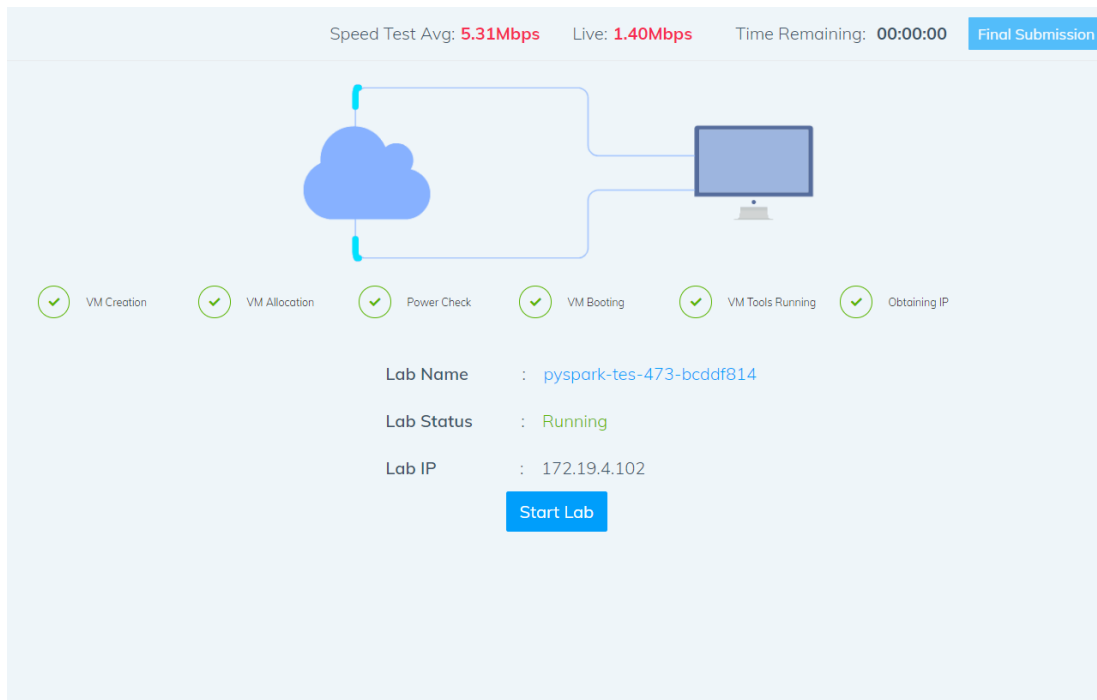
To develop use case all IDEs, Database required are available in VM. There is a README file on desktop containing any credentials you need.

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)

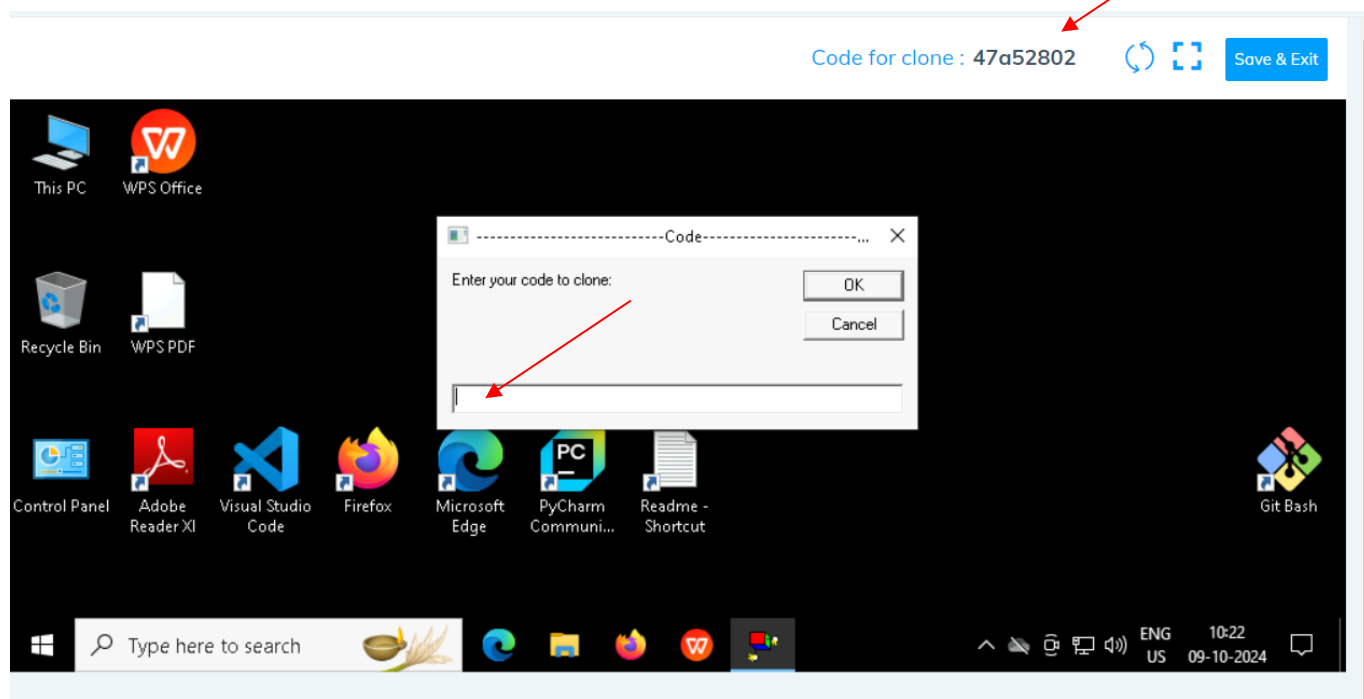
Launch Assessment Environment



Step 3 Click on the start lab button

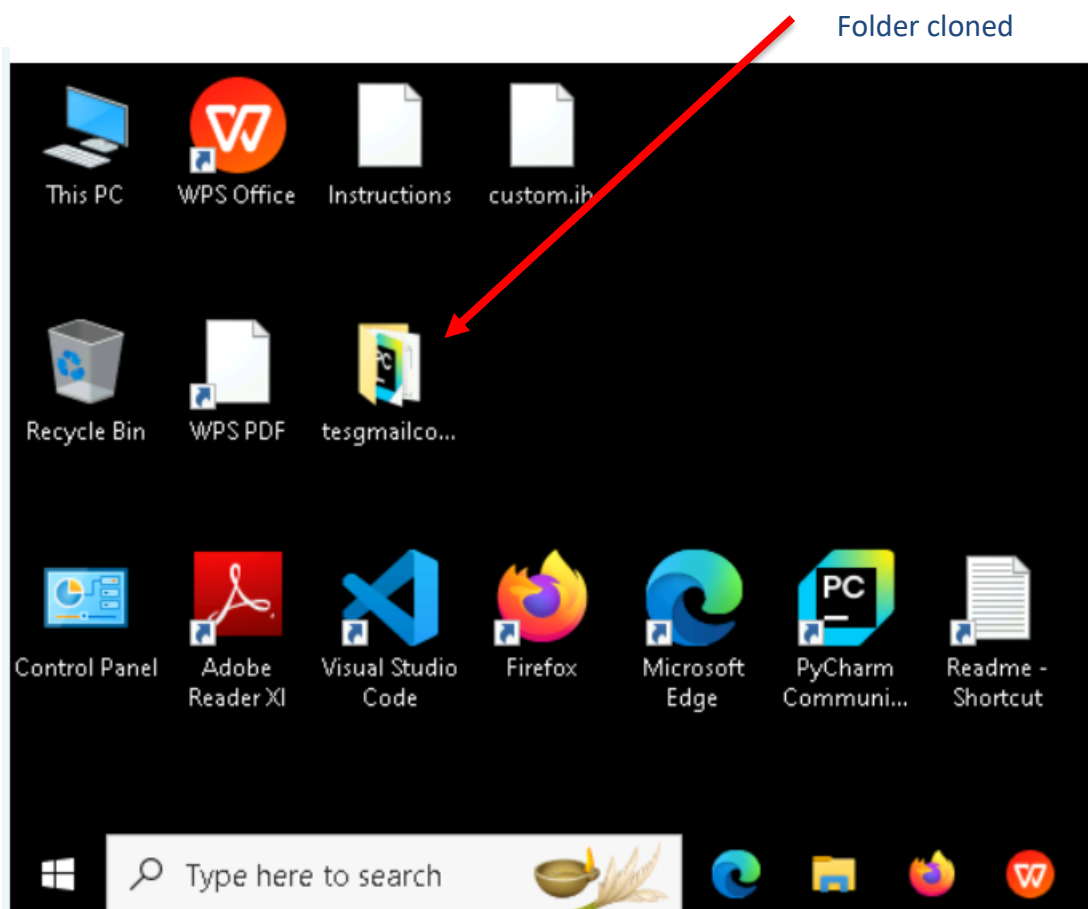
Step 4 you will get a window you need to type the code from that top corner

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

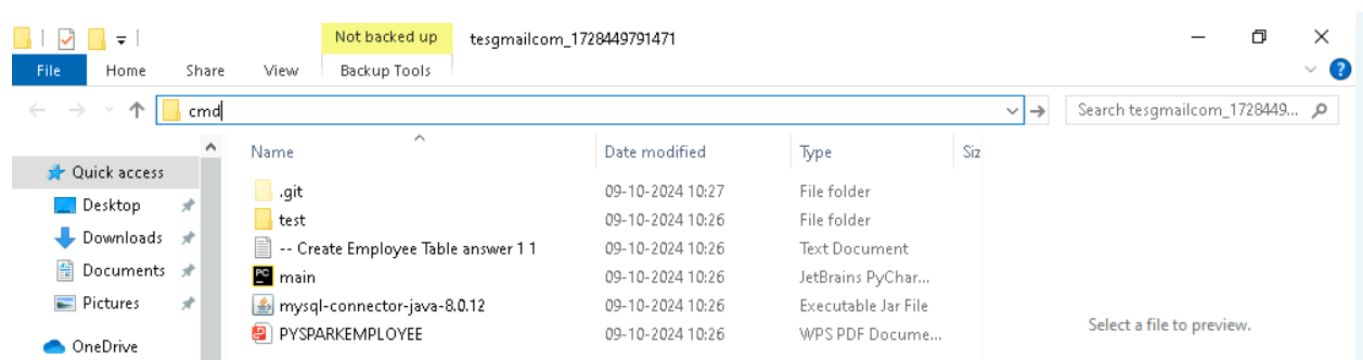


Click on ok

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

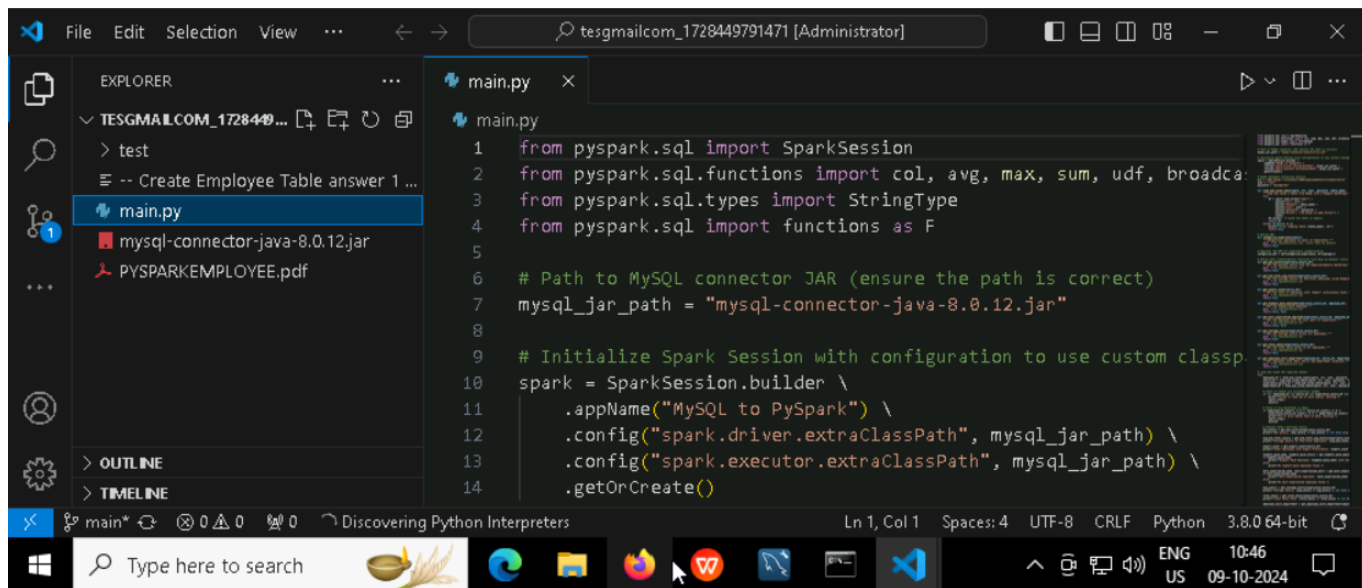


Step 6 go inside the folder type cmd in the top of the file explorer



- 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 : **Diabetes Prediction System**

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

## Objective

To build, train, and test a machine learning model that predicts diabetes status based on individual features. The project also includes functional and boundary testing.

Imagine you are part of a healthcare team in a bustling city where diabetes is becoming a growing public health concern. Early detection is critical for managing diabetes effectively and improving patient outcomes. However, limited resources and overburdened healthcare systems make it challenging to screen every individual.

To address this challenge, the hospital's research division has decided to develop an **AI-driven Diabetes Prediction System**. This system aims to predict whether a person is diabetic based on their lifestyle and medical history, enabling early intervention.

Your team, composed of data scientists and software engineers, is tasked with creating this prediction system. The system will use patient data such as age, BMI, blood glucose levels, and smoking history to predict diabetes risk. The project will also include rigorous testing to ensure reliability and robustness.

---

Healthcare startup has partnered with hospitals to leverage technology for public health. Rising diabetes rates have led to an urgent need for solutions that go beyond manual screening processes.

1. **Historical patient data** for training the model.
2. A **dataset of individuals** for testing the model predictions.

## Project Structure

The project contains the following files:

1. **data\_preprocessing.py**: Handles data preprocessing tasks like missing value imputation, scaling, and encoding.
2. **model\_training.py**: Trains the machine learning model and evaluates its performance.
3. **prediction.py**: Makes predictions for new individuals using the trained model and preprocessor.
4. **main.py**: Combines all the steps: preprocessing, training, and predictions.
5. **test\_functional.py**: Functional tests for key project components.

## Dataset Requirements

1. **Training Dataset (diabetes\_prediction\_dataset.csv)**:
  - Includes columns such as age, bmi, HbA1c\_level, blood\_glucose\_level, gender, smoking\_history, and diabetes.
2. **Prediction Dataset (persons\_for\_prediction.csv)**:
  - Includes individual details with columns such as name, age, bmi, HbA1c\_level, blood\_glucose\_level, gender, and smoking\_history.

## Steps to Execute

### 1. Data Preprocessing

- The data\_preprocessing.py script:
  - Handles missing data.
  - Scales numerical columns.
  - Encodes categorical columns.
  - Splits data into training and testing sets.

### 2. Model Training

- The model\_training.py script:
  - Trains a Random Forest classifier.
  - Saves the trained model as diabetes\_model.pkl.

### 3. Prediction

- The prediction.py script:
  - Uses the saved model and preprocessor to make predictions.
  - Outputs predictions for individuals in the persons\_for\_prediction.csv file.

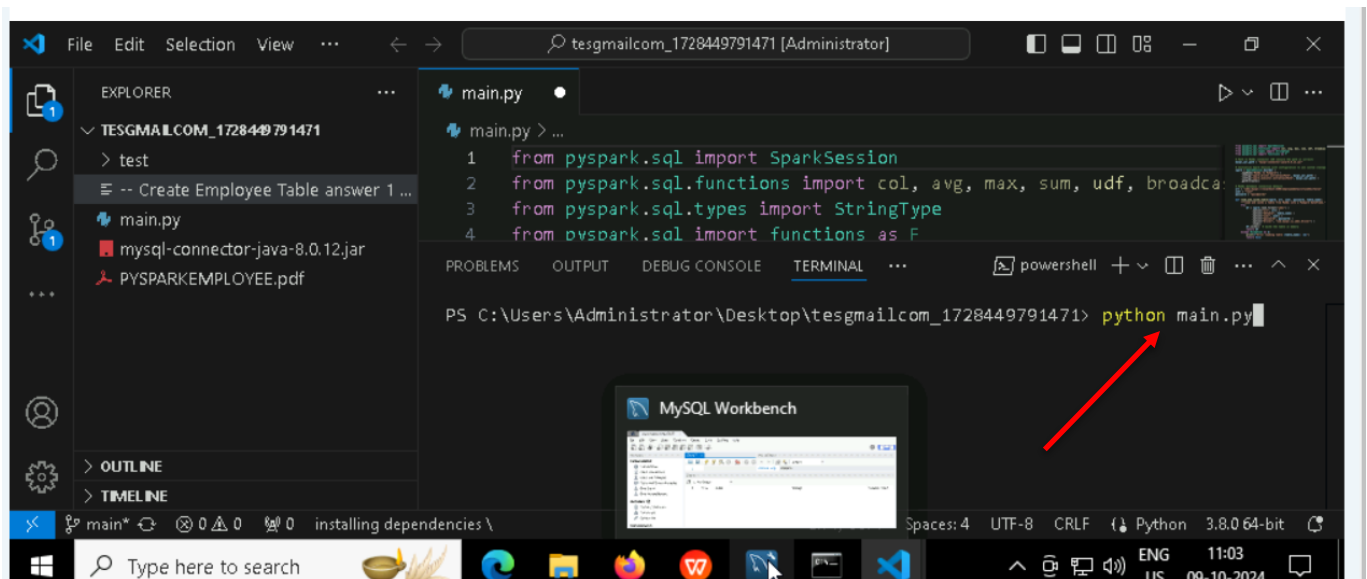
## Solve these Questions

1. Create a predictive model using the RandomForestClassifier
2. Ensure the model achieves an accuracy of 0.97.
3. Classify predictions for ten patients provided in the persons\_for\_prediction.csv file.
4. Include precision, recall, f1-score, and support as part of the model evaluation metrics.

## **Execution Steps to Follow:**

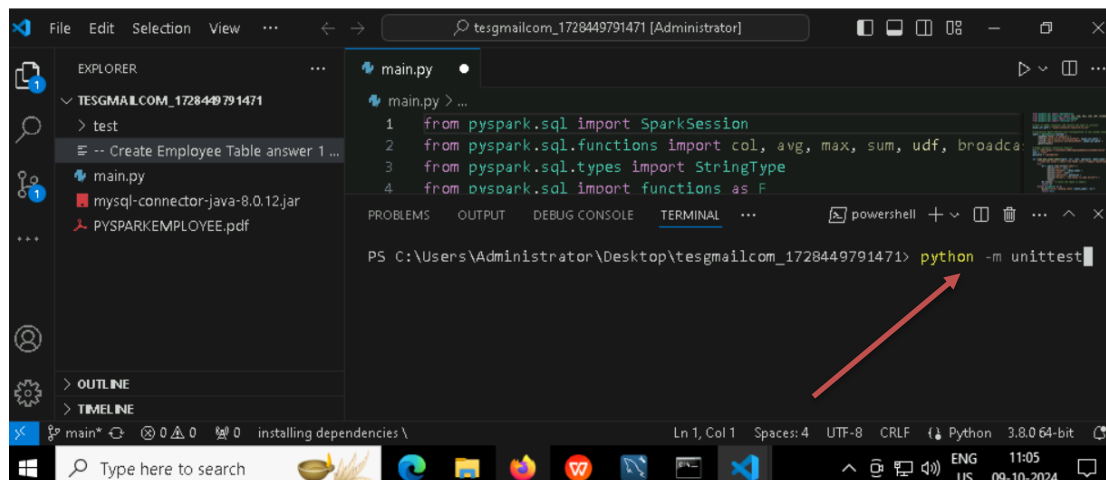
1. All actions like build, compile, running application, running test cases will be through Command Terminal.
2. To open the command terminal the test takers, need to go to Application menu (Three horizontal lines at left top) -> Terminal -> New Terminal
3. This editor Auto Saves the code
4. If you want 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  
main.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 main.py**



## To run the testcase

- **Python -m unittest**

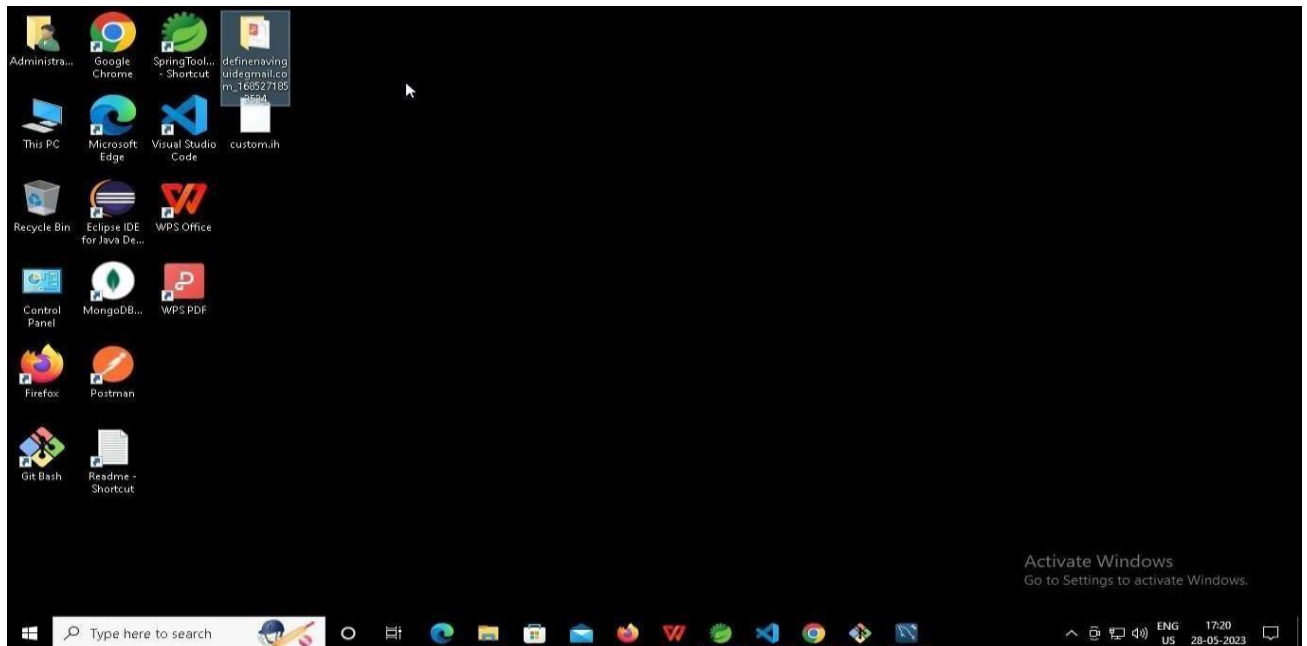
## Screenshot to push the application to github

-----X-----

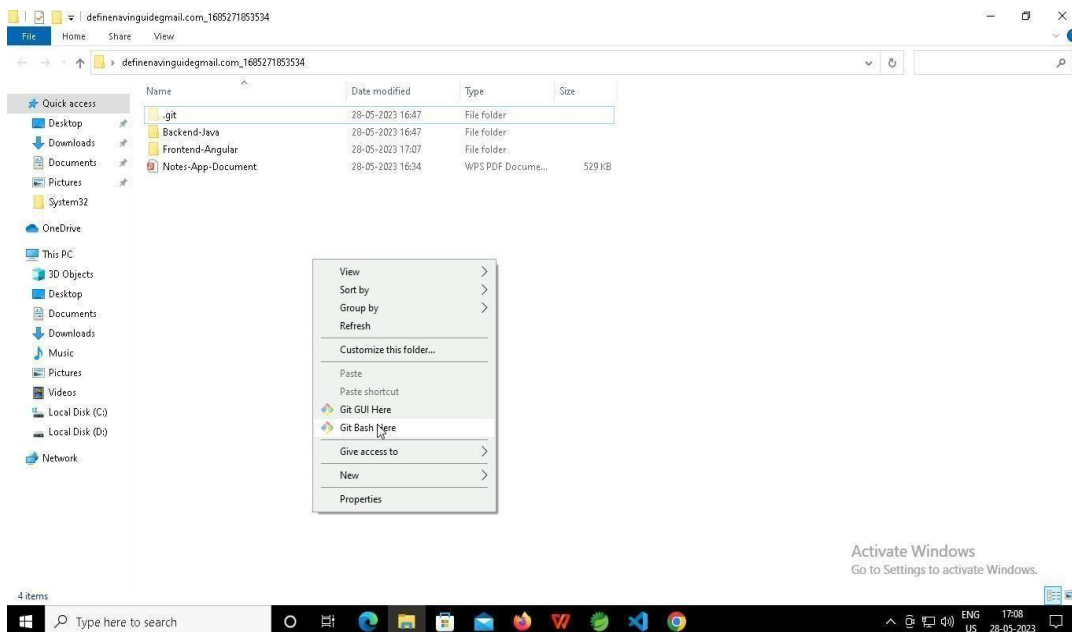
You can run test cases as many numbers of times and at any stage of Development, to check howmany test cases are passed/failed and accordingly refactor your code.

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/tesgmailto_1728449791471
$ 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_1728449791471 (main)
$
```

d. git add .

```
Administrator@2a5ee7ad258f58c MINGW64 ~/Desktop/tesgmailto_1728449791471 (main)
$ git add .
```

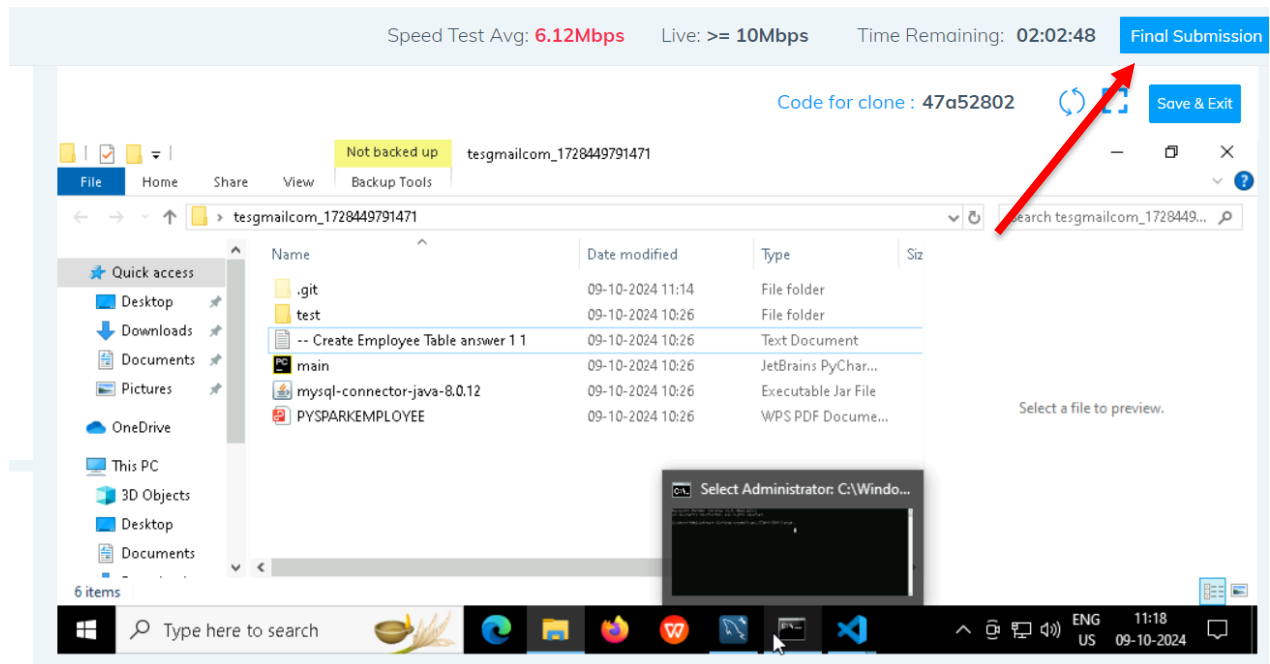
e. git commit -m "First commit"  
(You can provide any message every time you commit)

```
Administrator@2a5ee7ad258f58c MINGW64 ~/Desktop/tesgmailto_1728449791471 (main)
$ git commit -m "first commit"
[main f97ce24] first commit
1 file changed, 91 deletions(-)
delete mode 100644 templateespark.py
```

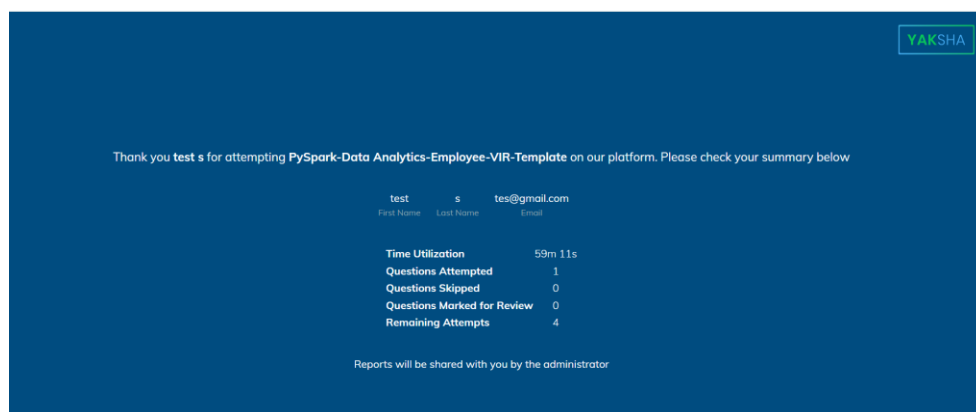
f. git push

```
Administrator@2a5ee7ad258f58c MINGW64 ~/Desktop/tesgmailto_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/tesgmailto_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-----