Systems & Toolchains for AI Engineers: Windows Installation Guide

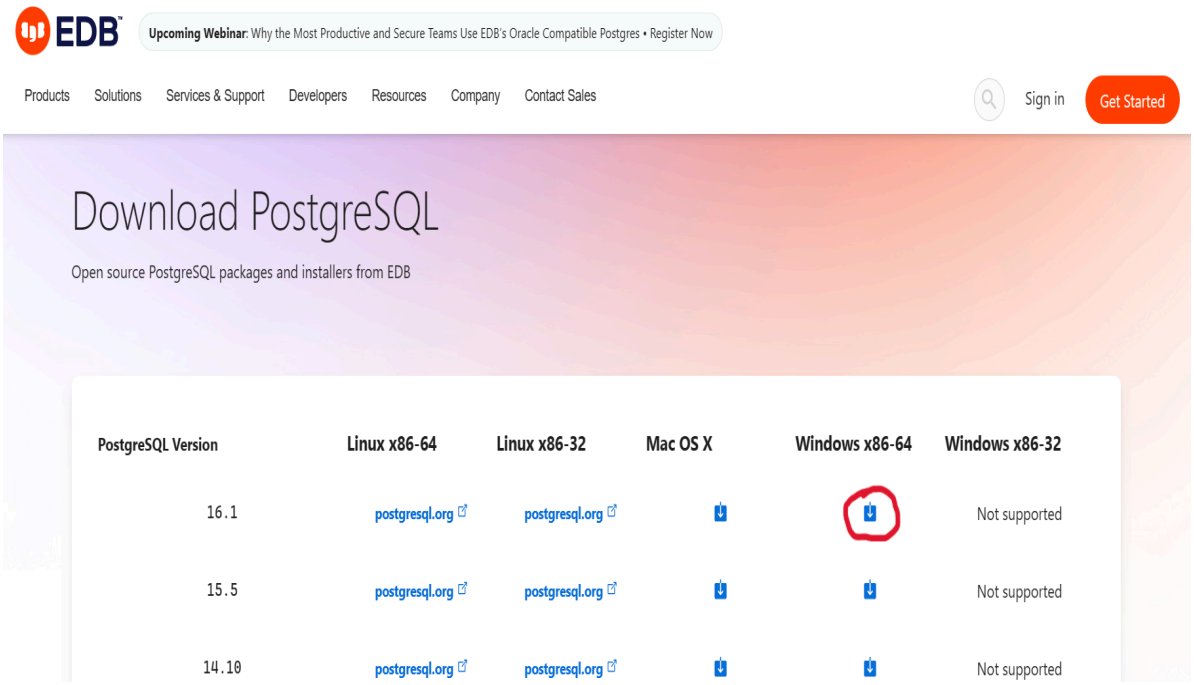
Disclaimer: this guide aims to help you get a high-level idea of required software installations. The provided steps in this guide may get outdated over time and/or may not match your exact operating system file hierarchy. Please use this document as a guide and not a strict document to follow.

PostgreSQL, PgAdmin

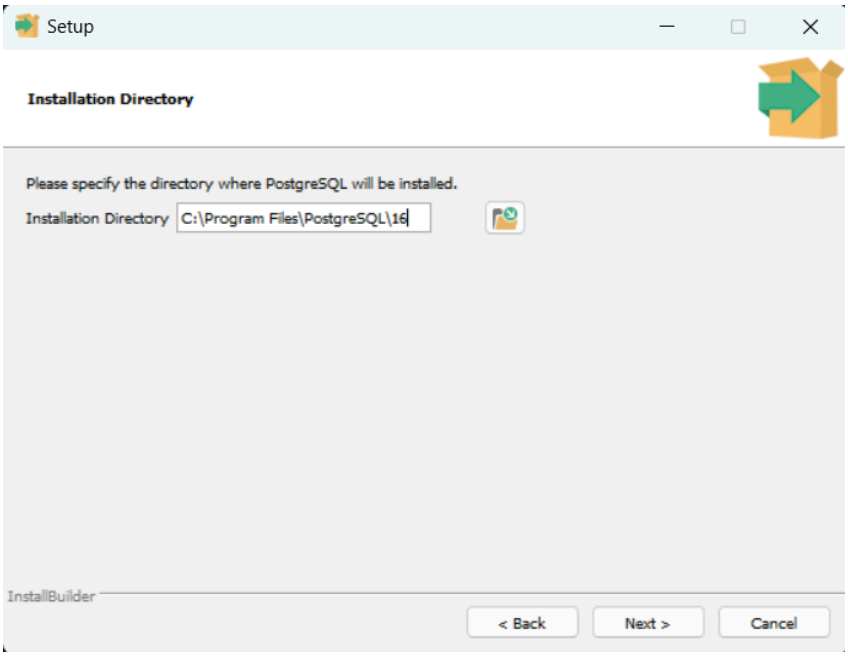
PostgreSQL is an open-source relational database management system, developed from the Ingres project and merged with SQL in 1996.

The PostgreSQL installation is as follows.

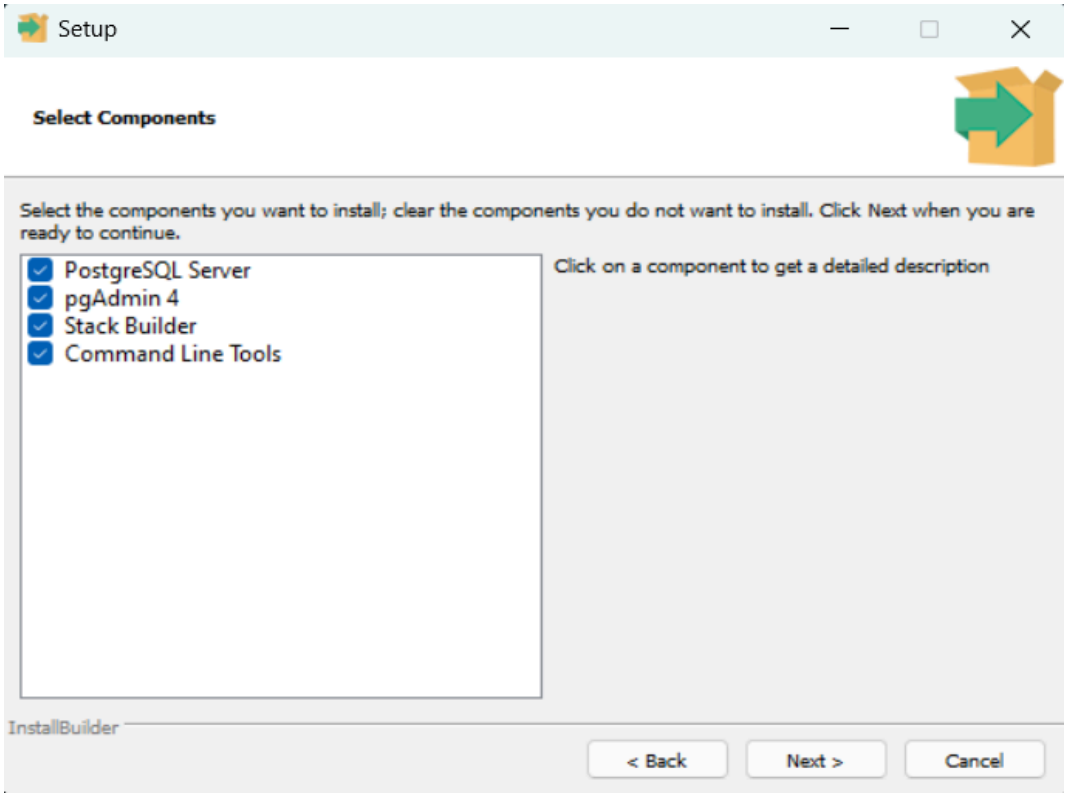
1. Download the latest installation package from Community DL Page (enterprisedb.com)



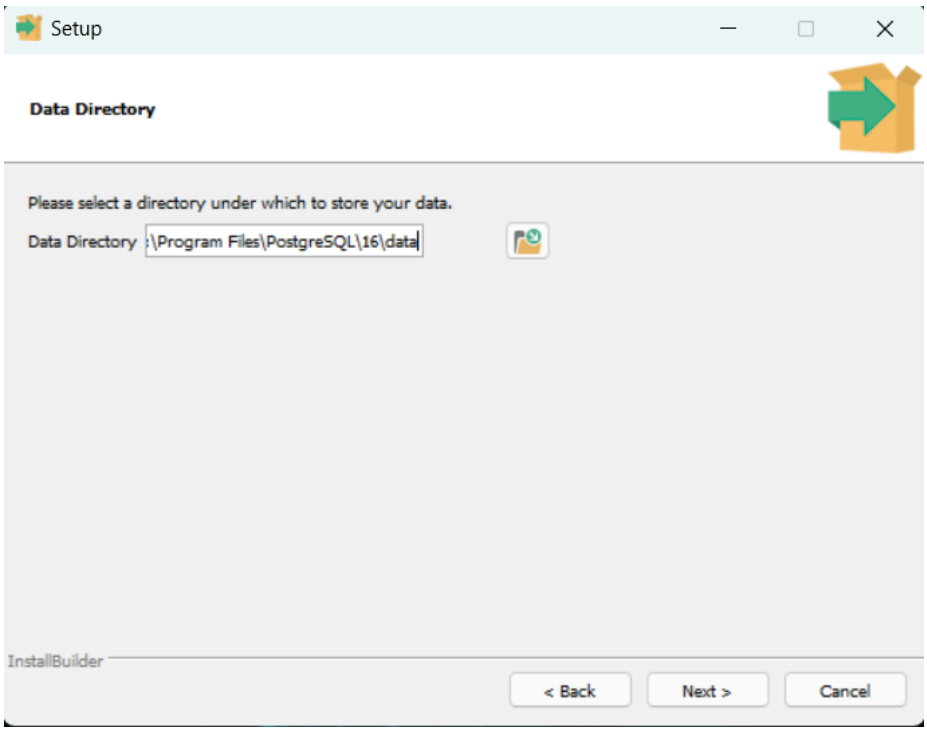
2. Open the setup wizard and follow the instructions.
    a. Leave the directory in the default location.



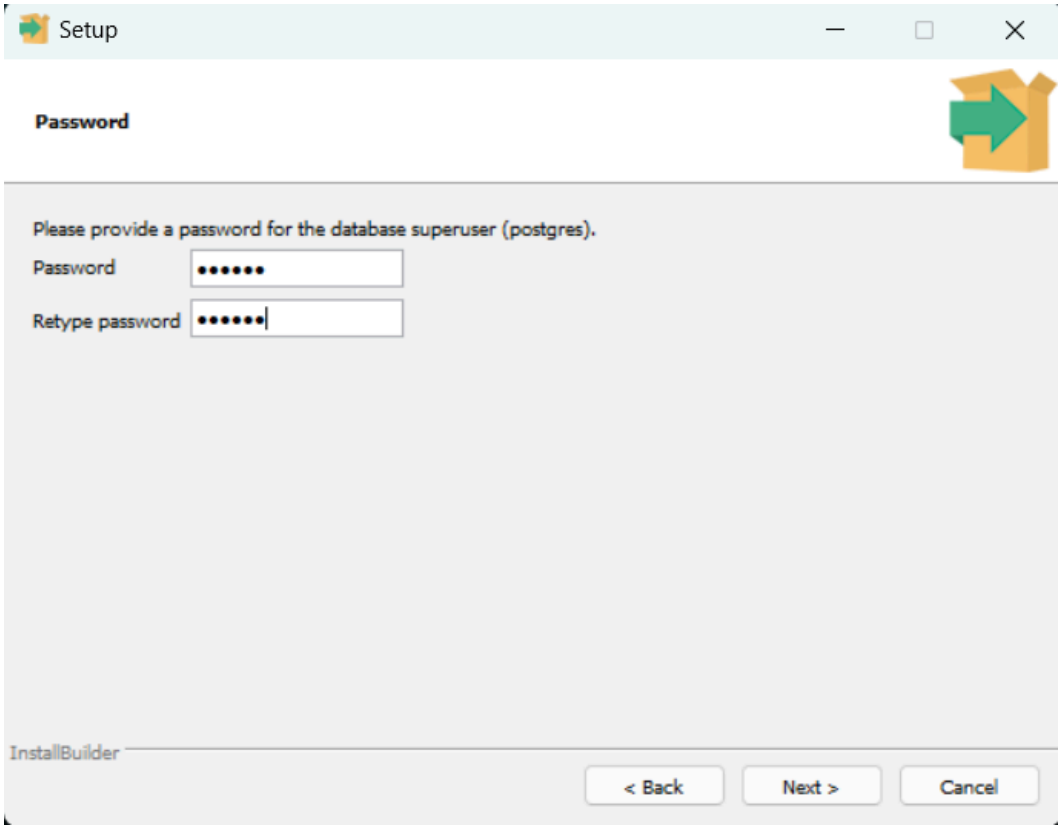
    b. Select the required components for installation. Leave everything checked, but ensure that postgreSQL server and pgAdmin 4 are selected!



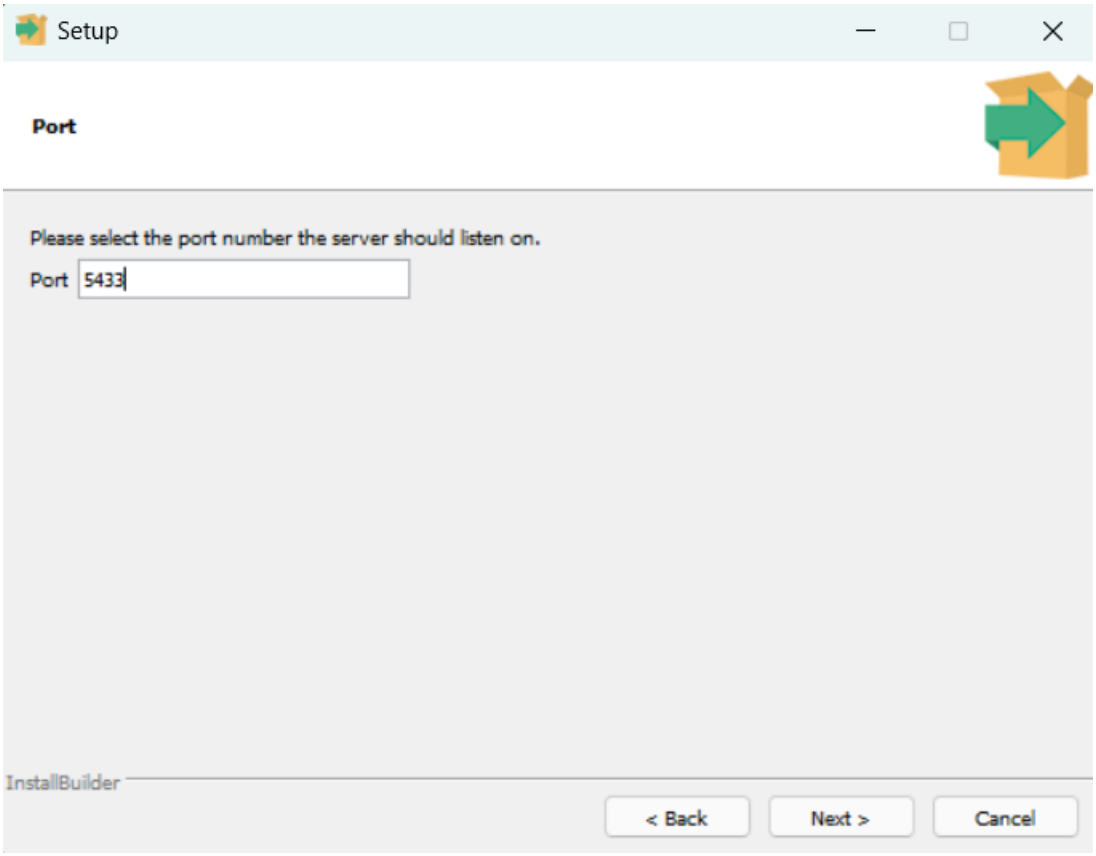
c. Leave the data in the default directory



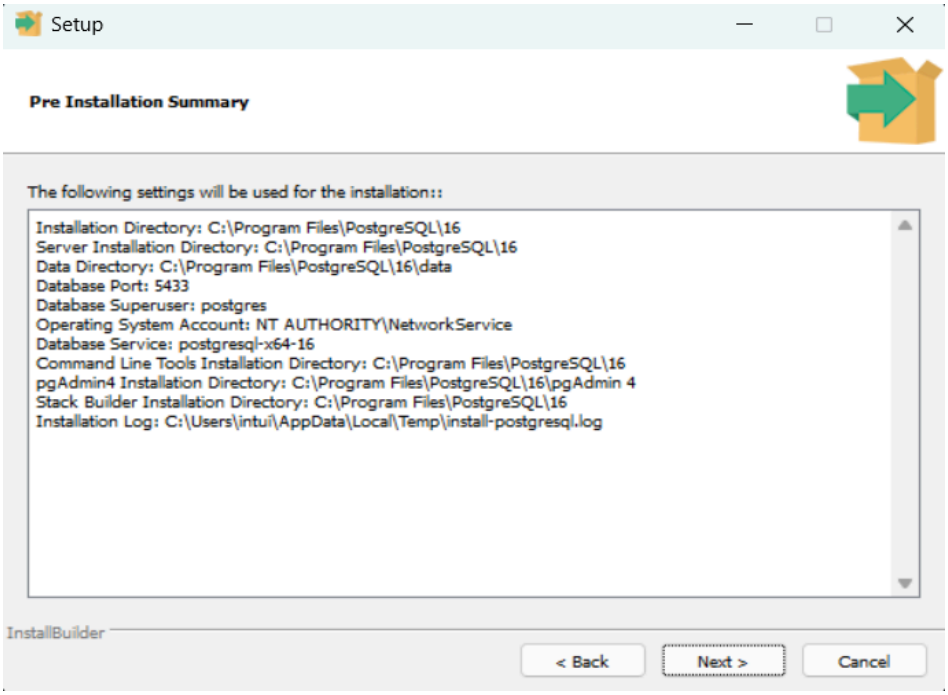
d. Set your superuser password



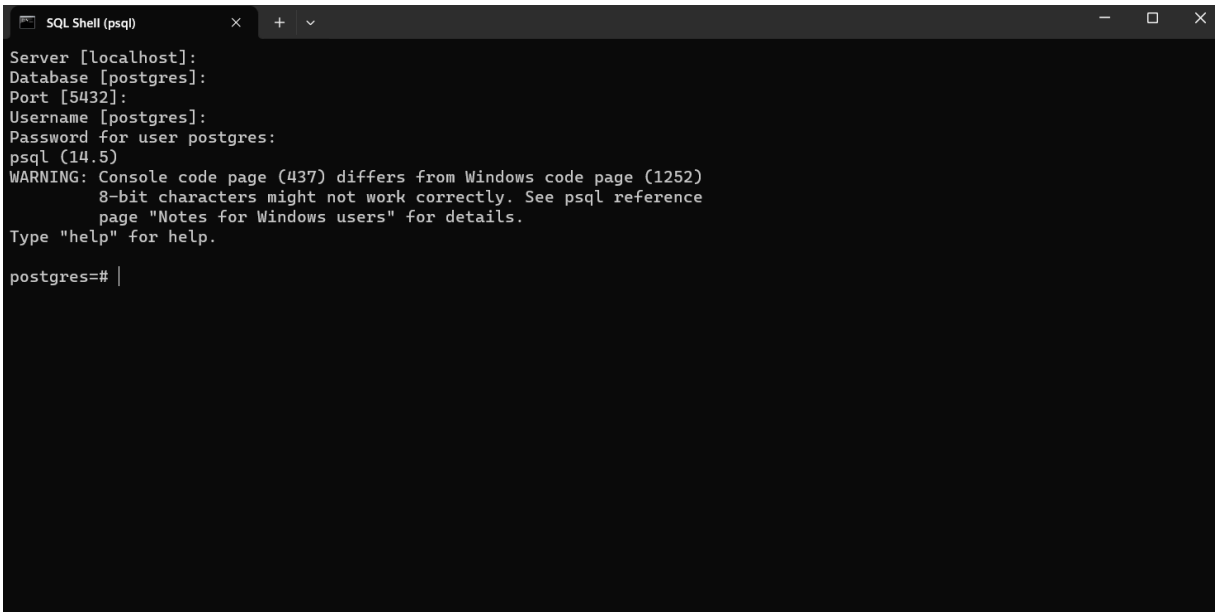
e. Set the port to the default one suggested by the installation wizard.



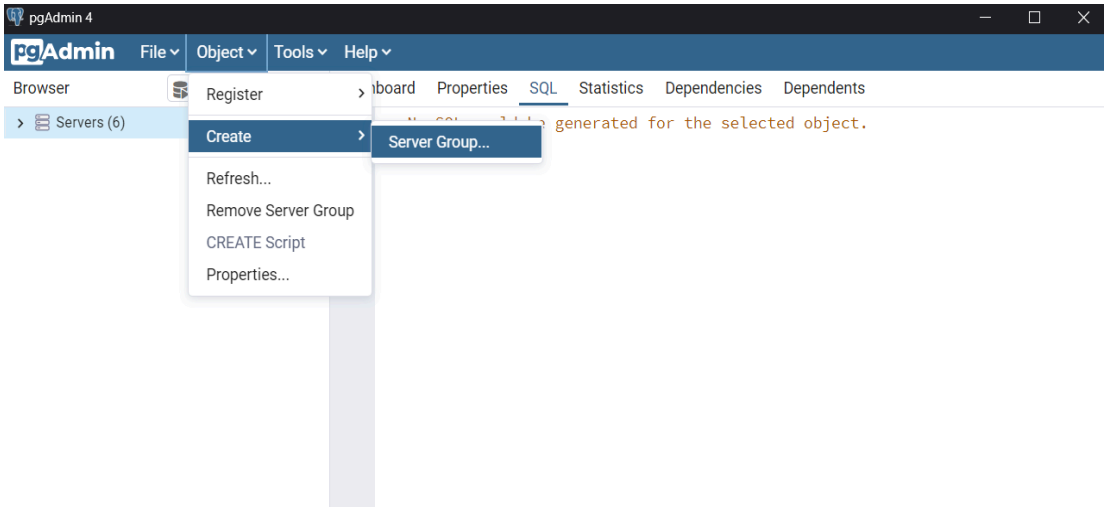
f. After verifying the pre installation summary, install the software



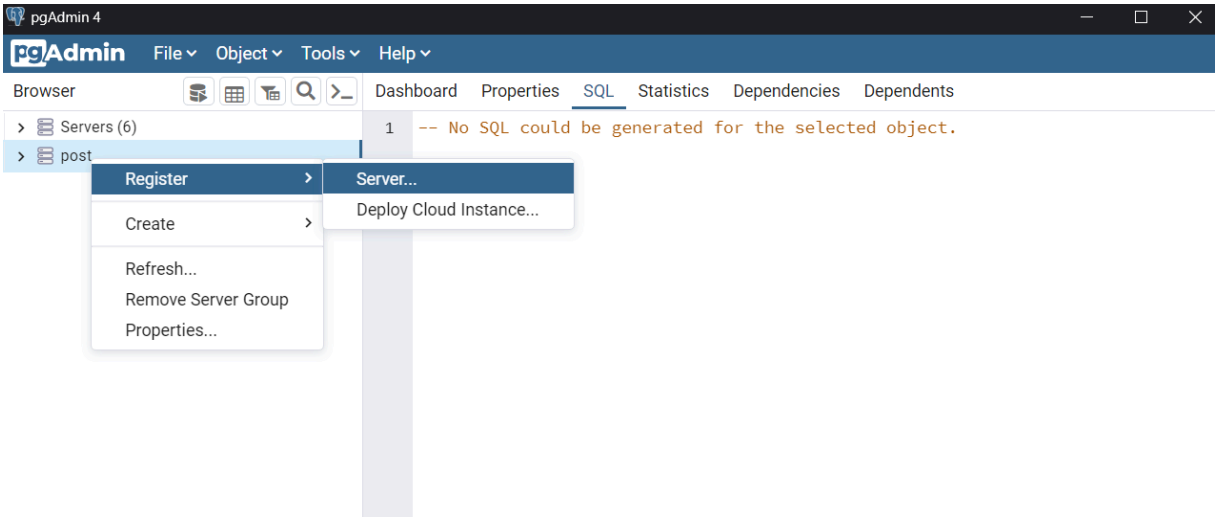
3. Open a psql shell and key in your superuser password to connect

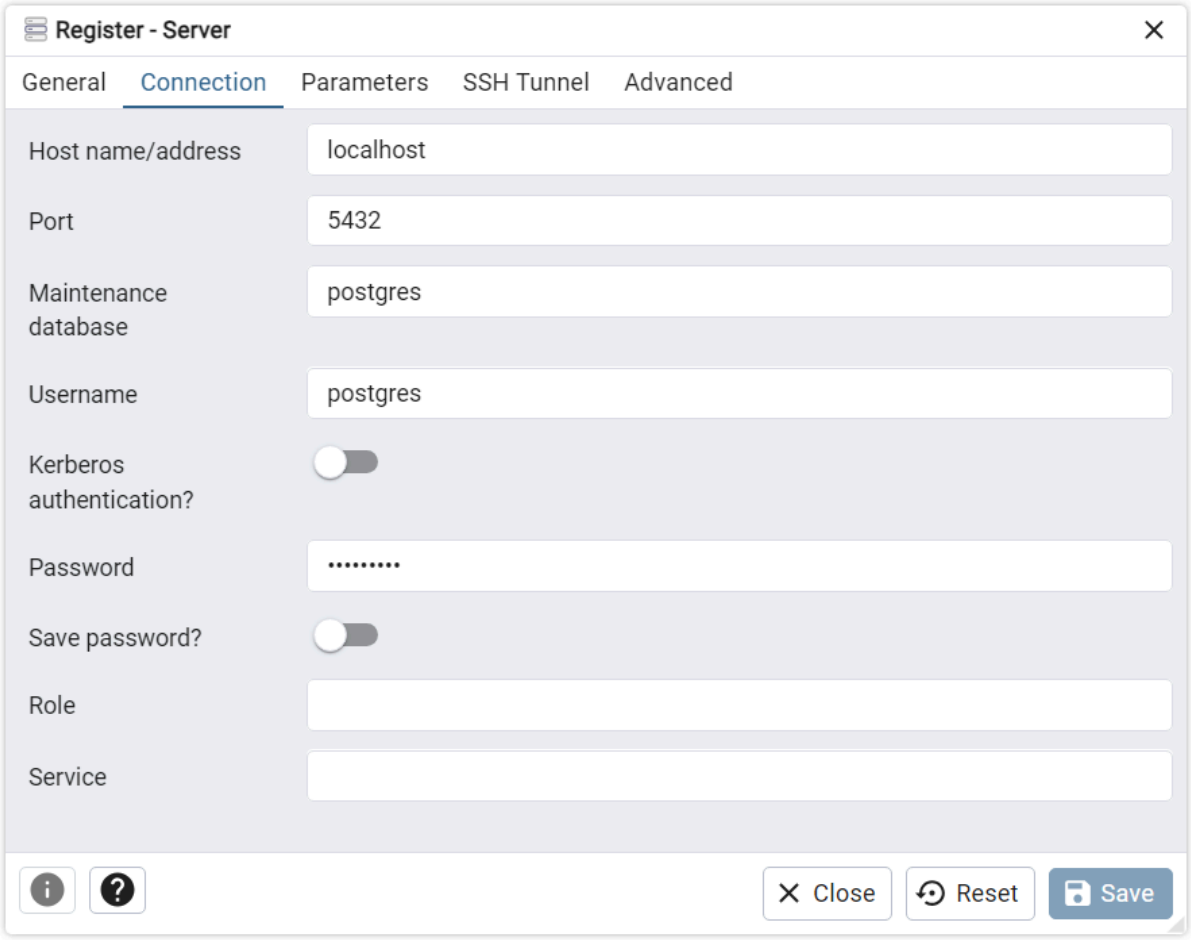


4. If connecting through pgAdmin, there are a few steps to follow.
    a. Create a new server group

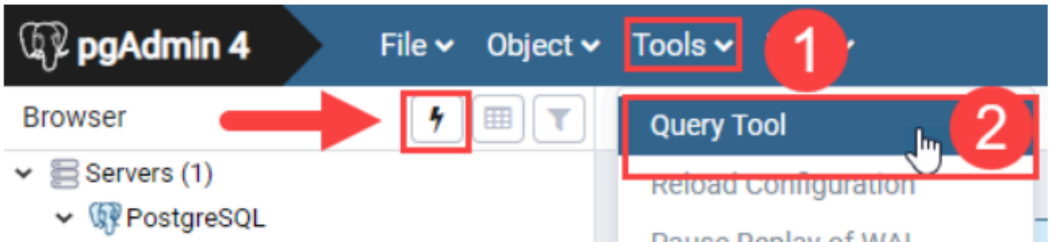


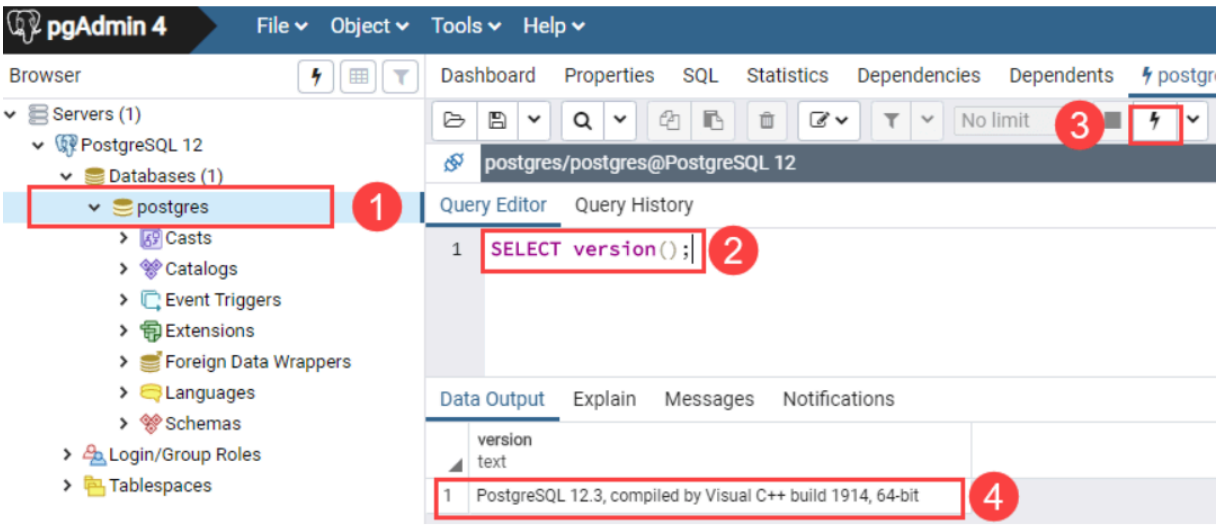
    b. Create a new server





c. Open and execute query tool to conduct queries



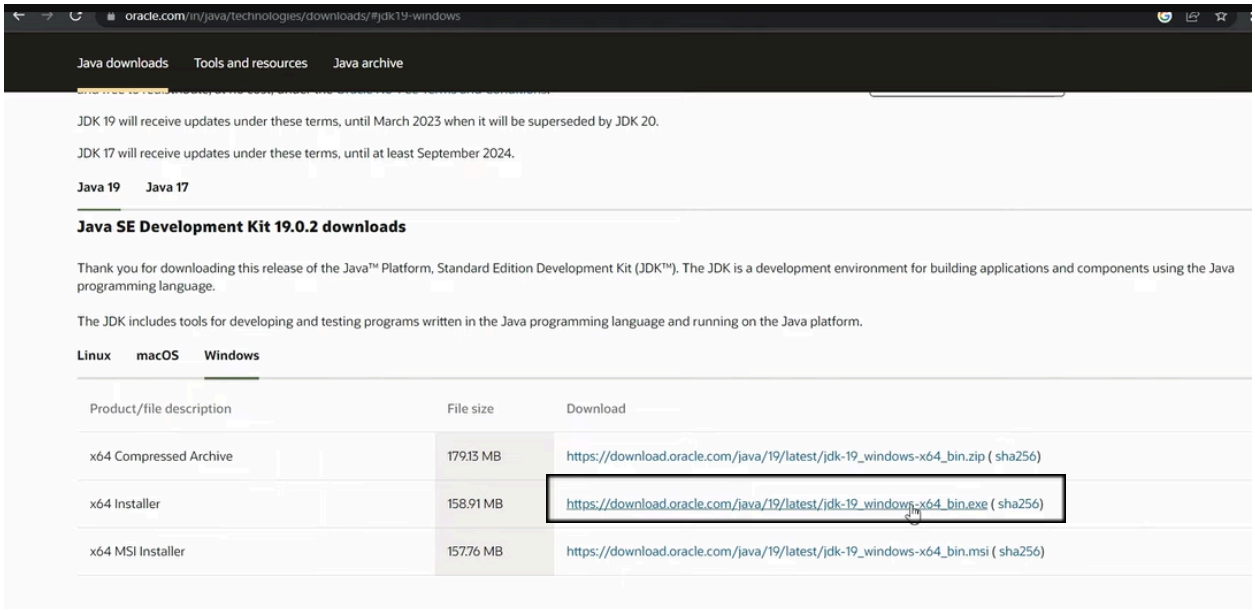

pgAdmin 4
File
Object
Tools
Help
Browser
Servers (1)
PostgreSQL 12
Databases (1)
postgres
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Schemas
Login/Group Roles
Tablespaces
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
No limit
postgres/postgres@PostgreSQL 12
Query Editor
Query History
SELECT version();
Data Output
Explain
Messages
Notifications
version
text
PostgreSQL 12.3, compiled by Visual C++ build 1914, 64-bit
1
2
3
4

Apache Spark

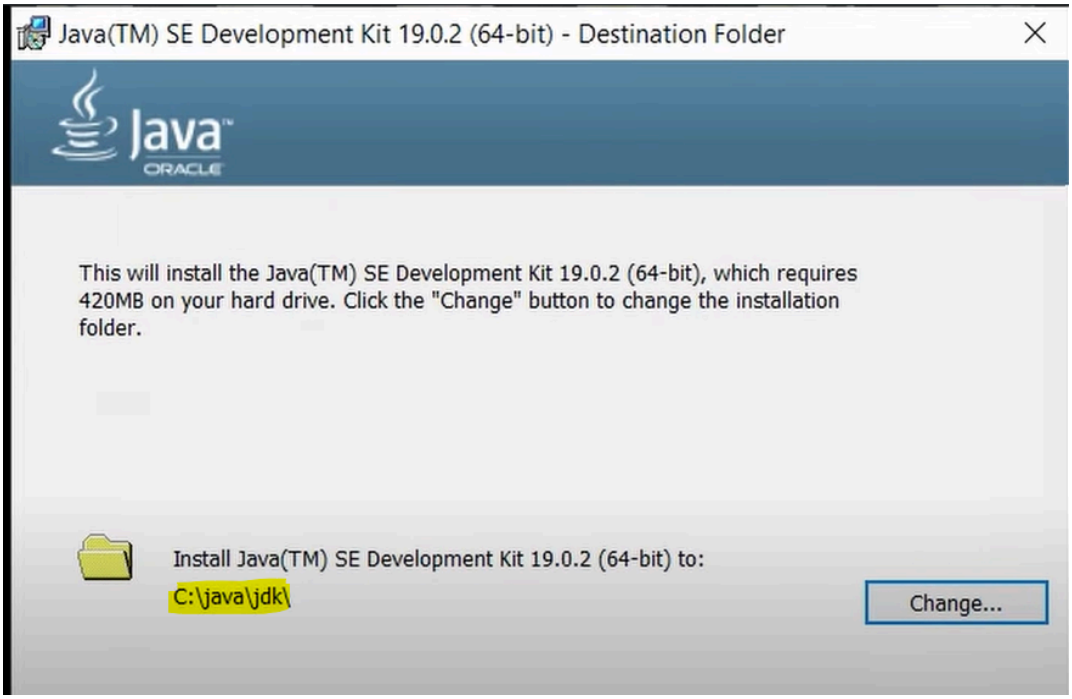
Apache Spark is an open-source unified analytics engine for large-scale data processing developed by AMPLab at UC Berkeley.

Install Java:

1. Download the JDK 11 or 17 from:Oracle Website and choose the installer file, where the arrow in the screenshot can be seen.

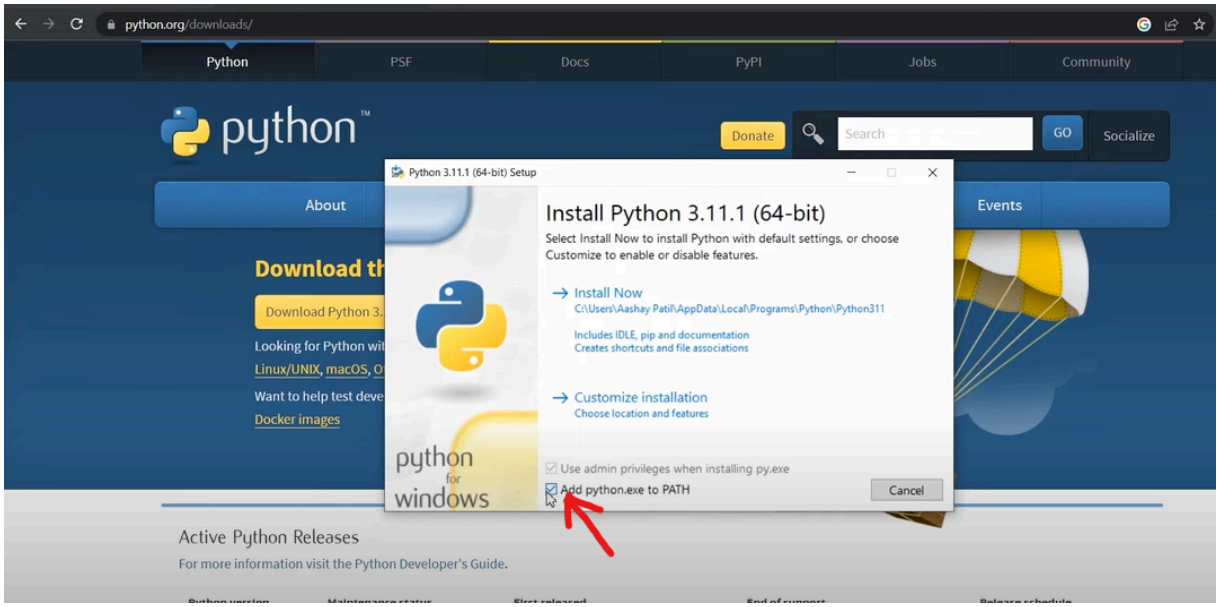


2. After downloading the jdk file, it is recommended to create a new directory for storing the java file.



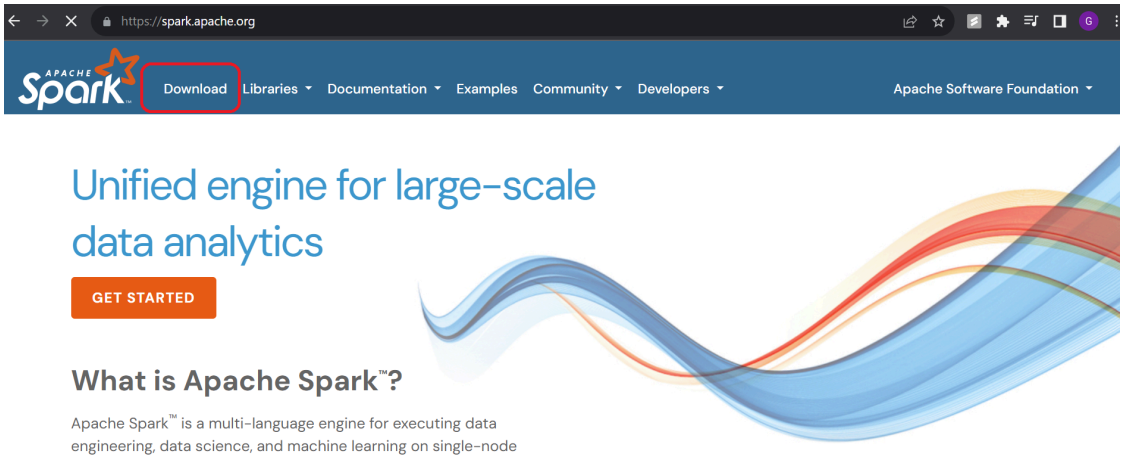
Install Python:

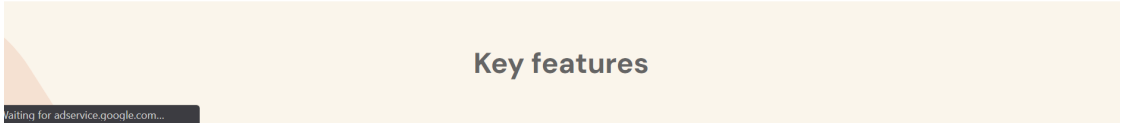
Install the latest version of Python and check the box as shown below. If the box is checked then there is no need to create a separate environment for python.

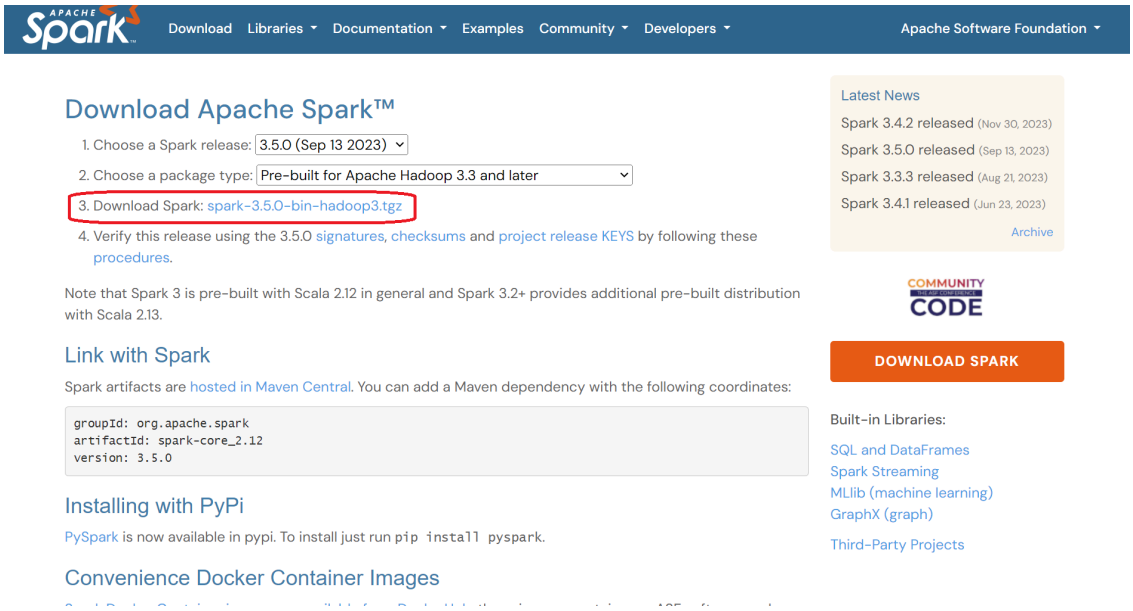


Install Apache Spark:

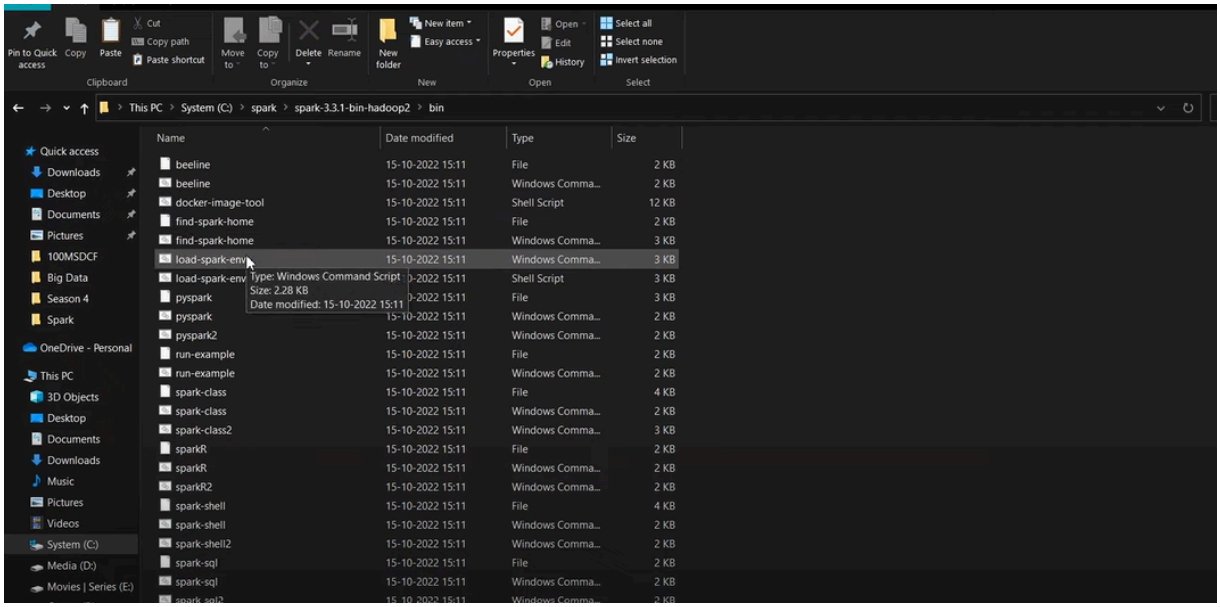
1. Download files from https://spark.apache.org/







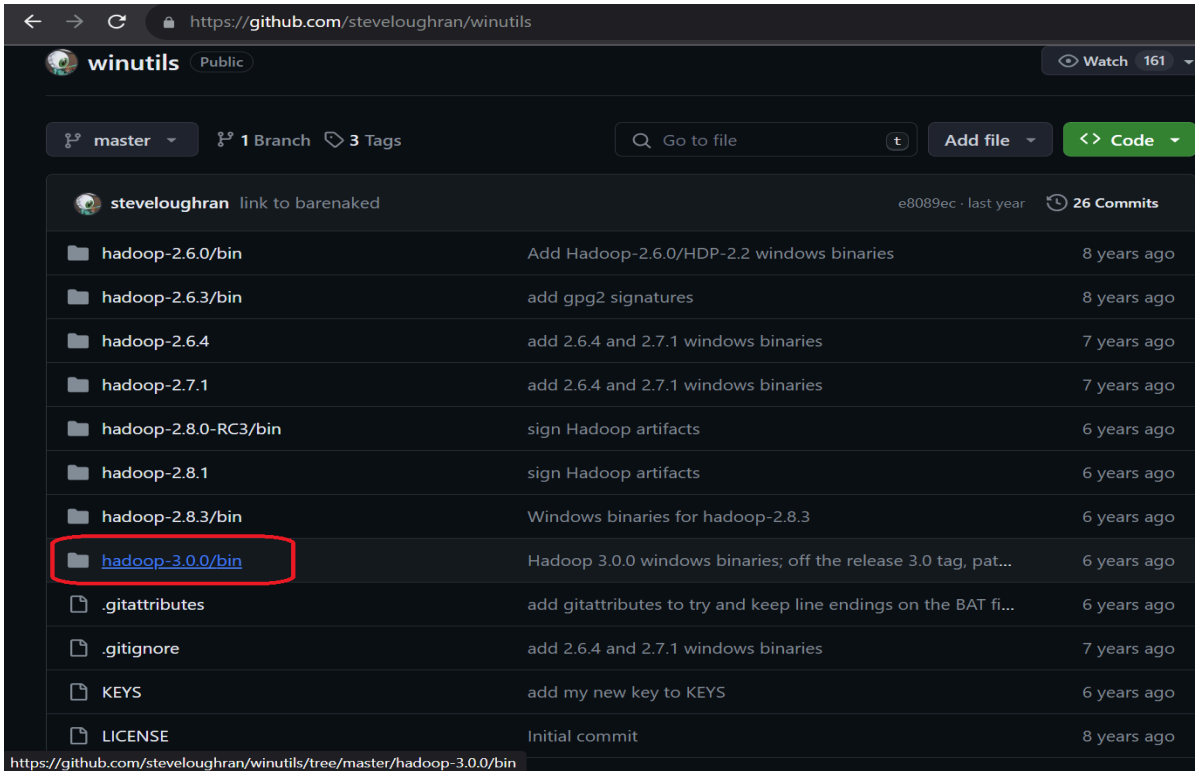
2. Once the download is complete, navigate to the downloads folder, create a new folder in **C drive** named “spark” and paste the file.
3. Now extract the .tar file at the above location in the C drive.
4. All the required files will be present in the bin directory as shown below:



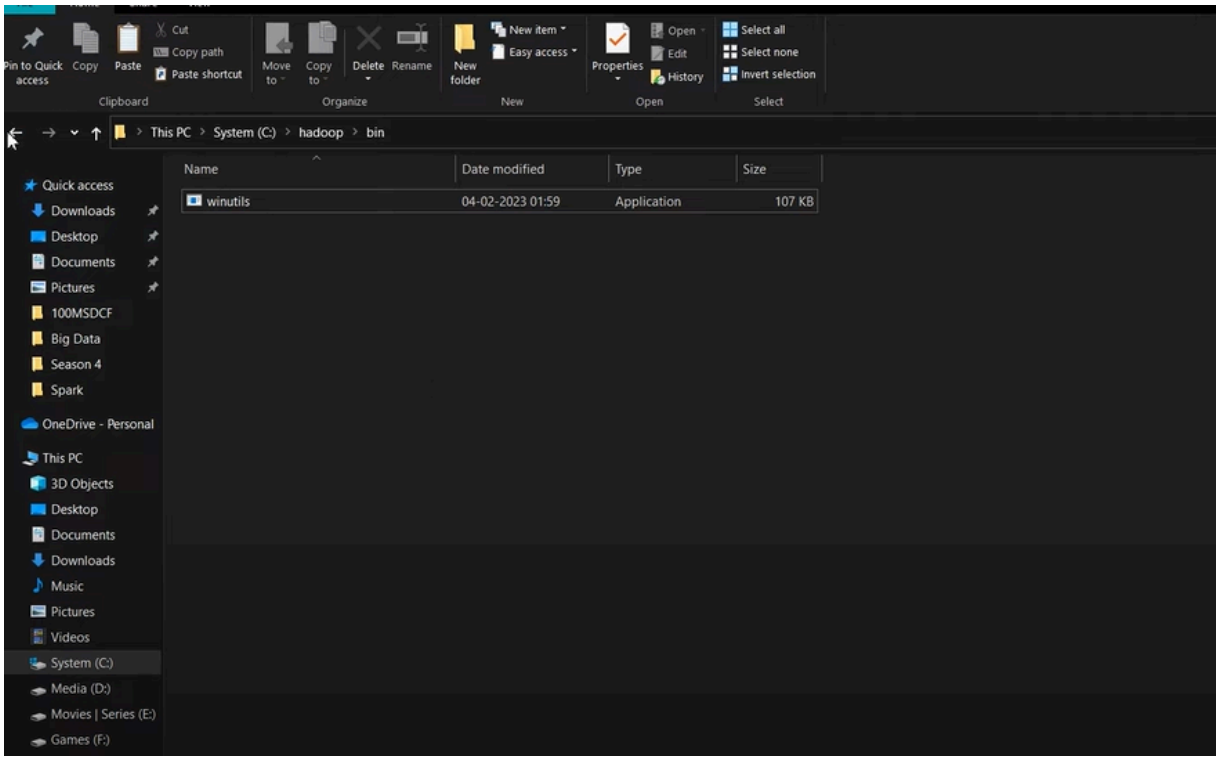
5. But before starting the Spark, winutils needs to be installed. The winutils file should be of the same version as the Spark version.

Install winutils:

1. The files with specific versions are present at the following location:https://github.com/kontext-tech/winutils



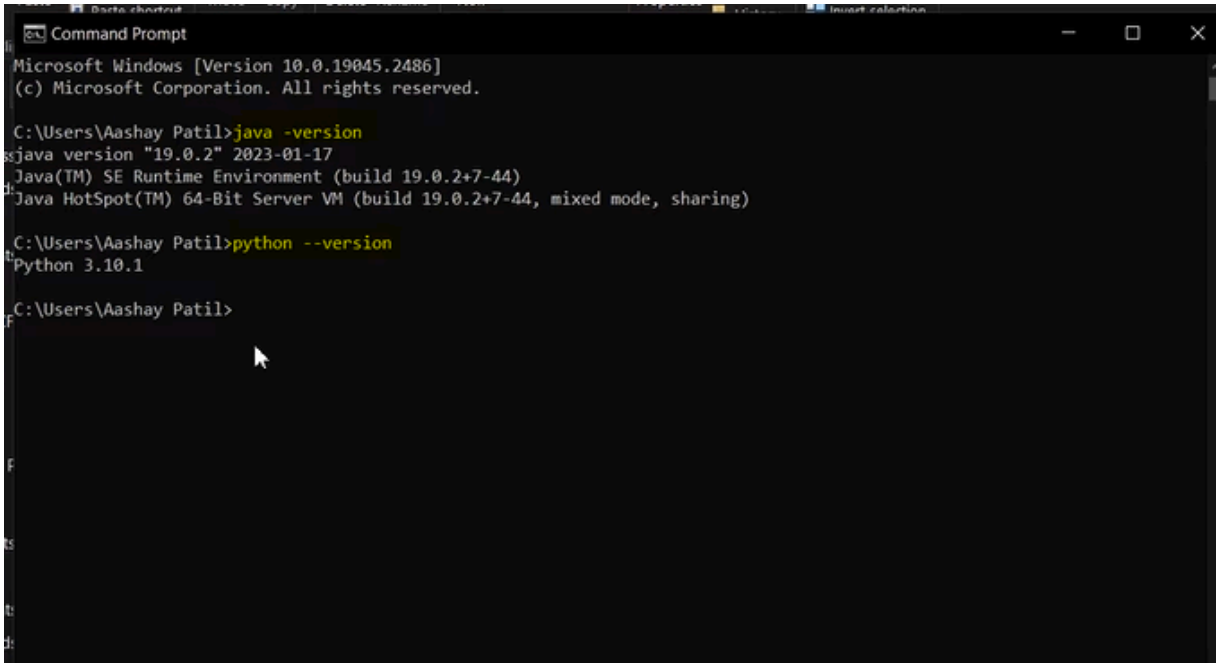
2. Create a new folder “hadoop” in the **C drive.** In the hadoop folder create another folder named “bin” and place the winutils file at this location.



Now you should be having three folders

- java,
- hadoop and
- spark

Check the java and python versions through command prompt as highlighted in the below screenshot:

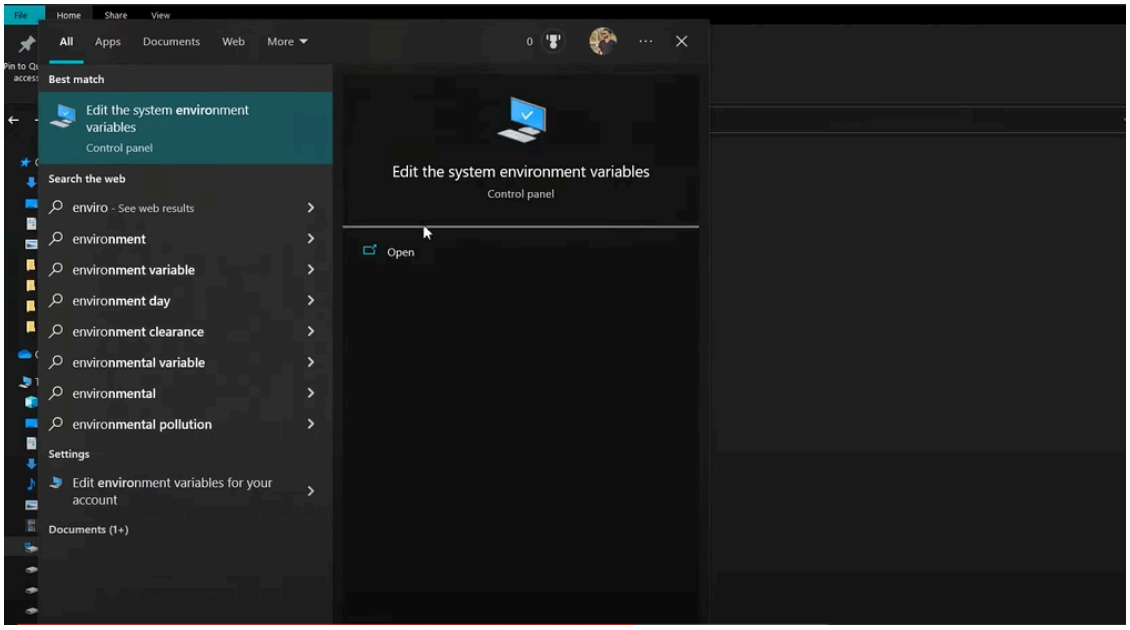


But now for spark and hadoop, the path needs to be set

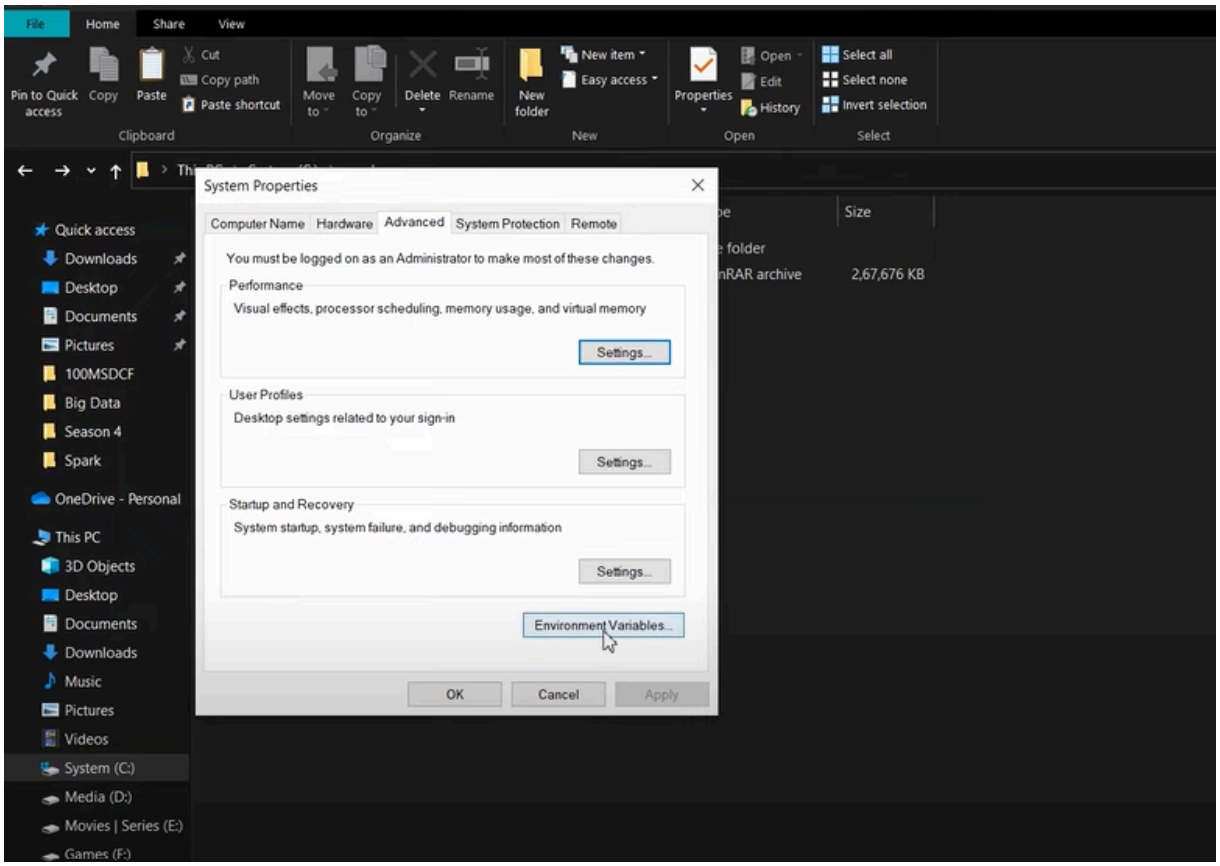
Setting Environment Variables*:

* Please note that the location of the environment variables and path to the bin folder should be very accurate, if not then you might face errors later on.

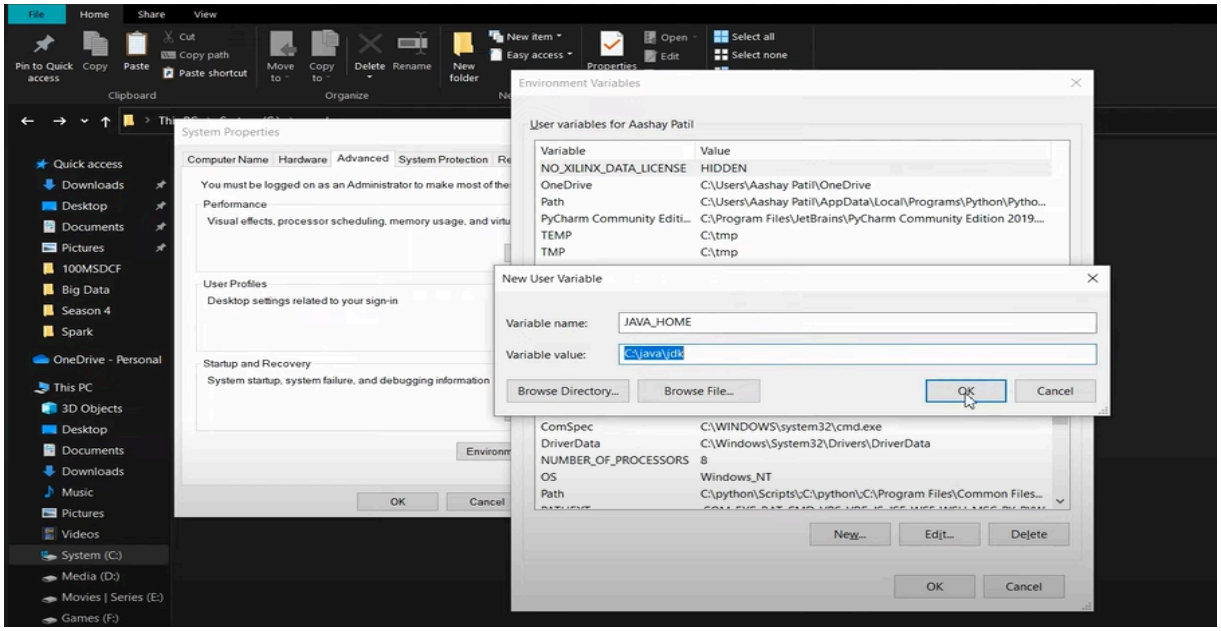
1. Search for Environment variables in windows.



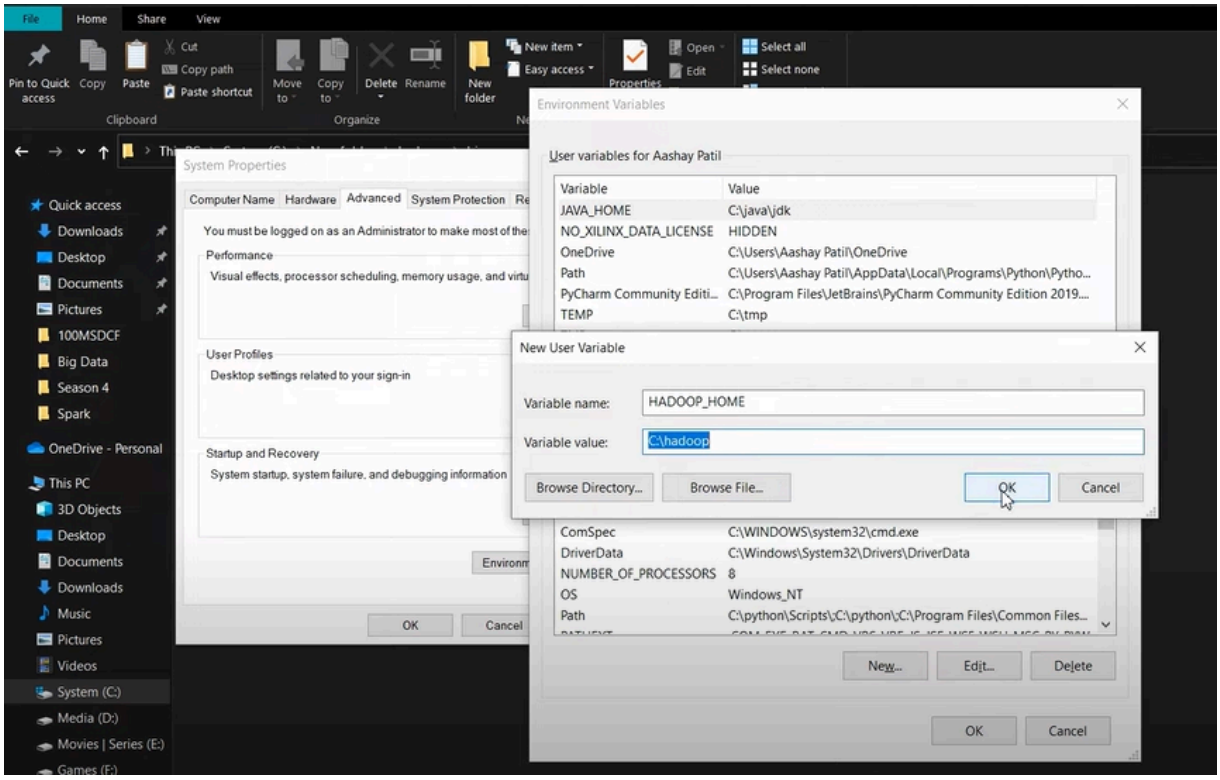
2. Go to “Edit the system environment variables” and select “Environment Variables”

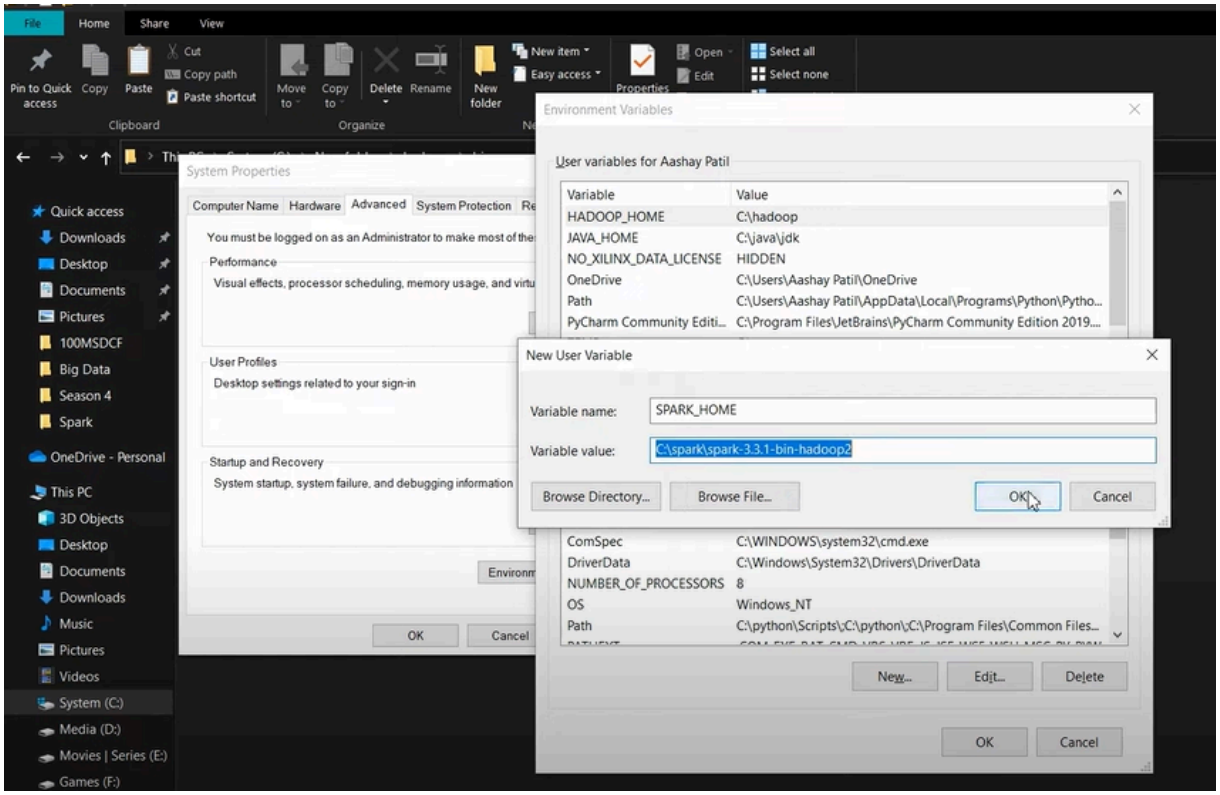


3. Java Path environment variable:
After installing Java, set the “JAVA_HOME” environment variable to the path where jdk file is present

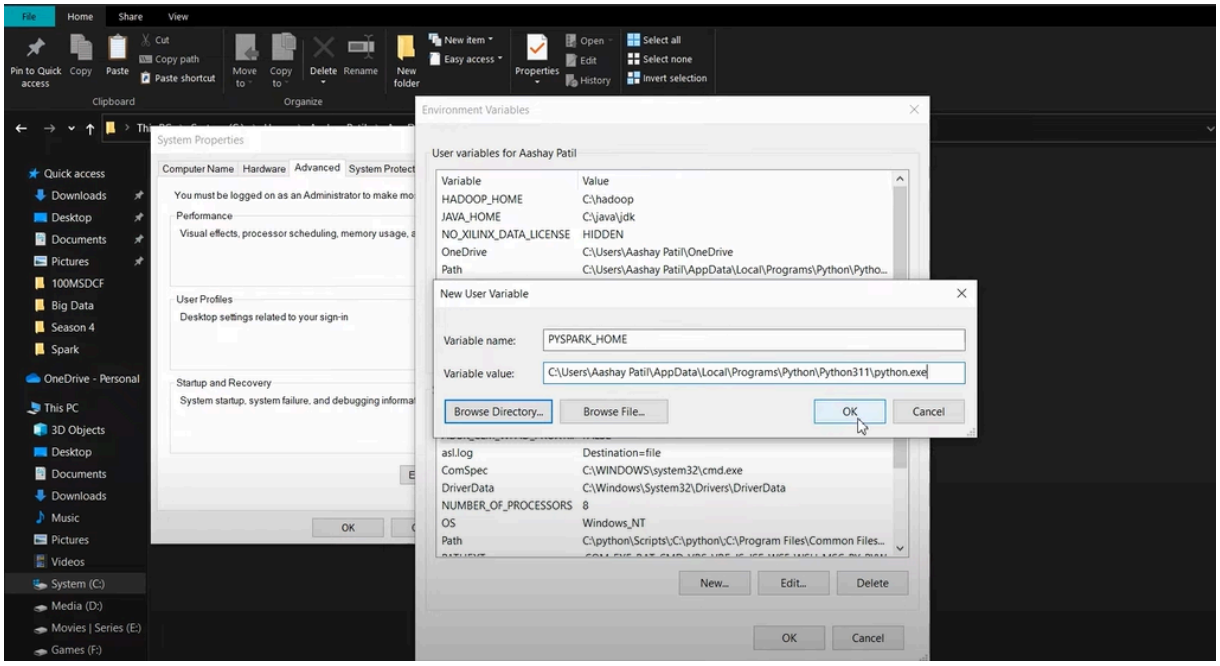


Similarly, follow the same to Hadoop and Spark



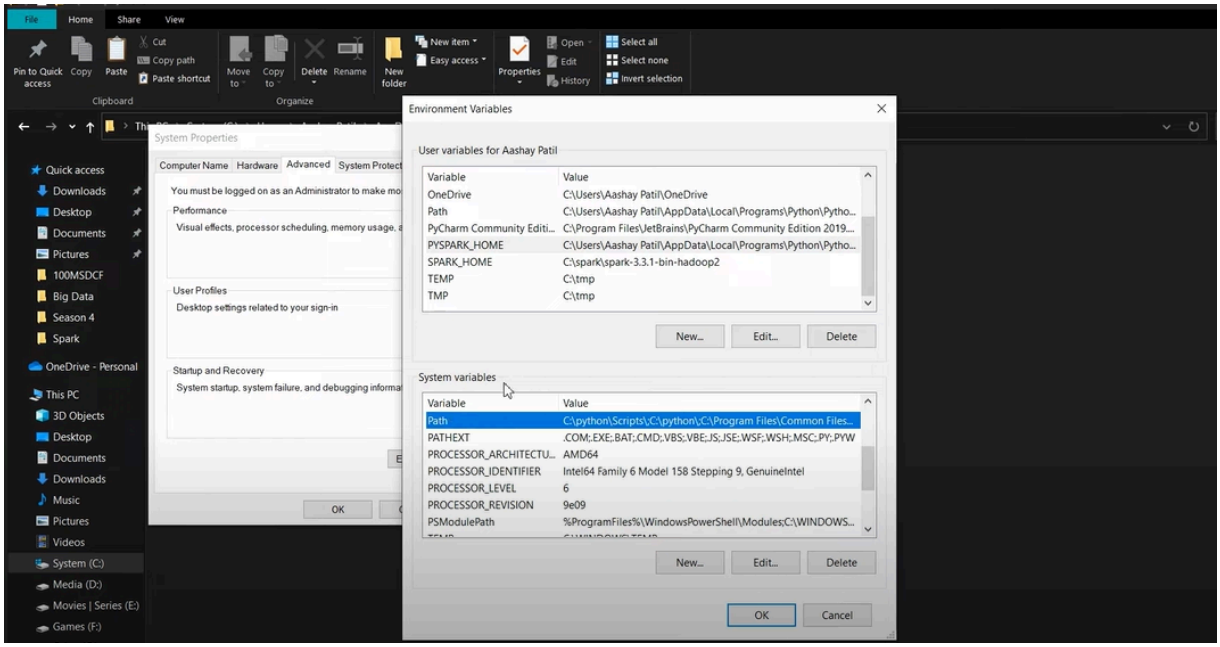


Additionally, to be on the safer side you can also add the path to the Python file. Please note that the location of the .exe file for the python version should be mentioned.

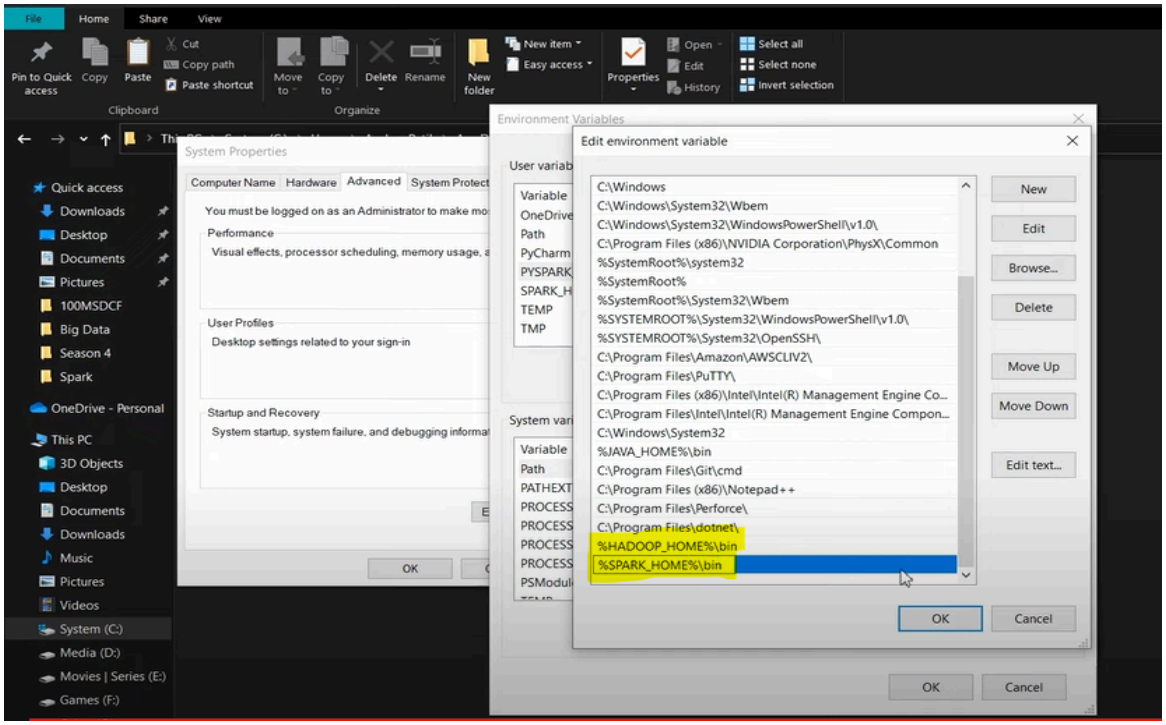


4. Provide path and the bin directories for the environments we just created:

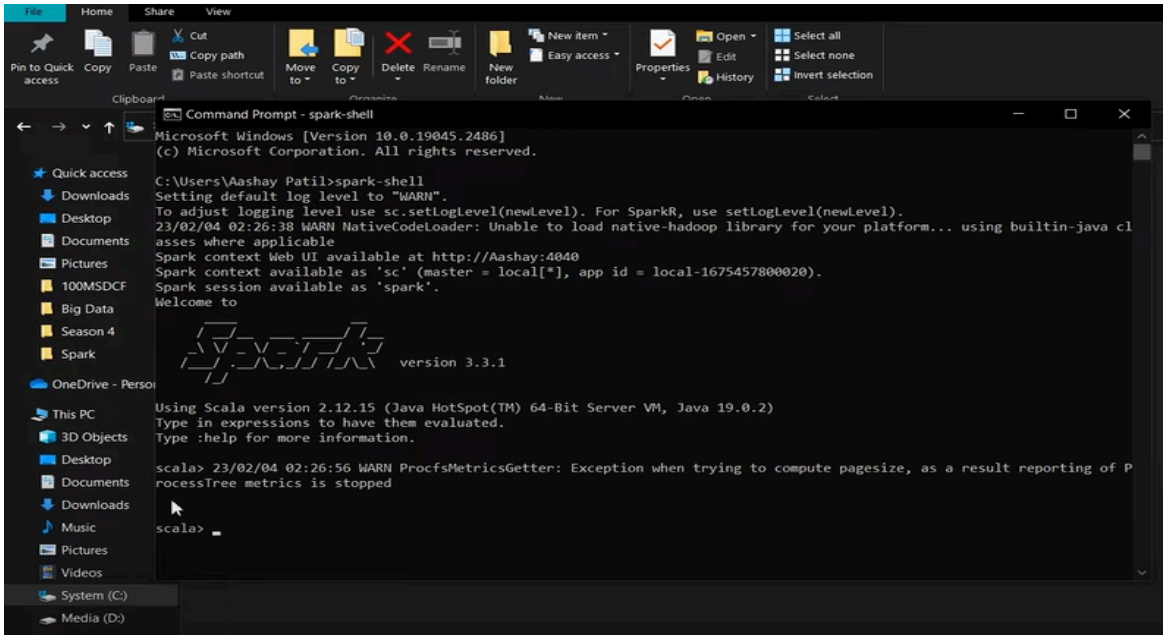
Go to “System variables” and select path, then click on the new option.



Add `%JAVA_HOME%\bin` to the “Path” environment variable, similarly for Spark and Hadoop as shown below:



To verify your installation, go to the command prompt and enter “spark-shell”. If the installation is correct then you should be getting similar output to that shown in the screenshot below:



TensorFlow

TensorFlow is an open source machine learning framework developed by Google Brain for machine learning operations.

Python can be downloaded from: https://www.python.org/downloads/

Anaconda Installation

The recommended installation procedure, as it will install Python among all other relevant dependencies. Anaconda installation for Windows can be found in the following link: Installing on Windows — Anaconda documentation

TensorFlow can be installed through Anaconda, by creating specific environments. It requires a minimum Windows distribution of Windows 10 and Python version of 3.8 for installation.

CUDA Toolkit Installation

If your computer is a CUDA-capable system, you can install TensorFlow together with the CUDA drivers, allowing for GPU acceleration with your TensorFlow operations.

Windows distributions supported:

- Microsoft Windows 11 21H2
- Microsoft Windows 11 22H2-SV2
- Microsoft Windows 10
- Microsoft Windows Server 2022
- Microsoft Windows Server 2019

CUDA installation is as follows.

a. Verify if system is CUDA-capable

   i. Open a run window from the Start Menu

   ii. Run the following command

   ```
   Control /name Microsoft.DeviceManager
   ```

   iii. Go to Display Adapters section of the Windows Device Managers

If the NVIDIA card is listed in https://developer.nvidia.com/cuda-gpus, the system is CUDA-capable.

b. Install CUDA - CUDA 11.8

Run the following command to install CUDA.

```
conda install -c conda-forge cudatoolkit==11.8 cudnn==8.9.7
```

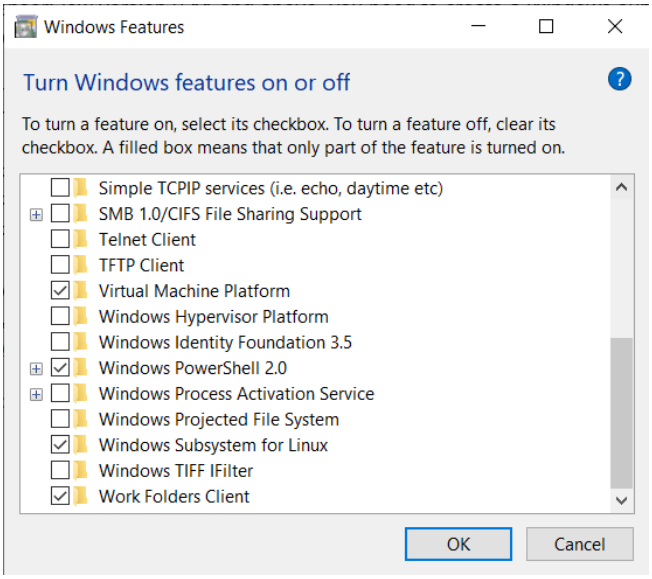
Note: Do not install TensorFlow with conda, it will not be the latest version. TensorFlow is officially released only in PyPi, so use pip for installation procedure.

Pip Installation (TensorFlow version 2.11 and above)

For TensorFlow version 2.11 and above, you need to create a Windows subsystem in Linux (WSL) for installation. The details are shown below.

Open the Windows Command Prompt/PowerShell and run the following command.

1. Turn Windows features on or off – Check the box that reads: Windows Subsystem for Linux



2. Restart your device as instructed
3. Run this command: `wsl --list -online`
4. Run the following command: `wsl --install -d Ubuntu-22.04`

Following installation, open WSL and set a username and password. If there are any errors, please troubleshoot through:
WSL/WSL/troubleshooting.md at main · MicrosoftDocs/WSL · GitHub

Steps:

1. Open WSL Shell from the Start menu.

2. pip installation: Ensure that pip(Python's package installer) is up-to-date by running:

   ```
   pip install --upgrade pip
   ```

   If pip not installed, install it:

   ```
   sudo apt install -y python3-pip
   ```

3. TensorFlow: Install TensorFlow by running

   a. For GPU installation:
   ```
   pip install tensorflow[and-cuda]
   ```

   b. For CPU installation:
   ```
   pip install tensorflow
   ```

4. Verify Installation: To check if TensorFlow was installed correctly, run the following code in python to verify:

   ```
   import tensorflow as tf
   print(tf.__version__)
   ```

Conda Installation (TensorFlow version 2.10)

TensorFlow 2.10 can be installed through the Anaconda distribution itself. Conda can be used for this installation procedure but this will not be the latest TensorFlow version.

1. For GPU Installation

   ```
   conda create -n <env_name> tensorflow-gpu
   ```

   ```
   conda activate <env_name>
   ```

2. For CPU Installation

   ```
   conda create -n <env_name> tensorflow
   ```

```
conda activate <env_name>
```

This will create an anaconda environment with TensorFlow 2.10 installed.

PyTorch

PyTorch is a machine learning framework based on the torch library originally developed by MetaAI and currently part of Linux Foundation.

It requires a minimum Windows distribution of Windows 10 and Python version of 3.8 for PyTorch 2.1.

Anaconda Installation

The recommended installation procedure, as it will install Python among all other relevant dependencies. Anaconda installation for Windows can be found in the following link: Installing on Windows — Anaconda documentation

1. Create and activate a new environment in Anaconda through the following code.

   ```
   conda create -n <env_name> anaconda
   ```

   ```
   conda activate <env_name>
   ```

   Notes:
   It is always recommended to create a new environment whenever installing any software. In case of any failure, the environment can always be deleted to prevent any problems at the root level.
   Setting the environment to anaconda enables it to install the standard packages such as numpy, pandas, scikit-learn etc.

2. Install the relevant PyTorch configuration files based on the computer's specifications.

   You can proceed with the PyTorch installation now.

- Conda Installation - CUDA 11.8

  ```
  conda install pytorch torchvision torchaudio
  pytorch-cuda=11.8 -c pytorch -c nvidia
  ```

- Pip Installation - CUDA 11.8

  ```
  pip install --upgrade pip
  ```

  ```
  pip install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu118
  ```

  If your system is not CUDA-capable, you can install the CPU version using the following code below.

- Conda Installation - CPU Installation

  ```
  conda install pytorch torchvision torchaudio cpuonly -c pytorch
  ```

- Pip Installation - CPU Installation

  ```
  pip install --upgrade pip
  ```

  ```
  pip install torch torchvision torchaudio
  ```

  Notes:
  Torchvision and torchaudio are common dependencies associated with PyTorch and can be installed concurrently.
  If using CUDA, install version 11.8. Version 12.1 has stability issues and may cause errors.

3. Verify your installation.

   You can verify your installation by running the following code.

   ```
   import torch
   ```

   ```
   print(torch.__version__)
   ```

   For CPU installation, it will return the following.

   `2.1.1+cpu` where 2.1.1 is the Torch version installed.

   For GPU installation, the code will return the following.

   `2.1.1+cn118` for PyTorch 2.1.1 and CUDA 11.8.

Additionally, you can verify GPU installation using the following code.

```
import torch

print(torch.cuda.is_available())
```

This should return True if the GPU installation is correct.