1. Download the folder vendorManagement
    1.1. Backend → Main project
    1.2. Requirements.txt
    1.3. This file

2. Create → virtual environment
    2.1. Navigate inside VendorManagement\ → open cmd
    2.2. >> python -m venv ENV

3. Activate → ENV
    3.1. >> ENV\Scripts\activate.bat → for terminal/cmd

4. Installing packages
    4.1. Manual installations
        4.1.1. Make sure ENV is active
        4.1.2. (ENV) >> python -m pip install django django-rest-framework django-cors-headers mysqlclient
    4.2. Or Try
        4.2.1. (ENV) >> python -m pip install -r requirements.txt

5. Database → mysql (used mysql Workbench)
    5.1. User → root or any other, pass → "your password"
    5.2. Create a schema named → vendormanagement
    5.3. More details → backend\backend\settings.py → DATABASES

6. Migrate changes to the database
    6.1. Navigate inside backend folder → >> cd backend
    6.2. There are 3 apps →
        6.2.1. vendorProfile
        6.2.2. vendorPurchaseOrder
        6.2.3. vendorPerformanceModel
    6.3. (ENV) >> python manage.py makemigrations
    6.4. (ENV) >> python manage.py migrate

7. Start the server
    7.1. (ENV) >> python manage.py runserver 8000

8. Django - Admin Details
    8.1. http://127.0.0.1:8000/admin/
    8.2. Username → VendorAdmin
    8.3. Password → 123
    8.4. Login to admin to view all updates.

vendorManagement Project Scenario:

Firstly, vendors will be registered

- Registered details of vendors can be seen in Vendor Profile in django admin
- Initially all performance values are set to 0

Creating / Registering a vendor

Url → http://127.0.0.1:8000/api/post-vendors/

- ➔ Name
- ➔ Contact → Phone, Email, etc
- ➔ Address

View all Registered Vendors

Url → http://127.0.0.1:8000/api/get-vendor/

View data of a specific vendor using id of that vendor

Url → http://127.0.0.1:8000/api/retrieve-vendor/5

Update data of a vendor

Url → http://127.0.0.1:8000/api/update-vendor/5

Delete a vendor

Url → http://127.0.0.1:8000/api/delete-vendor/5

Purchase Order is divided in 3 scenarios

- Take orders from client/customer
- Vendor will acknowledge client order
- Finally, after delivery, Feedback will be given by client

1) Take order from clients

<u>Client will place the order</u>

Url → http://127.0.0.1:8000/po-api/post-purchase-orders/

- ➔ Order date
- ➔ Item → format → {"A" : 1, "B" : 5 }
- ➔ Select a vendor

- When a client places an order, the issue date, acknowledge date, and delivery date are left blank.
- Quality rating and status are set to 0 and Pending respectively by default.
- Purchase Order Number (po_number) is generated automatically and quantity is calculated from items.

2) Acknowledge order from client

<u>Vendor will acknowledge the order and update dispatch details</u>

Url → http://127.0.0.1:8000/po-api/update-purchase-order-ack/**id/

- ➔ Update delivery date
- ➔ Issue date
- ➔ Acknowledge date

- Vendor updates issue date, acknowledge date and delivery date

3) Take feedback from the client and Performance of the vendor is updated.

Finally After delivery, client will provide Feedback

Url → http://127.0.0.1:8000/po-api/update-purchase-order-feedback/**id/

- ➔ Quality rating
- ➔ Status

- When client provides feedback, quality rating and status = Completed, Performance of vendor is reflected in Performance Model
- Change in issue date, acknowledge date, delivery date, quality rating and status can be seen in the Performance Model.
- Also issue date, acknowledge date, delivery date, quality rating are reflected in Vendor Profile

4) List all Orders of all vendors

Url → http://127.0.0.1:8000/po-api/get-purchase-orders/

5) Get details of a particular order

Url → http://127.0.0.1:8000/po-api/retrieve-purchase-order/**id/

6) Delete Purchase Order details

Url → http://127.0.0.1:8000/po-api/delete-purchase-order/**id/

7) List Performance of all Vendors

Url → http://127.0.0.1:8000/per-api/get-vendor-performance/