# Tài liệu mô tả các endpoint

## Tổng quan các endpoint :

Trong bài làm có tất cả 14 endpoint bao gồm:

* 8 endpoint sử dụng phương thức get gồm các Endpoint sau:
* Endpoint1: tìm kiến sản phẩm theo tên
* Endpoint2: in chi tiết hoá đơn theo OrderID
* Endpoint3: Lấy doanh thu theo thời gian
* Endpoint4: Lấy sản phẩm trong kho
* Endpoint5: Lấy danh sách sản phẩm
* Endpoint6: Lấy danh sách các khách hàng
* Endpoint7: Lấy danh sách các khách hàng mua theo từng sản phẩm
* Endpoint8: Lấy số lượng hoá đơn đã bán của nhân viên
* 6 endpoint sử dụng phương thức post gồm các Endpoint sau:
* Endpoint9: Thêm danh mục sản phẩm
* Endpoint10: Thêm shippper
* Endpoint11: Thêm sản phẩm
* Endpoint12: Thêm Khách hàng
* Endpoint13: Thêm Nhà cung cấp
* Endpoint14: Thêm hoá đơn
* 6 endpoint sử dụng numpy gồm các Endpoint sau:
* Endpoint1, Endpoint2, Endpoint4, Endpoint8, Endpoint9, Endpoint14.
* 11 endpoint sử dụng pandas gồm các Endpoint sau:
* Endpoint10, Endpoint2, Endpoint3, Endpoint4, Endpoint5, Endpoint6, Endpoint7, Endpoint8, Endpoint11, Endpoint12, Endpoint13.

## Endpoint 1: Tìm kiếm sản phẩm theo tên.

* Method: GET
* URL: /products/search
* Mô tả: Tìm kiếm thông tin về sản phẩm dựa trên tên sản phẩm.
* Tham số:
* product\_name (loại: str, query parameter): Tên sản phẩm cần tìm kiếm. Có thể truyền thông qua URL. Mặc định là None, và tối đa có thể là 40 ký tự.
* db (loại: Session, dependency): Đối tượng phiên làm việc với cơ sở dữ liệu, được đưa vào qua phương thức dependency get\_db.
* Xử lý:
* Sử dụng SQLAlchemy để truy vấn cơ sở dữ liệu và lấy thông tin sản phẩm.
* Sử dụng ilike để thực hiện tìm kiếm không phân biệt chữ hoa chữ thường.
* Sử dụng NumPy để tạo một mảng chứa tên các sản phẩm từ kết quả truy vấn.
* Trả về:
* Quantity (loại: int): Số lượng sản phẩm phù hợp với tên tìm kiếm.
* Name of products (loại: List[str]): Danh sách tên sản phẩm phù hợp.
* Product details (loại: List[Product]): Danh sách chi tiết sản phẩm từ kết quả truy vấn cơ sở dữ liệu.

## Endpoint 2: In chi tiết hoá đơn theo OrderID

* Method: GET
* URL: /orderdetail
* Mô tả: Lấy thông tin chi tiết của hóa đơn dựa trên OrderID.
* Tham số:
* db (loại: Session, dependency): Đối tượng phiên làm việc với cơ sở dữ liệu, được đưa vào qua phương thức dependency get\_db.
* orderID (loại: int, query parameter): OrderID của hóa đơn cần truy vấn.
* Xử lý:
* Sử dụng SQLAlchemy để truy vấn cơ sở dữ liệu và lấy thông tin chi tiết hóa đơn.
* Sử dụng NumPy để tạo một mảng chứa tất cả các OrderID từ cơ sở dữ liệu để kiểm tra xem OrderID được cung cấp có tồn tại hay không.
* Sử dụng Pandas để tạo DataFrame từ kết quả truy vấn để thuận tiện cho việc xử lý dữ liệu.
* Tính toán tổng giá trị đơn hàng dựa trên thông tin chi tiết sản phẩm, số lượng, giá và chiết khấu.
* Trả về:
* OrderDate (loại: str): Ngày đặt hàng của hóa đơn.
* CustomerID (loại: str): ID của khách hàng đặt hàng.
* Quantity (loại: int): Số lượng sản phẩm trong hóa đơn.
* Products (loại: List[dict]): Danh sách chi tiết sản phẩm trong hóa đơn, mỗi sản phẩm được biểu diễn dưới dạng từ điển.
* TotalOrderValue (loại: float): Tổng giá trị của hóa đơn sau khi áp dụng chiết khấu.

## Endpoint 3; Lấy doanh thu theo thời gian

* Method: GET
* URL: /revenue/{time\_period}
* Mô tả: Lấy thông tin doanh thu theo khoảng thời gian được chỉ định (ngày, tháng hoặc năm).
* Tham số:
* time\_period (loại: str, path parameter): Thời gian cần lấy doanh thu. Có thể là "daily", "monthly" hoặc "yearly".
* db (loại: Session, dependency): Đối tượng phiên làm việc với cơ sở dữ liệu, được đưa vào qua phương thức dependency get\_db.
* Xử lý:
* Sử dụng các hàm SQL để thực hiện truy vấn cơ sở dữ liệu để lấy thông tin doanh thu theo ngày, tháng hoặc năm.
* Sử dụng pandas để đọc kết quả truy vấn từ cơ sở dữ liệu và chuyển đổi chúng thành dạng DataFrame.
* Thực hiện các thao tác tiền xử lý dữ liệu, chẳng hạn như định dạng lại ngày đặt hàng, chuyển đổi kiểu dữ liệu để tránh lỗi khi sử dụng trong JSON.
* Trả về:
* Dữ liệu doanh thu được biểu diễn dưới dạng danh sách các bản ghi, mỗi bản ghi chứa thông tin về thời gian và doanh thu tương ứng.
* Xử lý lỗi:
* Nếu time\_period không phải là "daily", "monthly" hoặc "yearly", sẽ trả về một lỗi HTTP với mã 400 và thông báo "Invalid time period. Allowed values: daily, monthly, yearly".
* Nếu có lỗi trong quá trình truy vấn cơ sở dữ liệu hoặc xử lý dữ liệu, sẽ trả về lỗi HTTP với mã 500 và thông báo lỗi cụ thể.

## Endpoint 4: Lấy sản phẩm trong kho

* Method: GET
* URL: /product/stock
* Mô tả: Lấy thông tin về tổng số lượng sản phẩm tồn kho và chi tiết từng sản phẩm.
* Tham số:
* db (loại: Session, dependency): Đối tượng phiên làm việc với cơ sở dữ liệu, được đưa vào qua phương thức dependency get\_db.
* Xử lý:
* Sử dụng câu truy vấn SQL để lấy thông tin về ProductID, ProductName, và UnitsInStock từ bảng Products.
* Sử dụng pandas để đọc kết quả truy vấn từ cơ sở dữ liệu và chuyển đổi chúng thành dạng DataFrame.
* Tạo danh sách product\_data chứa thông tin chi tiết về sản phẩm từ DataFrame.
* Sử dụng hàm calculate\_total\_stock để tính tổng số lượng sản phẩm tồn kho từ danh sách product\_data.
* Trả về:
* Total\_stock (loại: int): Tổng số lượng sản phẩm tồn kho.
* Product (loại: List[dict]): Danh sách các sản phẩm và thông tin chi tiết của chúng, mỗi sản phẩm được biểu diễn dưới dạng từ điển.
* Xử lý lỗi:
* Nếu có lỗi trong quá trình truy vấn cơ sở dữ liệu hoặc xử lý dữ liệu, sẽ trả về lỗi HTTP với mã 500 và thông báo lỗi cụ thể.

## Endpoint 5: Lấy danh sách hoá đơn theo từng nhân viên.

* Method: GET
* URL: /employee-invoices/
* Mô tả: Lấy danh sách các chi tiết hóa đơn của một nhân viên dựa trên ID nhân viên.
* Tham số:
* employee\_id (loại: int, query parameter): ID của nhân viên cần lấy danh sách hóa đơn.
* db (loại: Session, dependency): Đối tượng phiên làm việc với cơ sở dữ liệu, được đưa vào qua phương thức dependency get\_db.
* Xử lý:
* Sử dụng câu truy vấn SQL để lấy thông tin chi tiết hóa đơn từ bảng orderdetails dựa trên EmployeeID từ bảng orders.
* Thực hiện truy vấn SQL và lấy kết quả sử dụng pandas.
* Chuyển kết quả thành danh sách các bản ghi (records) để trả về.
* Trả về:
* Một danh sách các chi tiết hóa đơn liên quan đến nhân viên có ID tương ứng.
* Chú ý:
* Sử dụng response\_model để xác định kiểu dữ liệu trả về của endpoint là một danh sách các đối tượng của kiểu OrderDetails được định nghĩa trong schema.
* Có thể cần thêm xác thực và quản lý lỗi cho việc xử lý truy vấn cơ sở dữ liệu và xử lý dữ liệu.

## Endpoint 6: lấy danh sách các khách hàng.

* Method: GET
* URL: /customers/
* Mô tả: Lấy danh sách thông tin khách hàng từ cơ sở dữ liệu.
* Tham số:
* skip (loại: int, query parameter): Số lượng bản ghi muốn bỏ qua từ đầu danh sách (mặc định là 0).
* limit (loại: int, query parameter): Số lượng bản ghi muốn lấy (mặc định là 50).
* db (loại: Session, dependency): Đối tượng phiên làm việc với cơ sở dữ liệu, được đưa vào qua phương thức dependency get\_db.
* Xử lý:
* Sử dụng SQLAlchemy để thực hiện truy vấn cơ sở dữ liệu và lấy thông tin của khách hàng.
* Sử dụng offset và limit để xác định phạm vi của kết quả truy vấn.
* Tạo DataFrame từ kết quả truy vấn để thuận tiện cho việc xử lý dữ liệu.
* Điều chỉnh các giá trị cần thiết (như Fax và PostalCode) để đảm bảo tính toàn vẹn của dữ liệu.
* Trả về:
* Một danh sách các khách hàng dưới dạng records, mỗi khách hàng được biểu diễn dưới dạng từ điển.
* Chú ý:
* Sử dụng response\_model để xác định kiểu dữ liệu trả về của endpoint là một danh sách các đối tượng của kiểu Customer được định nghĩa trong schema.
* Có thể cần thêm xác thực và quản lý lỗi cho việc xử lý truy vấn cơ sở dữ liệu và xử lý dữ liệu.

## Endpoint 7: Lấy danh sách khách hàng đã mua sản phẩm theo từng sản phẩm

* Method: GET
* URL: /product-customers/
* Mô tả: Lấy danh sách thông tin khách hàng đã mua một sản phẩm cụ thể dựa trên mã sản phẩm.
* Tham số:
* product\_id (loại: int, query parameter): Mã sản phẩm cần lấy thông tin về khách hàng.
* db (loại: Session, dependency): Đối tượng phiên làm việc với cơ sở dữ liệu, được đưa vào qua phương thức dependency get\_db.
* Xử lý:
* Sử dụng câu truy vấn SQL để lấy thông tin về khách hàng từ bảng customers dựa trên mối quan hệ với bảng orders và orderdetails.
* Thực hiện truy vấn SQL và lấy kết quả sử dụng pandas.
* Chuyển kết quả thành danh sách các bản ghi (records) để trả về.
* Trả về:
* Một danh sách các khách hàng (dưới dạng records) liên quan đến sản phẩm có mã tương ứng.
* Chú ý:
* Sử dụng response\_model để xác định kiểu dữ liệu trả về của endpoint là một danh sách các đối tượng của kiểu Customer được định nghĩa trong schema.
* Có thể cần thêm xác thực và quản lý lỗi cho việc xử lý truy vấn cơ sở dữ liệu và xử lý dữ liệu.

## Endpoint 8: lấy số lượng hoá đơn đã bán của nhân viên

* Method: GET
* URL: /employee-invoice-count/
* Mô tả: Lấy số lượng hóa đơn của một nhân viên dựa trên mã nhân viên.
* Tham số:
* employee\_id (loại: int, query parameter): Mã nhân viên cần lấy thông tin về số lượng hóa đơn.
* db (loại: Session, dependency): Đối tượng phiên làm việc với cơ sở dữ liệu, được đưa vào qua phương thức dependency get\_db.
* Xử lý:
* Sử dụng câu truy vấn SQL để lấy thông tin về EmployeeID và OrderID từ bảng orders dựa trên mã nhân viên.
* Thực hiện truy vấn SQL và lấy kết quả sử dụng pandas.
* Sử dụng NumPy để đếm số lượng hóa đơn từ kết quả truy vấn.
* Trả về:
* Một đối tượng JSON chứa thông tin về số lượng hóa đơn của nhân viên có mã tương ứng.
* Chú ý:
* Có thể cần thêm xác thực và quản lý lỗi cho việc xử lý truy vấn cơ sở dữ liệu và xử lý dữ liệu.
* Lưu ý rằng cách đếm số lượng hóa đơn ở đây có thể được thay đổi để đảm bảo độ chính xác của dữ liệu (ví dụ, sử dụng result.shape[0] thay vì np.array([result["OrderID"]]).size).

## Endpoint 9: Thêm danh mục sản phẩm

* Method: POST
* URL: /category
* Mô tả: Thêm mới một danh mục vào cơ sở dữ liệu.
* Tham số:
* cate (loại: CategoryCreate, body parameter): Đối tượng CategoryCreate từ schema, chứa thông tin về danh mục cần thêm mới.
* db (loại: Session, dependency): Đối tượng phiên làm việc với cơ sở dữ liệu, được đưa vào qua phương thức dependency get\_db.
* Xử lý:
* Kiểm tra xem tên danh mục (CategoryName) và mô tả (Description) có được cung cấp hay không. Nếu không, trả về lỗi HTTP với mã 400 và thông báo "All fields must be provided".
* Kiểm tra xem danh mục với tên đã tồn tại hay chưa trong cơ sở dữ liệu. Nếu đã tồn tại, trả về lỗi HTTP với mã 400 và thông báo "CategoryName already exists".
* Nếu chiều dài của tên danh mục (CategoryName) lớn hơn 15 ký tự, cắt bớt nó để đảm bảo rằng nó không vượt quá giới hạn chiều dài.
* Thêm mới đối tượng danh mục vào cơ sở dữ liệu, commit thay đổi.
* Làm mới đối tượng để lấy thông tin đã được lưu vào cơ sở dữ liệu.
* Trả về:
* Đối tượng danh mục mới được thêm vào cơ sở dữ liệu.
* Xử lý lỗi:
* Nếu có lỗi trong quá trình thêm mới danh mục, rollback thay đổi và trả về lỗi HTTP với mã 400 và thông báo lỗi cụ thể.

## Endpoint 10: Thêm shipper

* Method: POST
* URL: /Shipper/upload-data
* Mô tả: Thêm mới thông tin của các nhà vận chuyển từ file CSV vào cơ sở dữ liệu.
* Tham số:
* file (loại: UploadFile, body parameter): File CSV chứa thông tin về các nhà vận chuyển cần thêm mới.
* db (loại: Session, dependency): Đối tượng phiên làm việc với cơ sở dữ liệu, được đưa vào qua phương thức dependency get\_db.
* Xử lý:
* Kiểm tra xem file có phải là file CSV hay không dựa trên đuôi file. Nếu không, trả về lỗi HTTP với mã 400 và thông báo "File is not a CSV".
* Sử dụng thư viện pandas để đọc dữ liệu từ file CSV thành DataFrame.
* Duyệt qua từng dòng trong DataFrame, kiểm tra xem thông tin nhà vận chuyển đã tồn tại trong cơ sở dữ liệu hay chưa. Nếu chưa tồn tại, thêm thông tin nhà vận chuyển vào danh sách shippers.
* Nếu danh sách shippers rỗng, đồng nghĩa với việc không có dữ liệu mới hoặc dữ liệu đã tồn tại trong cơ sở dữ liệu. Trả về lỗi HTTP với mã 400 và thông báo "Data already exists or the file does not have matching data".
* Thêm danh sách **shippers** vào cơ sở dữ liệu và commit thay đổi.
* Trả về:
* Thông báo "CSV file uploaded and shippers added successfully" khi quá trình thêm mới thành công.
* Xử lý lỗi:
* Nếu có lỗi trong quá trình thêm mới thông tin nhà vận chuyển, rollback thay đổi và trả về lỗi HTTP với mã 400 và thông báo lỗi cụ thể.

## Endpoint 11: Thêm sản phẩm

* Method: POST
* URL: /Shipper/upload-data
* Mô tả: Thêm mới thông tin của các nhà vận chuyển từ file CSV vào cơ sở dữ liệu.
* Tham số:
* file (loại: UploadFile, body parameter): File CSV chứa thông tin về các nhà vận chuyển cần thêm mới.
* db (loại: Session, dependency): Đối tượng phiên làm việc với cơ sở dữ liệu, được đưa vào qua phương thức dependency get\_db.
* Xử lý:
* Kiểm tra xem file có phải là file CSV hay không dựa trên đuôi file. Nếu không, trả về lỗi HTTP với mã 400 và thông báo "File is not a CSV".
* Sử dụng thư viện pandas để đọc dữ liệu từ file CSV thành DataFrame.
* Duyệt qua từng dòng trong DataFrame, kiểm tra xem thông tin nhà vận chuyển đã tồn tại trong cơ sở dữ liệu hay chưa. Nếu chưa tồn tại, thêm thông tin nhà vận chuyển vào danh sách shippers.
* Nếu danh sách shippers rỗng, đồng nghĩa với việc không có dữ liệu mới hoặc dữ liệu đã tồn tại trong cơ sở dữ liệu. Trả về lỗi HTTP với mã 400 và thông báo "Data already exists or the file does not have matching data".
* Thêm danh sách shippers vào cơ sở dữ liệu và commit thay đổi.
* Trả về :
* Thông báo "CSV file uploaded and shippers added successfully" khi quá trình thêm mới thành công.
* Xử lý lỗi:
* Nếu có lỗi trong quá trình thêm mới thông tin nhà vận chuyển, rollback thay đổi và trả về lỗi HTTP với mã 400 và thông báo lỗi cụ thể.

## Endpoin 12: Thêm Khách Hàng

* Method: POST
* URL: /customers/upload-data
* Mô tả: Thêm mới thông tin của khách hàng từ file CSV vào cơ sở dữ liệu.
* Tham số:
* file (loại: UploadFile, body parameter): File CSV chứa thông tin về khách hàng cần thêm mới.
* db (loại: Session, dependency): Đối tượng phiên làm việc với cơ sở dữ liệu, được đưa vào qua phương thức dependency get\_db.
* Xử lý:
* Kiểm tra xem file có phải là file CSV hay không dựa trên đuôi file. Nếu không, trả về lỗi HTTP với mã 400 và thông báo "File is not a CSV".
* Sử dụng thư viện pandas để đọc dữ liệu từ file CSV thành DataFrame.
* Kiểm tra xem các cột bắt buộc (CustomerID, CompanyName, ContactName, ContactTitle, Address, City, PostalCode, Country, Phone, Fax) có tồn tại trong DataFrame hay không. Nếu không, trả về lỗi HTTP với mã 422 và thông báo "Missing columns" kèm theo các cột thiếu.
* Chuyển DataFrame thành danh sách các từ điển (records) chứa thông tin về khách hàng.
* Sử dụng unpacking (\*\*) để tạo danh sách các đối tượng Customer từ danh sách từ điển trên.
* Thêm danh sách các đối tượng Customer vào cơ sở dữ liệu và commit thay đổi.
* Trả về:
* Một đối tượng JSON chứa thông báo "Thêm khách hàng thành công!" và số lượng khách hàng đã được thêm mới.
* Xử lý lỗi:
* Nếu có lỗi trong quá trình thêm mới thông tin khách hàng, rollback thay đổi và trả về lỗi HTTP với mã 500 và thông báo lỗi cụ thể.

## Endpoin 13: Thêm Nhà cung Cấp

* Method: POST
* URL: /supplier
* Mô tả: Tạo mới thông tin của một nhà cung cấp vào cơ sở dữ liệu.
* Tham số:
* sup (loại: SupplierCreate, body parameter): Đối tượng SupplierCreate từ schema, chứa thông tin về nhà cung cấp cần tạo mới.
* db (loại: Session, dependency): Đối tượng phiên làm việc với cơ sở dữ liệu, được đưa vào qua phương thức dependency get\_db.
* Xử lý:
* Kiểm tra xem nhà cung cấp có tồn tại trong cơ sở dữ liệu hay không, dựa trên các trường thông tin như CompanyName, ContactName, ContactTitle, và PostalCode. Nếu tồn tại, trả về lỗi HTTP với mã 400 và thông báo "This supplier already exists in the database."
* Chuyển thông tin từ đối tượng SupplierCreate thành DataFrame để kiểm tra chiều dài của PostalCode.
* Nếu chiều dài của PostalCode lớn hơn 10 ký tự, trả về lỗi HTTP với mã 400 và thông báo "PostalCode length limited to 10 characters."
* Chuyển DataFrame thành từ điển và tạo mới đối tượng nhà cung cấp (Supplier) trong cơ sở dữ liệu.
* Commit thay đổi.
* Trả về:
* Một đối tượng JSON chứa thông báo "Add supplier is successful" và thông tin chi tiết về nhà cung cấp đã được tạo mới.
* Xử lý lỗi:
* Nếu có lỗi trong quá trình tạo mới thông tin nhà cung cấp, rollback thay đổi và trả về lỗi HTTP với mã 400 và thông báo lỗi cụ thể.

## Endpoin 14: Thêm Hoá đơn

* Method: POST
* URL: /orders/
* Mô tả: Tạo mới thông tin đơn đặt hàng và chi tiết đơn hàng vào cơ sở dữ liệu.
* Tham số:
* **order** (loại: OrderCreate, body parameter): Đối tượng OrderCreate từ schema, chứa thông tin về đơn đặt hàng cần tạo mới.
* **db** (loại: Session, dependency): Đối tượng phiên làm việc với cơ sở dữ liệu, được đưa vào qua phương thức dependency **get\_db**.
* Xử lý:
* Validate dữ liệu đơn đặt hàng sử dụng thư viện NumPy, đảm bảo rằng trường **OrderDate** không được là **None**.
* Tạo một đối tượng **Orders** mới từ dữ liệu của đơn đặt hàng và thêm nó vào cơ sở dữ liệu.
* Commit thay đổi và làm mới đối tượng **Orders** để lấy thông tin đã được lưu vào cơ sở dữ liệu.
* Duyệt qua từng sản phẩm trong danh sách **Products** của đơn đặt hàng, kiểm tra tính hợp lệ của dữ liệu sản phẩm sử dụng NumPy (giá và số lượng phải lớn hơn 0).
* Tạo mới các đối tượng **OrderDetails** cho mỗi sản phẩm trong đơn đặt hàng và thêm chúng vào cơ sở dữ liệu.
* Commit thay đổi và làm mới đối tượng **Orders** để lấy thông tin đã được cập nhật từ cơ sở dữ liệu.
* Trả về:
* Đối tượng JSON chứa thông tin về đơn đặt hàng đã được tạo mới (sau khi đã được làm mới).
* Xử lý lỗi:
* Nếu có lỗi trong quá trình tạo mới thông tin đơn đặt hàng hoặc chi tiết đơn hàng, rollback thay đổi và trả về lỗi HTTP với mã 400 và thông báo lỗi cụ thể.

# Tài liệu mô Tả Swagger UI

## **Tìm kiếm sản phẩm theo tên**

* **Endpoint:** GET /products/search
* **Mô tả:** Tìm kiếm thông tin sản phẩm theo tên sản phẩm. Sử dụng thư viện NumPy để thực hiện các thao tác trên dữ liệu.
* **Đầu vào:** product\_name (query parameter, kiểu: str, tùy chọn): Tên sản phẩm cần tìm kiếm.
* **Đầu ra:** Dữ liệu JSON chứa thông tin về số lượng sản phẩm, tên các sản phẩm và chi tiết các sản phẩm.

## In chi tiết hóa đơn theo OrderID

* **Endpoint:** GET /orderdetail
* **Mô tả:** Lấy thông tin chi tiết của hóa đơn dựa trên OrderID. Sử dụng thư viện NumPy và Pandas để xử lý dữ liệu.
* **Đầu vào:**orderID (query parameter, kiểu: int, bắt buộc): ID của hóa đơn cần lấy thông tin chi tiết.
* **Đầu ra:**Dữ liệu JSON chứa thông tin như ngày đặt hàng, mã khách hàng, số lượng sản phẩm, danh sách sản phẩm và tổng giá trị đơn hàng.

## Lấy doanh thu theo thời gian

* **Endpoint:** GET /revenue/{time\_period}
* **Mô tả:** Lấy thông tin doanh thu theo khoảng thời gian (ngày, tháng, hoặc năm). Sử dụng thư viện Pandas để thực hiện các thao tác trên dữ liệu.
* **Đầu vào:** time\_period (path parameter, kiểu: str, bắt buộc): Khoảng thời gian cần lấy doanh thu (daily, monthly, hoặc yearly).
* **Đầu ra:** Dữ liệu JSON chứa thông tin doanh thu theo khoảng thời gian đã chọn.

## Lấy sản phẩm trong kho

* **Endpoint:** GET /product/stock
* **Mô tả:** Lấy tổng số lượng sản phẩm trong kho và danh sách sản phẩm. Sử dụng thư viện NumPy để tính toán tổng số lượng sản phẩm.
* **Đầu vào:** Không có.
* **Đầu ra:**Dữ liệu JSON chứa thông tin về tổng số lượng sản phẩm và danh sách sản phẩm trong kho.

## Lấy danh sách hoá đơn theo từng nhân viên

* **Endpoint:** GET /employee-invoices/
* **Mô tả:** Lấy danh sách hoá đơn của một nhân viên dựa trên ID của nhân viên đó.
* **Đầu vào:**employee\_id (query parameter, kiểu: int, bắt buộc): ID của nhân viên cần lấy danh sách hoá đơn.
* **Đầu ra:**Dữ liệu JSON chứa danh sách các hoá đơn của nhân viên.

## Lấy danh sách khách hàng

* **Endpoint:** GET /customers/
* **Mô tả:** Lấy danh sách khách hàng với khả năng skip và limit để phân trang. Sử dụng thư viện Pandas để chuyển đổi dữ liệu từ SQLAlchemy thành JSON.
* **Đầu vào:**
* skip (query parameter, kiểu: int, tùy chọn): Số lượng bản ghi cần bỏ qua.
* limit (query parameter, kiểu: int, tùy chọn): Số lượng bản ghi cần lấy.
* **Đầu ra:** Dữ liệu JSON chứa danh sách khách hàng.

## Lấy danh sách khách hàng đã mua sản phẩm

* **Endpoint:** GET /product-customers/
* **Mô tả:** Lấy danh sách khách hàng đã mua sản phẩm dựa trên mã sản phẩm. Sử dụng thư viện Pandas để thực hiện truy vấn SQL và chuyển đổi dữ liệu thành JSON.
* **Đầu vào:**product\_id (query parameter, kiểu: int, bắt buộc): ID của sản phẩm cần kiểm tra.
* **Đầu ra:**Dữ liệu JSON chứa danh sách các khách hàng đã mua sản phẩm.

## Lấy số lượng hoá đơn đã bán của nhân viên

* **Endpoint:** GET /employee-invoice-count/
* **Mô tả:** Lấy số lượng hoá đơn đã bán của một nhân viên dựa trên ID của nhân viên.
* **Đầu vào:**employee\_id (query parameter, kiểu: int, bắt buộc): ID của nhân viên cần kiểm tra.
* **Đầu ra:**Dữ liệu JSON chứa số lượng hoá đơn đã bán của nhân viên.

## Thêm mới danh mục sản phẩm

* **Endpoint:** POST /category
* **Mô tả:** Thêm mới danh mục sản phẩm vào cơ sở dữ liệu.
* **Đầu vào:** Dữ liệu JSON chứa thông tin về danh mục sản phẩm, bao gồm CategoryName và Description.
* **Đầu ra:** Dữ liệu JSON chứa thông tin về danh mục sản phẩm đã được thêm mới.

## Thêm shipper

* **Endpoint:** POST /Shipper/upload-data
* **Mô tả:** Thêm thông tin shipper từ file CSV vào cơ sở dữ liệu. Sử dụng thư viện Pandas để đọc dữ liệu từ file CSV và thao tác trên dữ liệu.
* **Đầu vào:** file (formData, kiểu: file, bắt buộc): File CSV chứa thông tin về shipper.
* **Đầu ra:**
* Nếu quá trình thêm thành công, trả về thông báo "CSV file uploaded and shippers added successfully".
* Nếu không có dữ liệu mới hoặc dữ liệu trong file không phù hợp, trả về lỗi với mã status\_code 400.

## Thêm sản phẩm

* **Endpoint:** POST /products/upload-data
* **Mô tả:** Thêm thông tin sản phẩm từ file CSV vào cơ sở dữ liệu. Sử dụng thư viện Pandas để đọc dữ liệu từ file CSV và thao tác trên dữ liệu.
* **Đầu vào:** file (formData, kiểu: file, bắt buộc): File CSV chứa thông tin về sản phẩm.
* **Đầu ra:**
* Nếu quá trình thêm thành công, trả về thông báo "CSV file uploaded success".
* Nếu không có dữ liệu mới hoặc dữ liệu trong file không phù hợp, trả về lỗi với mã status\_code 400 hoặc 500 nếu có lỗi ngoại lệ.

## Thêm khách hàng

* **Endpoint:** POST /customers/upload-data
* **Mô tả:** Thêm thông tin khách hàng từ file CSV vào cơ sở dữ liệu. Sử dụng thư viện Pandas để đọc dữ liệu từ file CSV và thao tác trên dữ liệu.
* **Đầu vào:** file (formData, kiểu: file, bắt buộc): File CSV chứa thông tin về khách hàng.
* **Đầu ra:**
* Nếu quá trình thêm thành công, trả về thông báo "Thêm khách hàng thành công!" và số lượng khách hàng đã được thêm.
* Nếu không có dữ liệu mới hoặc dữ liệu trong file không phù hợp, trả về lỗi với mã status\_code 400 hoặc 500 nếu có lỗi ngoại lệ.

## Thêm nhà cung cấp

* **Endpoint:** POST /supplier
* **Mô tả:** Thêm thông tin nhà cung cấp vào cơ sở dữ liệu. Sử dụng thư viện Pandas để chuyển đổi dữ liệu đầu vào từ Pydantic Model sang DataFrame và kiểm tra chiều dài của PostalCode.
* **Đầu vào:** Dữ liệu JSON chứa thông tin về nhà cung cấp, bao gồm CompanyName, ContactName, ContactTitle, PostalCode.
* **Đầu ra:**
* Nếu quá trình thêm thành công, trả về thông báo "Add supplier is successful" và chi tiết về nhà cung cấp đã thêm.
* Nếu nhà cung cấp đã tồn tại hoặc chiều dài của PostalCode vượt quá 10 ký tự, trả về lỗi với mã status\_code 400.

## Thêm hoá đơn

* **Endpoint:** POST /orders/
* **Mô tả:** Thêm thông tin hoá đơn và chi tiết hoá đơn vào cơ sở dữ liệu. Sử dụng thư viện NumPy để kiểm tra tính hợp lệ của dữ liệu và Pandas để thao tác trên dữ liệu.
* **Đầu vào:** Dữ liệu JSON chứa thông tin về hoá đơn, bao gồm OrderDate và danh sách sản phẩm (Products).
* **Đầu ra:**
* Dữ liệu JSON chứa thông tin về hoá đơn đã được thêm vào cơ sở dữ liệu.

Top of Form