Given the following DDL, your task is to create a view that will give the user an option to send orders to fulfillment per the below specs.The fulfillment API endpoint will be: http://localhost/orders. The request body should be in json and contain the following root elements:OrderIdFirstNameLastNameAddressCityStateCountryProducts SKU QuantityThe process should be intelligent enough to group products into same shipments by address and by product category. For example, if there are multiple orders with the same shipping address, the shipments should be grouped together as long as the product categories are the same. Products not in the same category should be in separate shipments (separate API call).Since you are calling http://localhost/orders, the API call does not have to return a 200, but you must be able to show the request of the API call and the attempt.

Bonus points if you are able to use the Unit of Work / Repository pattern for data access.

DDL TO BE FOLLOWED:

DROP TABLE IF EXISTS TestOrders  
DROP TABLE IF EXISTS TestProducts  
DROP TABLE IF EXISTS TestCategories  
DROP TABLE IF EXISTS TestProductCategories  
DROP TABLE IF EXISTS TestOrderProducts  
  
DECLARE @OrderId int  
CREATE TABLE TestOrders(id int IDENTITY, FirstName varchar(255) , LastName varchar(255), Address varchar(255), City varchar(255), State varchar(255), Country varchar(255))  
CREATE TABLE TestProducts (id int IDENTITY, Name varchar(255), SKU varchar(255), Description varchar(255))  
CREATE TABLE TestCategories (id int IDENTITY, Name varchar(255))  
CREATE TABLE TestProductCategories(ProductId int, CategoryId int)  
CREATE TABLE TestOrderProducts(id int IDENTITY, OrderId int, ProductId int, Quantity int, Price money, Total money)  
  
INSERT INTO TestOrders(FirstName, LastName, Address, City, State, Country)  
SELECT 'Jeff', 'Cheung', '150 Golf Course Rd', 'South Burlington', 'VT', 'USA'  
SET @OrderId = @@IDENTITY  
  
INSERT INTO TestOrderProducts(OrderId, ProductId, Quantity, Price, Total)  
VALUES(@OrderId, 1, 1, 9.99, 9.99)  
  
INSERT INTO TestOrderProducts(OrderId, ProductId, Quantity, Price, Total)  
VALUES(@OrderId, 3, 1, 4.99, 9.99)  
  
INSERT INTO TestOrders(FirstName, LastName, Address, City, State, Country)  
SELECT 'Jeff', 'Cheung', '150 Golf Course Rd', 'South Burlington', 'VT', 'USA'  
  
SET @OrderId = @@IDENTITY  
  
INSERT INTO TestOrderProducts(OrderId, ProductId, Quantity, Price, Total)  
VALUES(@OrderId, 2, 1, 9.99, 9.99)  
  
INSERT INTO TestOrders(FirstName, LastName, Address, City, State, Country)  
SELECT 'Jeff', 'Cheung', '150 Dorset St Ste 245-236', 'South Burlington', 'VT', 'USA'  
SET @OrderId = @@IDENTITY  
  
INSERT INTO TestOrderProducts(OrderId, ProductId, Quantity, Price, Total)  
VALUES(@OrderId, 1, 1, 9.99, 9.99)  
  
INSERT INTO TestOrderProducts(OrderId, ProductId, Quantity, Price, Total)  
VALUES(@OrderId, 3, 1, 4.99, 9.99)  
  
INSERT INTO TestOrderProducts(OrderId, ProductId, Quantity, Price, Total)  
VALUES(@OrderId, 2, 1, 9.99, 9.99)  
  
INSERT INTO TestCategories(Name)  
SELECT ('Category 1')  
UNION  
SELECT ('Category 2')  
  
INSERT INTO TestProducts(Name, SKU, Description)  
SELECT 'Argan Product', 'Argan', 'Argan Test Product'  
UNION  
SELECT 'Argan Product 2', 'Argan2', 'Argan Test Product #2'  
UNION  
SELECT 'Idro Product', 'Idro', 'Idro Test Product'  
  
INSERT INTO TestProductCategories(ProductId, CategoryId)  
SELECT 1, 1  
UNION  
SELECT 2, 1  
UNION  
SELECT 3, 2  
  
that is the ddl that needs to be included