Feign Client

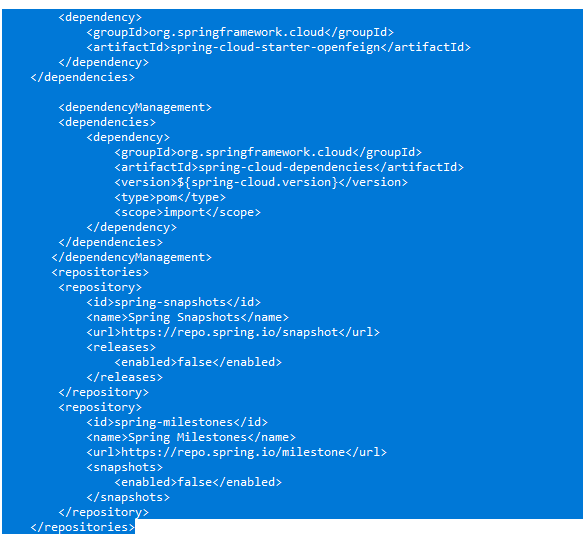
Para ello hay que agregar la dependencia del openfeign.

<dependency>

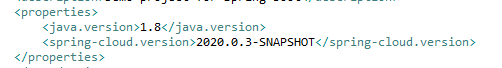
<groupId>org.springframework.cloud</groupId>

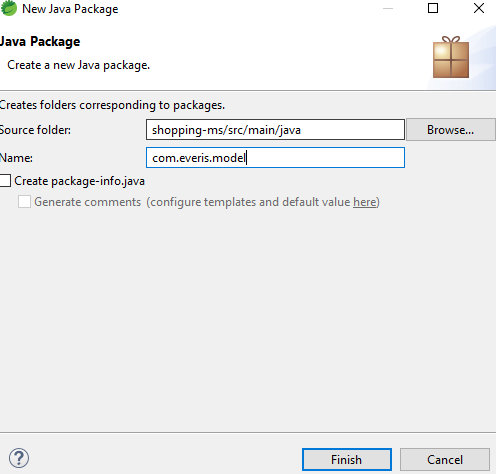
<artifactId>spring-cloud-starter-openfeign</artifactId>

</dependency>

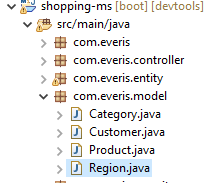


No olvidar agregar la versión

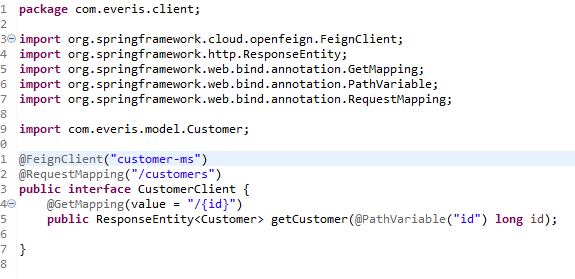


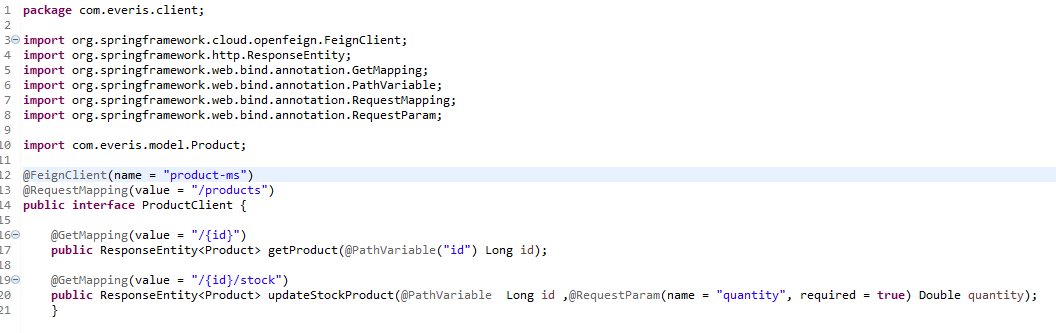


Agregamos la estructrura de las clases entity a usar en de los microservicios a consumir

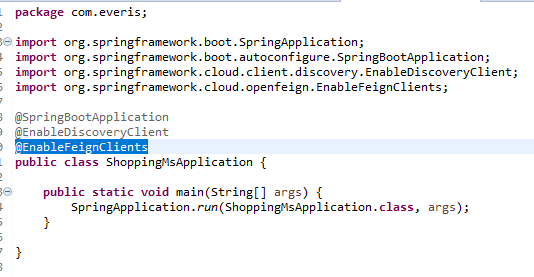


Se agrega las interfaces que harán uso del consumo de los microservicios





En la clase principal habilitamos los clientes de feign. @EnableFeignClients



Modificando la clase servicio InvoiceServiceImpl para consumir las clases feign

@Autowired

CustomerClient customerClient;

@Autowired

ProductClient productClient;

Modificando la clase de creación, para que se actualice el stock

@Override

**public** Invoice createInvoice(Invoice invoice) {

Invoice invoiceDB = invoiceRepository.findByNumberInvoice ( invoice.getNumberInvoice () );

**if** (invoiceDB !=**null**){

**return** invoiceDB;

}

invoice.setState("CREATED");

invoiceDB = invoiceRepository.save(invoice);

invoiceDB.getItems().forEach( invoiceItem -> {

productClient.updateStockProduct( invoiceItem.getProductId(), invoiceItem.getQuantity() \* -1);

});

**return** invoiceRepository.save(invoice);

}

Para obtener la factura con los campos completos:

@Override

**public** Invoice getInvoice(Long id) {

Invoice invoice= invoiceRepository.findById(id).orElse(**null**);

**if** (**null** != invoice ){

Customer customer = customerClient.getCustomer(invoice.getCustomerId()).getBody();

invoice.setCustomer(customer);

List<InvoiceItem> listItem=invoice.getItems().stream().map(invoiceItem -> {

Product product = productClient.getProduct(invoiceItem.getProductId()).getBody();

invoiceItem.setProduct(product);

**return** invoiceItem;

}).collect(Collectors.*toList*());

invoice.setItems(listItem);

}

**return** invoice ;

}

Agregando a las clases entity los campos faltantes:

**package** com.everis.entity;

**import** lombok.AllArgsConstructor;

**import** lombok.Builder;

**import** lombok.Data;

**import** java.util.ArrayList;

**import** java.util.Date;

**import** java.util.List;

**import** com.everis.model.Customer;

@Data

@AllArgsConstructor @Builder

**public** **class** Invoice {

**private** Long id;

**private** String numberInvoice;

**private** String description;

**private** Long customerId;

**private** Date createAt;

**private** List<InvoiceItem> items;

**private** String state;

**private** Customer customer;

**public** Invoice(){

items = **new** ArrayList<>();

}

**public** Invoice(**long** id, String numberInvoice, String description, **long** customerId, Date createAt, List<InvoiceItem> items, String state) {

**this**.id = id;

**this**.numberInvoice = numberInvoice;

**this**.description = description;

**this**.customerId = customerId;

**this**.createAt = createAt;

**this**.items =items;

**this**.state = state;

customer = **new** Customer();

items = **new** ArrayList<>();

}

}

**package** com.everis.entity;

**import** com.everis.model.Product;

**import** lombok.AllArgsConstructor;

**import** lombok.Builder;

**import** lombok.Data;

@Data

@AllArgsConstructor @Builder

**public** **class** InvoiceItem {

**private** Long id;

**private** Double quantity;

**private** Double price;

**private** Long productId;

**private** Double subTotal;

**private** Product product;

**public** InvoiceItem(Long id, Double quantity, Double price,Long productId){

**this**.id = id;

**this**.quantity = quantity;

**this**.price = price;

**this**.productId = productId;

product = **new** Product();

}

**public** Double getSubTotal(){

**if** (**this**.price >0 && **this**.quantity >0 ){

**return** **this**.quantity \* **this**.price;

}**else** {

**return** (**double**) 0;

}

}

**public** InvoiceItem(){

**this**.quantity=(**double**) 0;

**this**.price=(**double**) 0;

}

}

