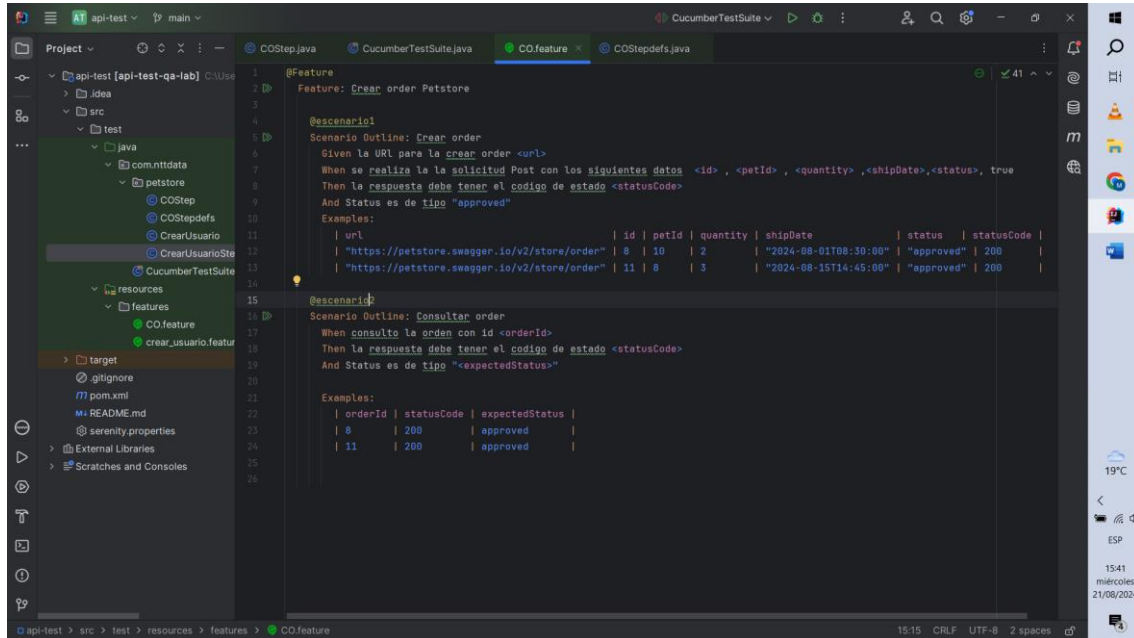


Debe crear 1 solo Feature que contenga los siguientes 2 Escenarios:

- Creación de Order: POST /store/order
- Consulta de Order: GET /store/order/{orderId}

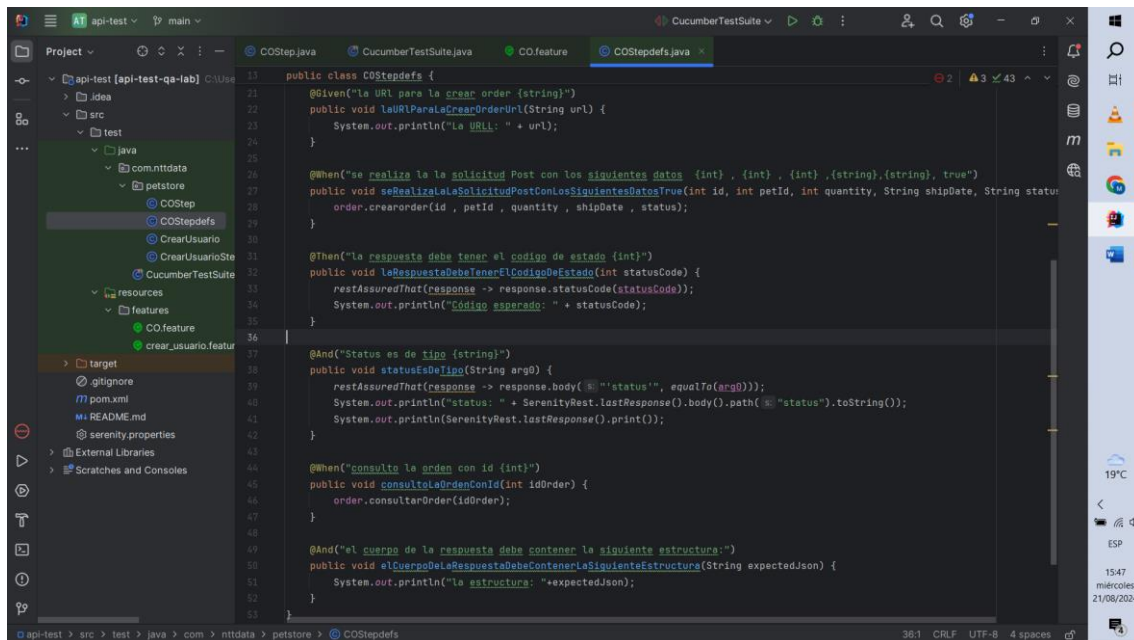
Gherkin de crear order para el post y get en "CO.feature"



The screenshot shows an IDE with the file 'CO.feature' open. The feature is titled 'Feature: Crear order Petstore'. It contains two scenarios: '@Escenario1' for creating an order and '@Escenario2' for consulting an order. Both scenarios include 'Given', 'When', and 'Then' steps, followed by an 'Examples' table.

```
1 Feature: Crear order Petstore
2
3
4 @Escenario1
5 Scenario Outline: Crear order
6   Given la URL para la crear order <url>
7   When se realiza la solicitud Post con los siguientes datos <id> , <petId> , <quantity> , <shipDate> , <status> , true
8   Then la respuesta debe tener el código de estado <statusCode>
9   And Status es de tipo "approved"
10
11 Examples:
12   | url | id | petId | quantity | shipDate | status | statusCode |
13   | "https://petstore.swagger.io/v2/store/order" | 8 | 10 | 2 | "2024-08-01T08:30:00" | "approved" | 200 |
14   | "https://petstore.swagger.io/v2/store/order" | 11 | 8 | 3 | "2024-08-15T14:45:00" | "approved" | 200 |
15
16 @Escenario2
17 Scenario Outline: Consultar order
18   When consulto la orden con id <orderId>
19   Then la respuesta debe tener el código de estado <statusCode>
20   And Status es de tipo "<expectedStatus>"
21
22 Examples:
23   | orderId | statusCode | expectedStatus |
24   | 8 | 200 | approved |
25   | 11 | 200 | approved |
```

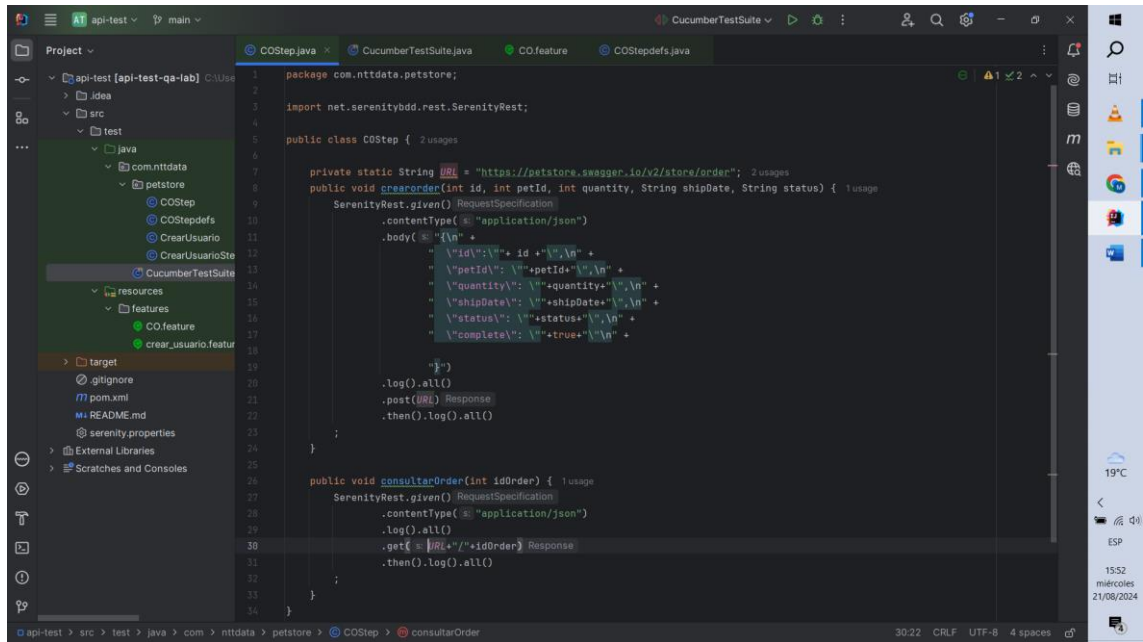
Creación del Step Definition en clase "COSTepdefs"



The screenshot shows the 'COSTepdefs' class in the IDE. It implements the step definitions for the scenarios in 'CO.feature'. The class includes methods for 'La URL para la crear order', 'se realiza la solicitud Post con los siguientes datos', 'la respuesta debe tener el código de estado', 'Status es de tipo', 'consulto la orden con id', and 'el cuerpo de la respuesta debe contener la siguiente estructura'.

```
11 public class COSTepdefs {
12
13   @Given("la URL para la crear order {string}")
14   public void laURLParaLaCrearOrderUrl(String url) {
15     System.out.println("la URL: " + url);
16   }
17
18   @When("se realiza la solicitud Post con los siguientes datos {int} , {int} , {int} , {string} , {string} , true")
19   public void seRealizaLaSolicitudPostConLosSiguientesDatosTrue(int id, int petId, int quantity, String shipDate, String status) {
20     order.crearOrder(id, petId, quantity, shipDate, status);
21   }
22
23   @Then("la respuesta debe tener el código de estado {int}")
24   public void laRespuestaDebeTenerElCodigoDeEstado(int statusCode) {
25     restAssuredThat(response -> response.statusCode(statusCode));
26     System.out.println("Código esperado: " + statusCode);
27   }
28
29   @And("Status es de tipo {string}")
30   public void statusEsDeTipo(String arg0) {
31     restAssuredThat(response -> response.body().path("status", equalTo(arg0)));
32     System.out.println("status: " + SerenityRest.lastResponse().body().path("status").toString());
33     System.out.println(SerenityRest.lastResponse().print());
34   }
35
36   @When("consulto la orden con id {int}")
37   public void consultoLaOrdenConId(int idOrder) {
38     order.consultarOrder(idOrder);
39   }
40
41   @And("el cuerpo de la respuesta debe contener la siguiente estructura:")
42   public void elCuerpoDeLaRespuestaDebeContenerLaSiguienteEstructura(String expectedJson) {
43     System.out.println("la estructura: " + expectedJson);
44   }
45 }
```

Creación del Step en clase "COStep"



```
package com.nttdata.petstore;

import net.serenitybdd.rest.SerenityRest;

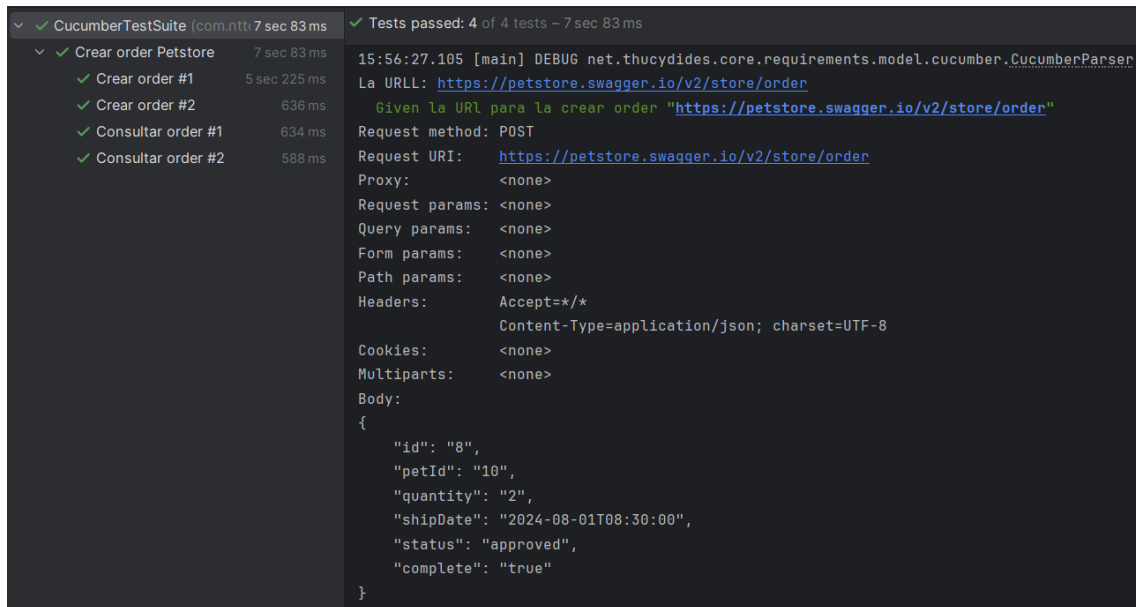
public class COStep {

    private static String URL = "https://petstore.swagger.io/v2/store/order";

    public void createOrder(int id, int petId, int quantity, String shipDate, String status) {
        SerenityRest.given() RequestSpecification
            .contentType("application/json")
            .body("{\"id\":\"" + id + "\",\n" +
                "\"petId\":\"" + petId + "\",\n" +
                "\"quantity\":\"" + quantity + "\",\n" +
                "\"shipDate\":\"" + shipDate + "\",\n" +
                "\"status\":\"" + status + "\",\n" +
                "\"complete\":\"" + true + "\""}")
            .log().all()
            .post(URL) Response
            .then().log().all()
        ;
    }

    public void consultarOrder(int idOrder) {
        SerenityRest.given() RequestSpecification
            .contentType("application/json")
            .log().all()
            .get(URL + "/" + idOrder) Response
            .then().log().all()
        ;
    }
}
```

Ejecución del escenario 1 y escenario 2



```
✓ CucumberTestSuite (com.nttd) 7 sec 83 ms ✓ Tests passed: 4 of 4 tests - 7 sec 83 ms
  ✓ Crear order Petstore 7 sec 83 ms
    ✓ Crear order #1 5 sec 225 ms
    ✓ Crear order #2 636 ms
    ✓ Consultar order #1 634 ms
    ✓ Consultar order #2 588 ms

15:56:27.105 [main] DEBUG net.thucydides.core.requirements.model.cucumber.CucumberParser
La URL: https://petstore.swagger.io/v2/store/order
Given la URL para la crear order "https://petstore.swagger.io/v2/store/order"
Request method: POST
Request URI: https://petstore.swagger.io/v2/store/order
Proxy: <none>
Request params: <none>
Query params: <none>
Form params: <none>
Path params: <none>
Headers: Accept=*/*
Content-Type=application/json; charset=UTF-8
Cookies: <none>
Multiparts: <none>
Body:
{
  "id": "8",
  "petId": "10",
  "quantity": "2",
  "shipDate": "2024-08-01T08:30:00",
  "status": "approved",
  "complete": "true"
}
```

The screenshot shows a terminal window with two main sections. The top section displays test results for a CucumberTestSuite, showing four tests passed in 7 seconds and 83 milliseconds. The tests are: 'Crear order Petstore' (7 sec 83 ms), 'Crear order #1' (5 sec 225 ms), 'Crear order #2' (636 ms), and 'Consultar order #1' (634 ms). The bottom section shows the output of a REST client, including connection details (keep-alive, access-control-allow-origin: *, access-control-allow-methods: GET, POST, DELETE, PUT, access-control-allow-headers: Content-Type, api_key, Authorization) and a JSON response for a POST request. The response is: { "id": 8, "petId": 10, "quantity": 2, "shipDate": "2024-08-01T08:30:00.000+0000", "status": "approved", "complete": true }. Below the JSON response, there is a comment in Spanish: 'When se realiza la la solicitud Post con los siguientes datos 8 , 10 , 2 , "2024-08-01T08:30:00", "approved", true'. This is followed by another comment: 'Then la respuesta debe tener el codigo de estado 200'. The status is confirmed as 'approved'. The bottom part of the terminal shows a log message: '15:56:31.993 [main] INFO -'.

```

CucumberTestSuite (com.nnti) 7 sec 83 ms
  ✓ Crear order Petstore 7 sec 83 ms
  ✓ Crear order #1 5 sec 225 ms
  ✓ Crear order #2 636 ms
  ✓ Consultar order #1 634 ms
  ✓ Consultar order #2 588 ms

Tests passed: 4 of 4 tests - 7 sec 83 ms

Connection: keep-alive
Access-Control-Allow-Origin: *
Access-Control-Allow-Methods: GET, POST, DELETE, PUT
Access-Control-Allow-Headers: Content-Type, api_key, Authorization
Server: Jetty(9.2.9.v20150224)

{
  "id": 8,
  "petId": 10,
  "quantity": 2,
  "shipDate": "2024-08-01T08:30:00.000+0000",
  "status": "approved",
  "complete": true
}

When se realiza la la solicitud Post con los siguientes datos 8 , 10 , 2 , "2024-08-01T08:30:00", "approved", true
Then la respuesta debe tener el codigo de estado 200
status: approved
{"id":8,"petId":10,"quantity":2,"shipDate":"2024-08-01T08:30:00.000+0000","status":"approved","complete":true}
{"id":8,"petId":10,"quantity":2,"shipDate":"2024-08-01T08:30:00.000+0000","status":"approved","complete":true}
And Status es de tipo "approved"
15:56:31.993 [main] INFO -
  
```