**Test notes**

In the following document I am going to explain what I thought when exploring the application, the different bugs that I found and the different doubts that surfaced while I explored the API:

**Doubts:**

* While exploring the application, the first doubts that I had were related to the backend logic of the endpoint, because I had the contracts of the different services but I didn’t have the expected result of some test nor the input that I should use to test.
  + ¿In the order model, what does the quantity refer to? ¿Should it be validated?
  + ¿The shipment date should be consistent with the order status?
* To face this doubts I decided to assume some things regarding the internal logic of the application. For example:
  + There should be an internal validation regarding the quantity of a certain order, assuming that the quantity corresponds to the number of pets.
  + If the store order service is exposed to users so that they can create purchase orders, users shouldn’t be able to create orders with a shipDate that has already passed. In addition, the shipDate and the status of the order should be consistent. For instance, the system should not allow the creation of orders in which the shipDate is in the future and the status is “delivered”.
  + There should be a relation between orders and pets, so the system should reject the creation of orders were pets are not registered.
* My test target was the store server. However, the functionality for getting the inventory depends on the status of the pets that are registered in the system (This means that the service is affected by the Pet service). While exploring the rest API functionalities for adding and modifying pets I found some inconsistency:
  + The functionality for adding and updating pets specifies that the status is a string. On the other hand, the find by status functionality suggests querying by using three different values (available, pending, sold). Additionally, the inventory functionality returns multiple values when grouping by status. So, my question regarding this situation is: The application really requires managing multiples growing values for the status field, or the status field should be bounded to the three values mentioned before?
* The pet model has a field called complete that is a boolean value. What does this field refer to?
* Regarding the business, who’s the end user of the application? In other words, is the application going to be used by the customer or by an employee?
* Why isn’t there a service to update the order information?
  + Can the status of an order change over time?
  + Is the same post service used to update the information?
    - If yes, this can cause a bottleneck, as many requests may go to the same url.
* Do orders follow a lifecycle? Should they always start in pending and then move to pending and delivered?

**Bugs found**

* As I said before, I assumed that the system should validate the quantity that is send in a post order request. Regarding this I found two bugs:
  + The system does not appear to validate the quantity of an order, so any purchase order with any given value (that isn’t too big) can be created.
  + When posting orders with a quantity value that is too big, the system returns error code 500 (Something bad happened). This means that invalid input is not filtered out.
* Orders from a past time should not be created but are. (Based on the assumption that I made before).
* Orders can be created even when the petId does not belong to any pet that is registered in the system. This can cause data inconsistencies. Invalid data should be validated.
* Orders can be created with negative values in quatity. This can cause data inconsistencies. Invalid data should be validated.
* According to the documentation, if any order request (get, post, delete) is provided with an invalid id, the system should return error code 400(Invalid id provided or invalid order). However:
  + When posting order with an invalid id, the returned value is error 500.
  + When sending a get or a delete request with an invalid id, the returned value is error 405.
* Orders can be created with inconsistent logical values: The system allows the creation of orders with a future shipment date and a delivered status. This should be validated as an order cannot be in a delivered status if the shipment date is in the future.
* Regarding the managing of the different possible values for pet status, I think the approach should be changed, as this makes pet status tracking more difficult. In addition, I think that the system should not allow the creation of purchase orders for pets that are already sold, changing the status approach should simplify this validation. Currently, the system allows the creation of orders with pets that are already sold.