Лаба 5. Тестирование API.

Работа с API

https://petstore.swagger.io
В Postman использовать https://petstore.swagger.io/v2
Перед выполнением задания необходимо ознакомиться с API. Все запросы на создание новых сущностей и изменение существующих – фиктивные. Однако их можно вызывать.

1.Работа с Postman'ом
а) Создайте отдельную коллекцию с названием **Shop**
б_0) Добавьте в коллекцию (из предыдущего пункта) запрос для добавления животного с id, например, 11 (можете свой корректный id придумать)
б) Добавьте в коллекцию запрос для получения животного с id своим, из пункта б_0
в) Добавьте в коллекцию GET запрос по недопустимому id, добавить в него тесты, проверяющие наличие ошибки (status code и прочее) в ответе
г) Добавьте в коллекцию GET запрос по слишком большому id, добавить в него тесты, проверяющие наличие ошибки (status code и прочее) в ответе (возможно совпадут с тестами из пункта в)
д) Добавьте метод для изменения информации о домашнем питомце целиком (PUT). Проверить, что ответ на запрос возвращает новые данные
е) Допишите необходимые тесты (в разделы pre-tests, tests). Какие ещё необходимы, см. ниже
ё) Загрузите коллекцию в classroom (в Postman см. "Export")
ж) Выдайте мне ссылку на коллекцию
2. Создайте **maven** Java-проект и напишите реализацию запросов, которые выполняют те же действия, что и в части 1(Postman).
а) В папку tests добавьте тесты на запросы GET, POST, PUT, DELETE.
б) Залейте проект в git-репозиторий (gitlab, bitbucket, github) в отдельной ветке (не master), добавьте меня (username: michaelbakl либо baklykovmichael@gmail.com) с правом комментирования и создайте pull request из текущей ветки в master
в) Сообщите (нажмите сдать задание)

Необходимо ещё протестировать следующие состояния:
1. Корректное выполнение запроса POST (который содержит все необходимые поля) должно возвращать ответ, в котором успешный код (200/201/202), нет упоминания об ошибке, должно быть тело и должно быть формата json.
2. Корректное выполнение запроса GET должно возвращать ответ, в котором, кроме требований к запросу POST (из п. 1), тело должно содержать еще и текущие имеющиеся данные (исходя из API). Здесь в контексте данной задачи достаточно проверить, что какие-то поля "non null" (т.е. что в них записаны данные)