

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Бэк-энд разработка

Отчет

Домашняя работа №4

Выполнила:

Платонова Александра

Группа К3339

Проверил:

Добряков Д. И.

Санкт-Петербург

2025 г.

Задача

- реализовать тестирование API средствами Postman;
- написать тесты внутри Postman.

Ход работы

Для каждой сущности, созданной в лабораторной работе №1, были написаны тесты внутри Postman. Далее приведены примеры некоторых из них.

При создании общежития производится проверка на наличие указанного адреса, а также добавлен негативный тест, который проверяет отображение ошибки в случае дублирования названия общежития.

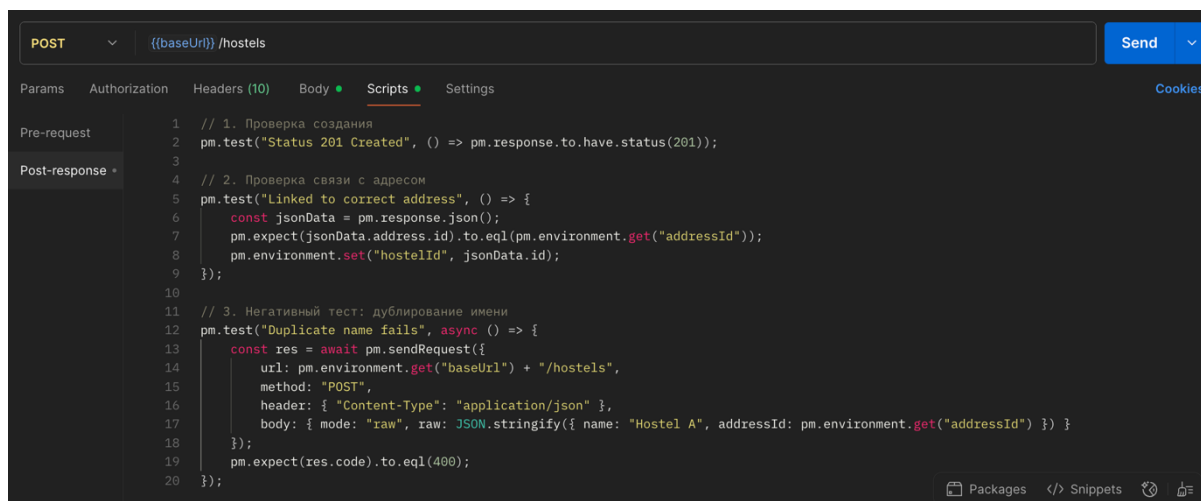


Рисунок 1 - Тесты при создании общежития

Для создания комнаты написаны следующие тесты: проверка указанной площади (должна быть больше минимальной), проверка существования указанного id общежития и возвращение ошибки, если такого id не существует.

```
1 // 1. Проверка минимальной площади
2 pm.test("Area validation", () => {
3     const jsonData = pm.response.json();
4     pm.expect(jsonData.area).to.be.above(0);
5     pm.environment.set("roomId", jsonData.id);
6 });
7
8 // 2. Проверка связи с общежитием
9 pm.test("Linked to correct hostel", () => {
10    const jsonData = pm.response.json();
11    pm.expect(jsonData.hostel.id).to.eql(pm.environment.get("hostelId"));
12 });
13
14 // 3. Негативный тест: неверный hostelId
15 pm.test("Invalid hostelId fails", async () => {
16    const res = await pm.sendRequest({
17        url: pm.environment.get("baseUrl") + "/rooms",
18        method: "POST",
19        header: { "Content-Type": "application/json" },
20        body: { mode: "raw", raw: JSON.stringify({ hostelId: 99999, beds: 2 }) }
21    });
22    pm.expect(res.code).to.eql(404);
23 });
```

Рисунок 2 - Тесты при создании комнаты

При создании договора о заселении производится проверка на корректный формат даты, а также на существование указанных id (проживающего и комнаты).

```
1 // 1. Проверка даты
2 pm.test("Future date fails", async () => {
3     const res = await pm.sendRequest({
4         url: pm.environment.get("baseUrl") + "/check-ins",
5         method: "POST",
6         header: { "Content-Type": "application/json" },
7         body: { mode: "raw", raw: JSON.stringify({ date_from: "2030-01-01", residentId: pm.environment.get("residentId"), roomId: pm.environment.get("roomId") }) }
8     });
9     pm.expect(res.code).to.eql(400);
10 });
11
12 // 2. Проверка связей
13 pm.test("Linked to resident and room", () => {
14     const jsonData = pm.response.json();
15     pm.expect(jsonData.resident.id).to.eql(pm.environment.get("residentId"));
16     pm.expect(jsonData.room.id).to.eql(pm.environment.get("roomId"));
17     pm.environment.set("checkInId", jsonData.id);
18 });
```

Рисунок 3 - Тесты при заселении

Для формирования ссылки на оплату были реализованы следующие тесты: проверка на допустимые статусы оплаты, проверка на отрицательную сумму.

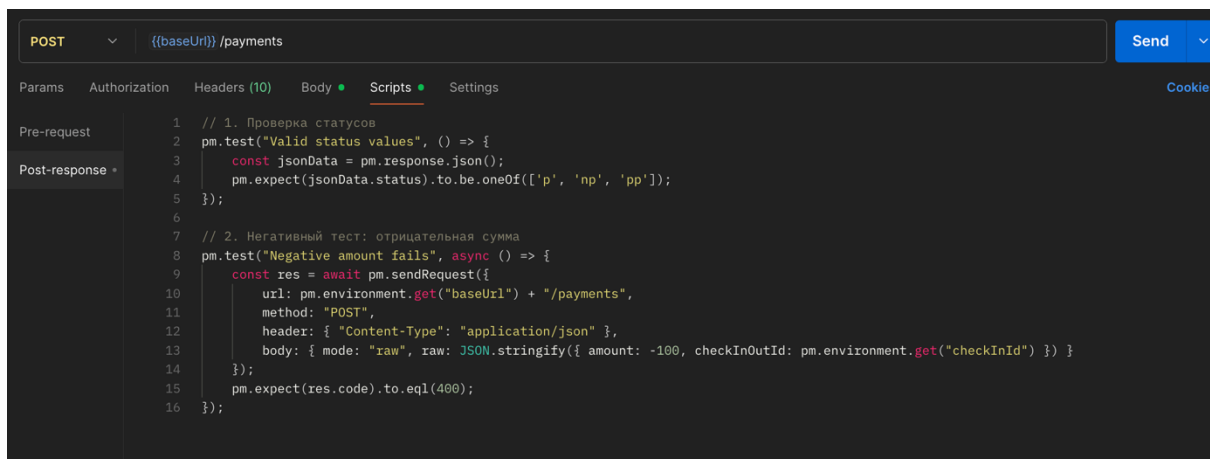


Рисунок 4 - Тесты при оплате

Вывод

В ходе выполнения домашнего задания были реализованы тесты для всех ранее объявленных сущностей средствами Postman.