Оглавление

[**Product** 2](#_Toc42531621)

[**Получение данных** 2](#_Toc42531622)

[**Получение всех данных** 2](#_Toc42531623)

[**Получение данных по id товара** 2](#_Toc42531624)

[**Создание товара** 4](#_Toc42531625)

[**Редактирование товара** 5](#_Toc42531626)

[**Удаление товара** 7](#_Toc42531627)

[**Order** 8](#_Toc42531628)

[**Получение данных** 8](#_Toc42531629)

[**Получение всех заказов** 8](#_Toc42531630)

[**Получение всех заказов по имени покупателя** 9](#_Toc42531631)

[**Получение данных о заказе по id** 10](#_Toc42531632)

[**Добавление заказа** 11](#_Toc42531633)

[**Редактирование заказа** 12](#_Toc42531634)

[**Удаление заказа** 13](#_Toc42531635)

**Product** – класс, характеризующий товар на сервере

## **Получение данных**

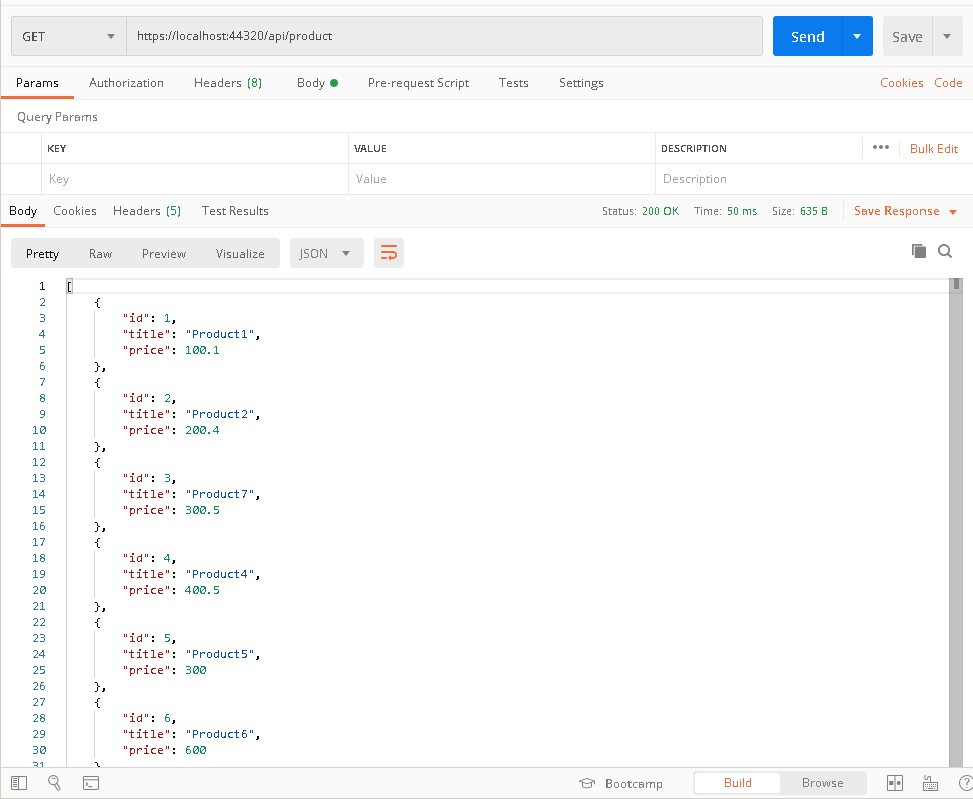
### **Получение всех данных**

Для получения всех товаров используется метод GET по адресу “api/product”

Пример запроса:



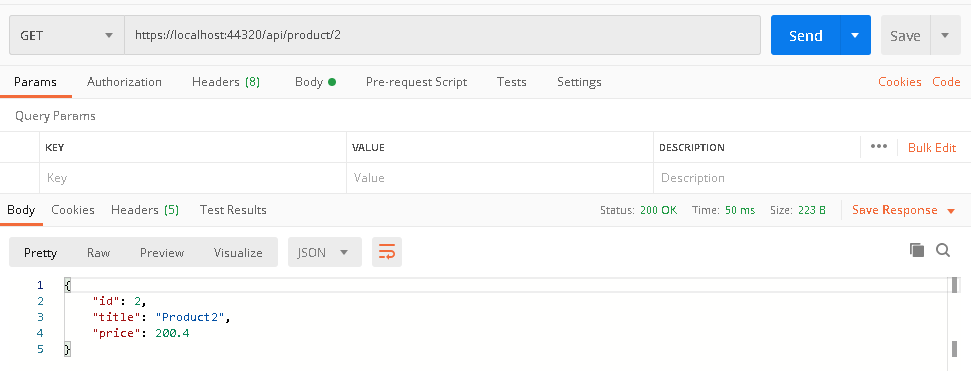
Пример ответа сервера:



### **Получение данных по id товара**

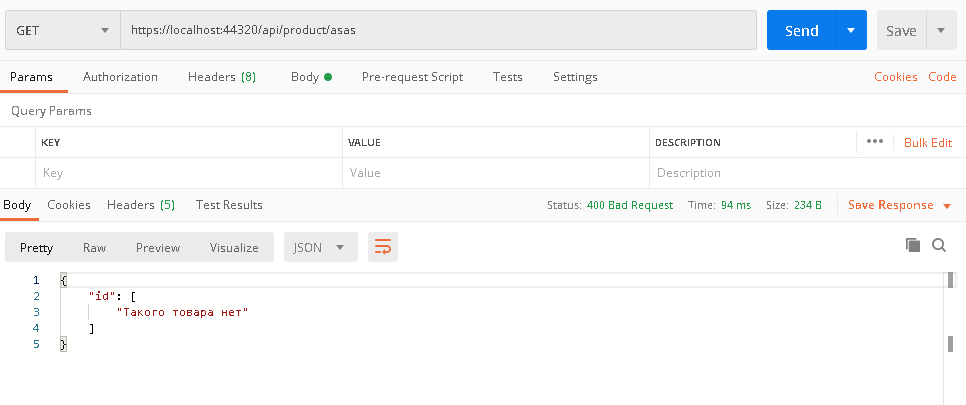
Для получения информации о конкретном товаре используется метод GET по адресу “api/product/{id}”

Пример такого запроса и ответ на него со стороны сервера:



На вход, как id ожидается целочисленная переменная. В случае некорректного ввода или ввода id, которого нет, сервер отправит ответ со статусной ошибкой 400 и комментарием проблемы.

Пример:



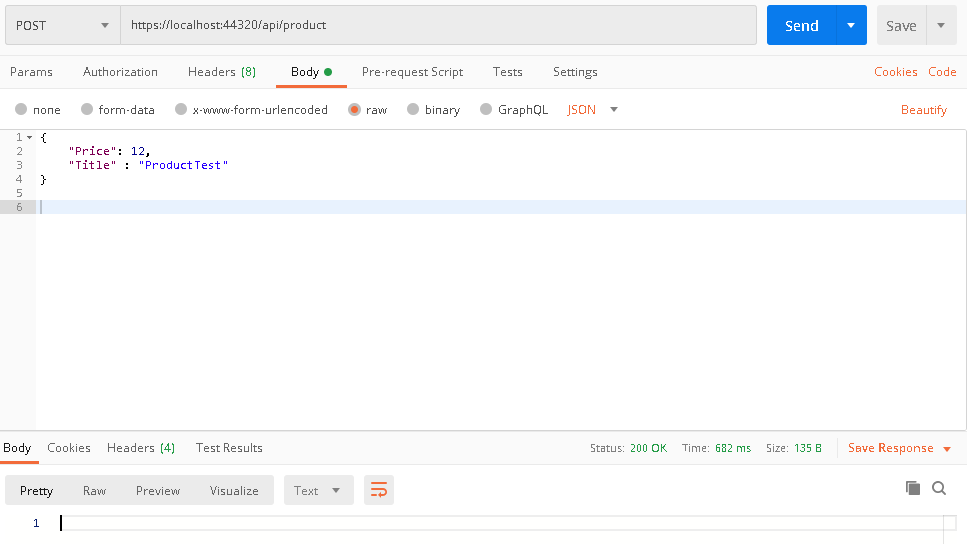
## **Создание товара**

Для создания товара используется метод POST по адресу “/api/product”

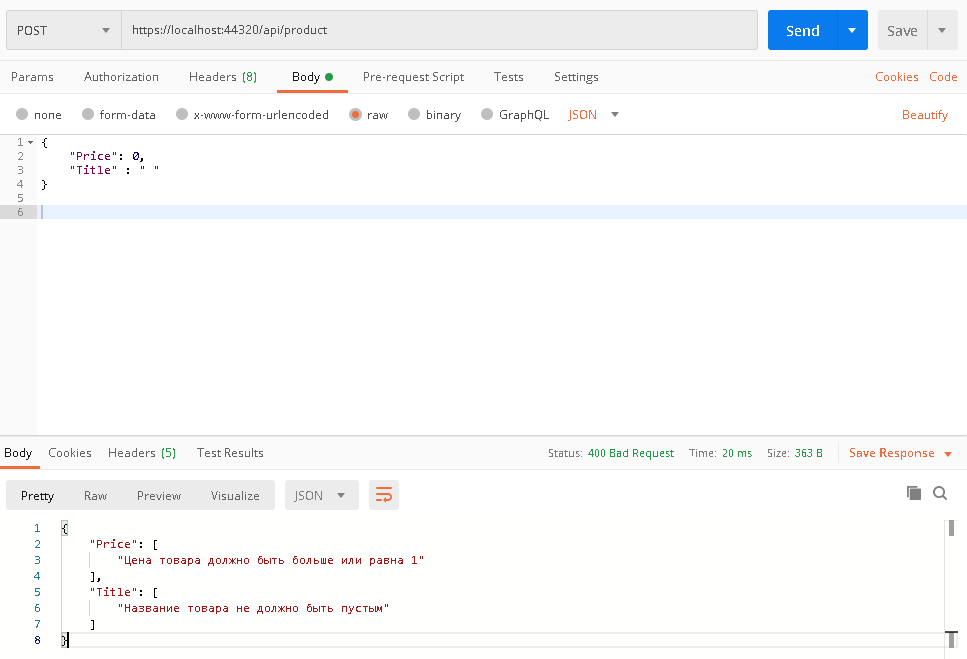
Параметры в body:

* “Title” – название продукта. Название не может состоять из пробелов или быть пустым.
* “Price” – цена данного продукта. Цена является float типом, и она должна быть больше 0.

Пример запроса для создания продукта:



В случае ошибки, ответ сервера будет выглядеть так



Если запрос прошел успешно, то будет возвращен ответ с кодом 200.

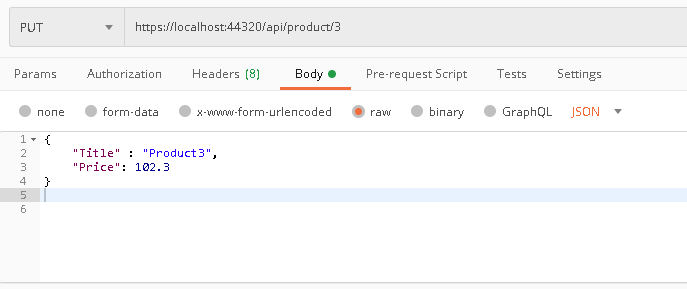
## **Редактирование товара**

Для редактирования товара используется метод PUT по адресу “/api/product/{id}”

Параметры в body:

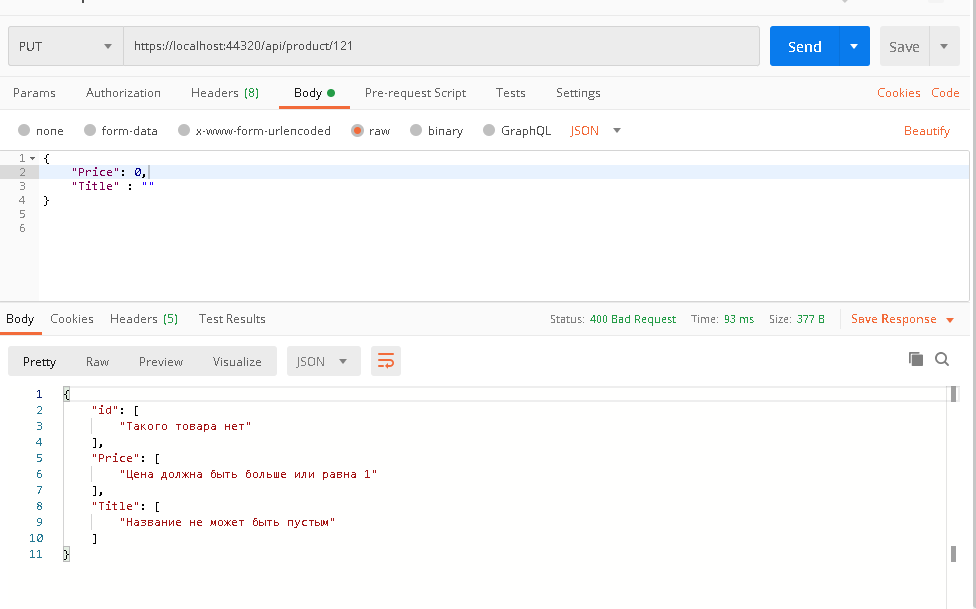
* “Title” – название продукта. Название не может состоять из пробелов или быть пустым.
* “Price” – цена данного продукта. Цена является float типом, и она должна быть больше 0.

Пример запроса для редактирования продукта:



При успешном редактировании сервер вернет статусный код 200.

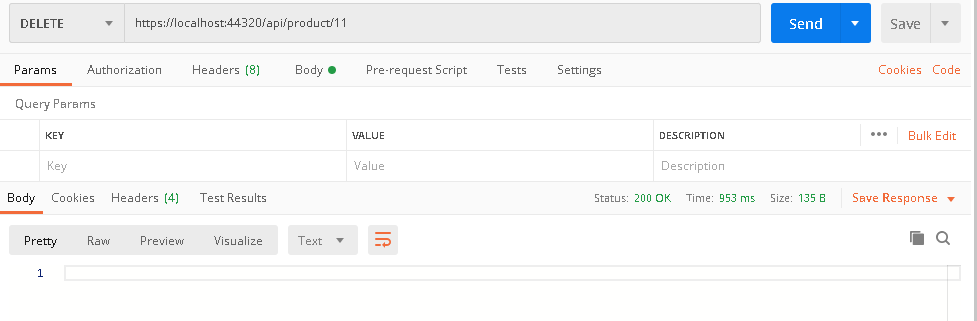
Если произошла ошибка, сервер вернет статусный код 400 и комментарии к ошибкам. Пример:



## **Удаление товара**

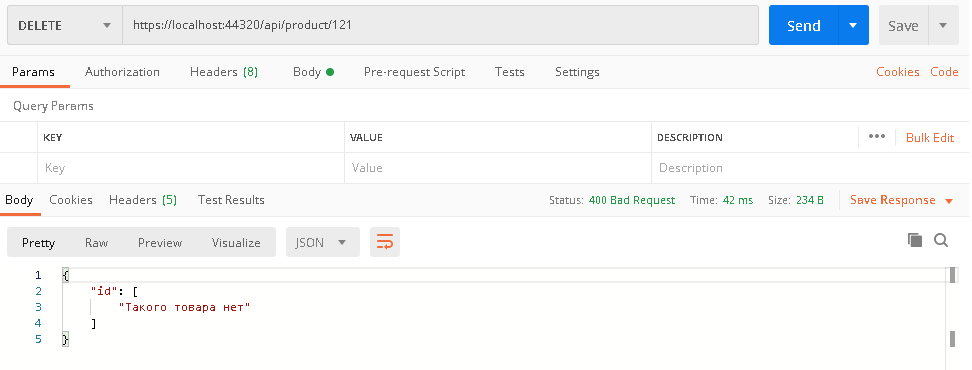
Для удаления товара используется метод DELETE по адресу “/api/product/{id}”

Пример запроса для удаления продукта:



При успешном удалении сервер вернет статусный код 200.

Если произошла ошибка, сервер вернет статусный код 400 и комментарии к ошибкам. Пример:



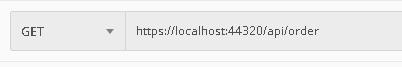
**Order** - класс, характеризующий заказ товара на сервере

## **Получение данных**

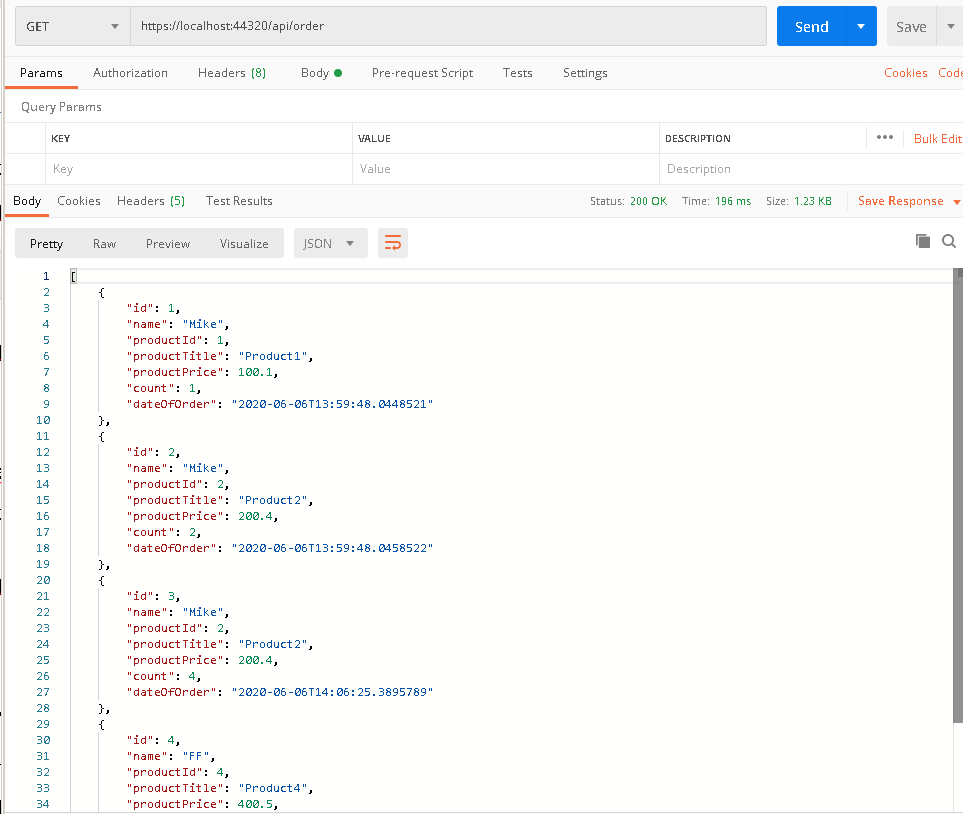
### **Получение всех заказов**

Для получения всех заказов используется метод GET по адресу “api/order”

Пример запроса:



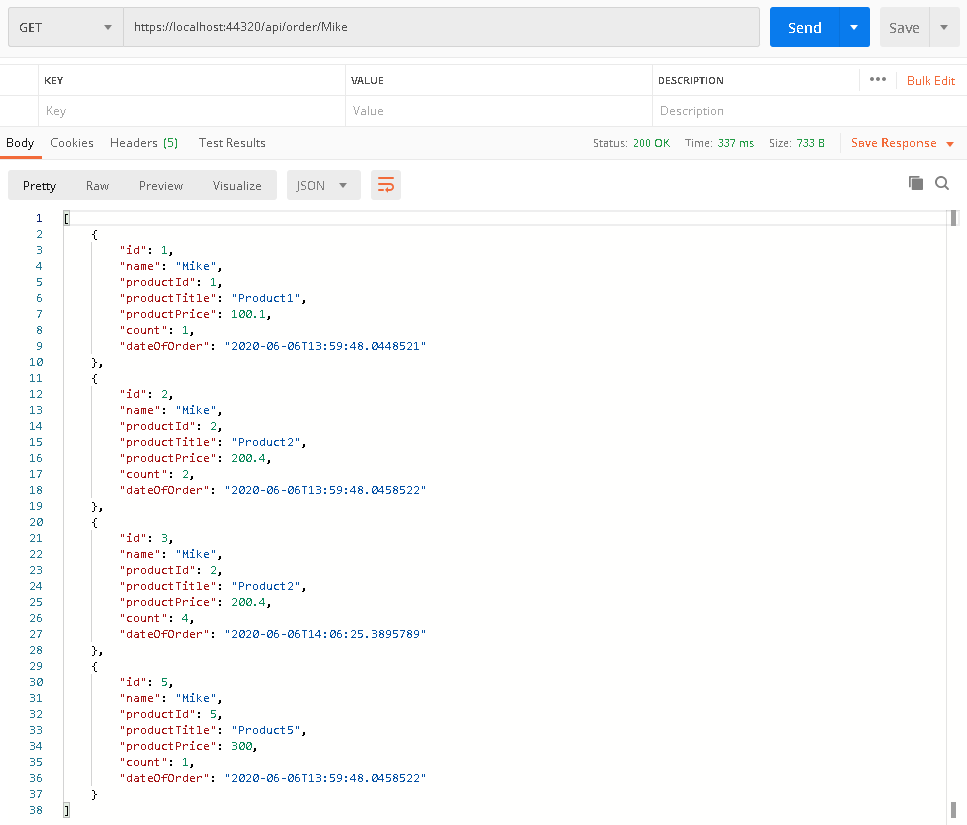
Пример ответа сервера:



### **Получение всех заказов по имени покупателя**

Для получения информации о заказах конкретного пользователя используется метод GET по адресу “api/product/{name}”

Пример такого запроса и ответ:

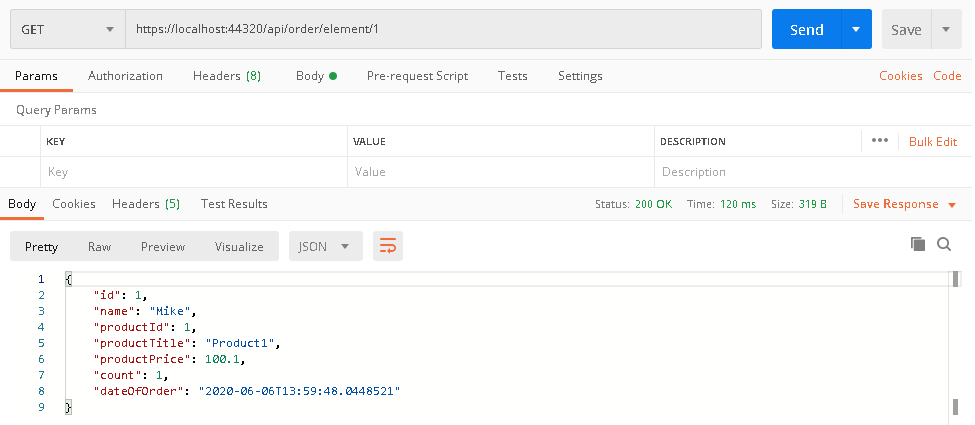


На вход, как name ожидается стока переменная. В случае если такого пользователя нет, сервер отправит ответ со статусной ошибкой 404.

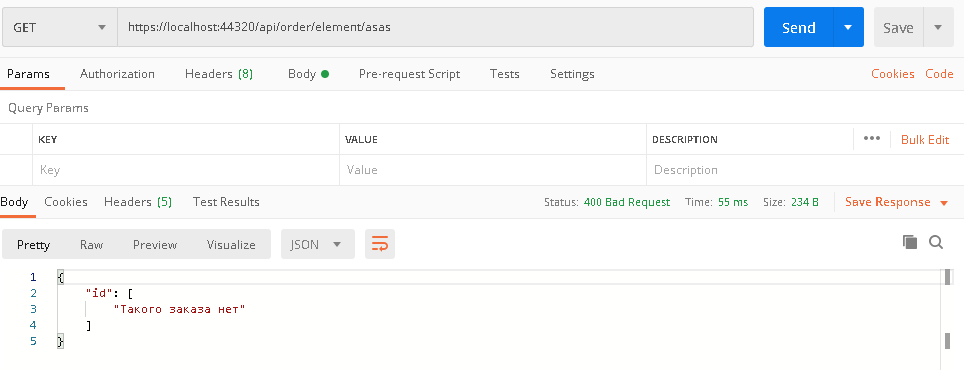
### **Получение данных о заказе по id**

Для получения информации о заказах по id используется метод GET по адресу “api/product/element/{id}”

Пример такого запроса и ответ на него со стороны сервера:



Если произошла ошибка, сервер вернет статусный код 400 и комментарии к ошибкам. Пример:



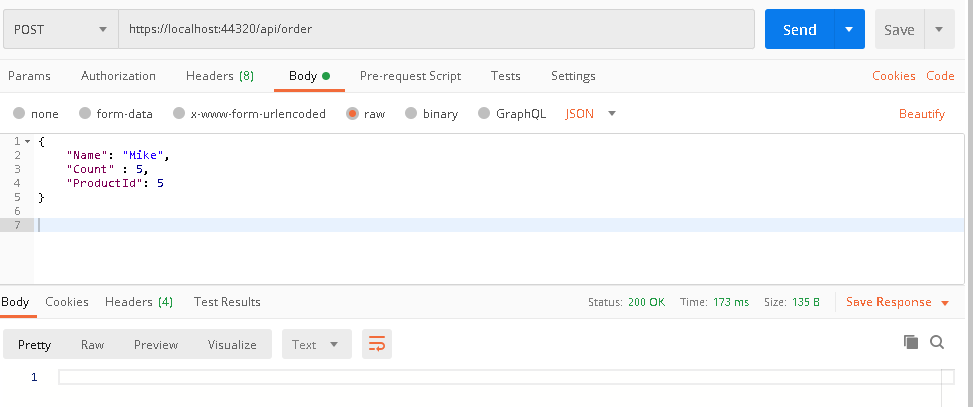
## **Добавление заказа**

Для создания заказа используется метод POST по адресу “/api/order”

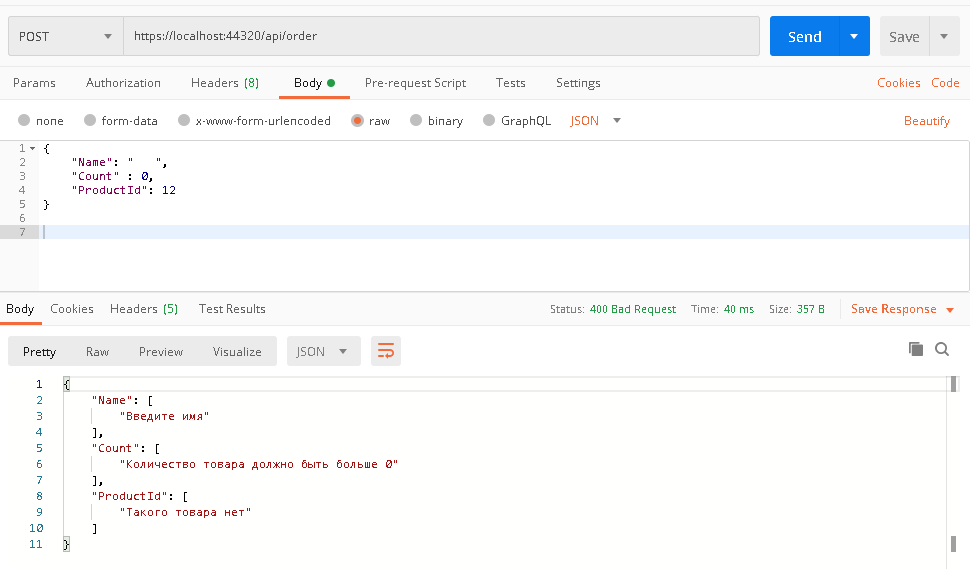
Параметры в body:

* “Name” – имя или псевдоним покупателя. Поле должно быть не пустым.
* “ProductId” – идентификатор товара. Если идентификатор некорректный, будет выдана ошибка.
* “Count” – количество товара к покупке. Количество должно быть больше или равно 1.

Пример запроса для создания продукта:



В случае ошибки, сервер ответит статусным кодом 400 и комментарии к ошибкам:



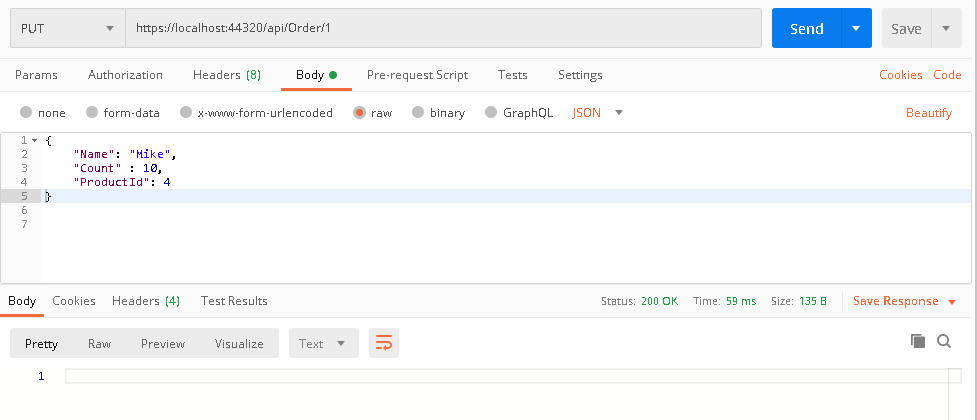
## **Редактирование заказа**

Для редактирования товара используется метод PUT по адресу “/api/order/{id}”

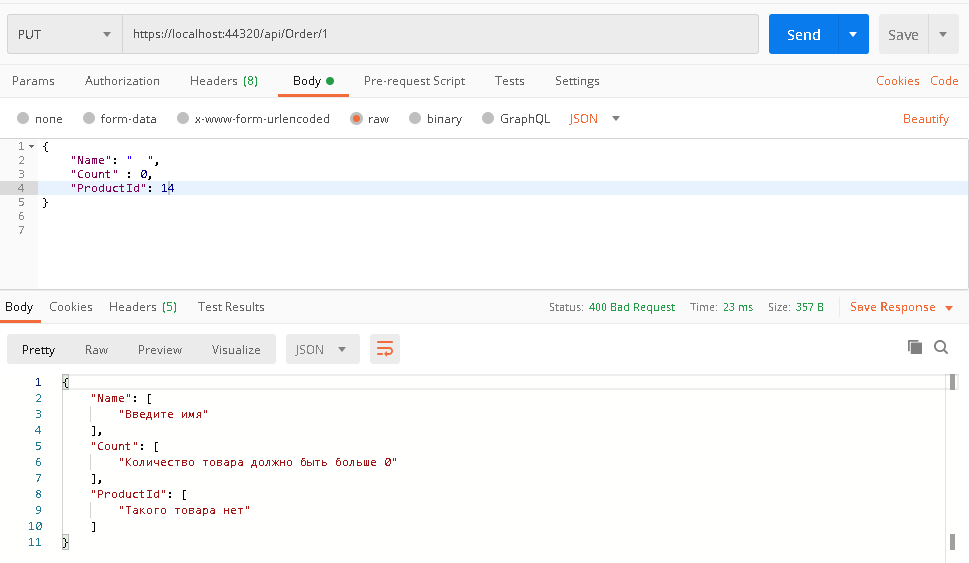
Параметры в body:

* “Name” – имя или псевдоним покупателя. Поле должно быть не пустым.
* “ProductId” – идентификатор товара. Если идентификатор некорректный, будет выдана ошибка.
* “Count” – количество товара к покупке. Количество должно быть больше или равно 1.

Пример запроса для редактирования продукта:



В случае ошибки, сервер ответит статусным кодом 400 и комментарии к ошибкам:



## **Удаление заказа**

Для удаления заказа используется метод DELETE по адресу “/api/order/{id}”

Пример запроса для удаления заказа:



При успешном удалении сервер вернет статусный код 200.

Если произошла ошибка, сервер вернет статусный код 400 и комментарии к ошибкам. Пример:

