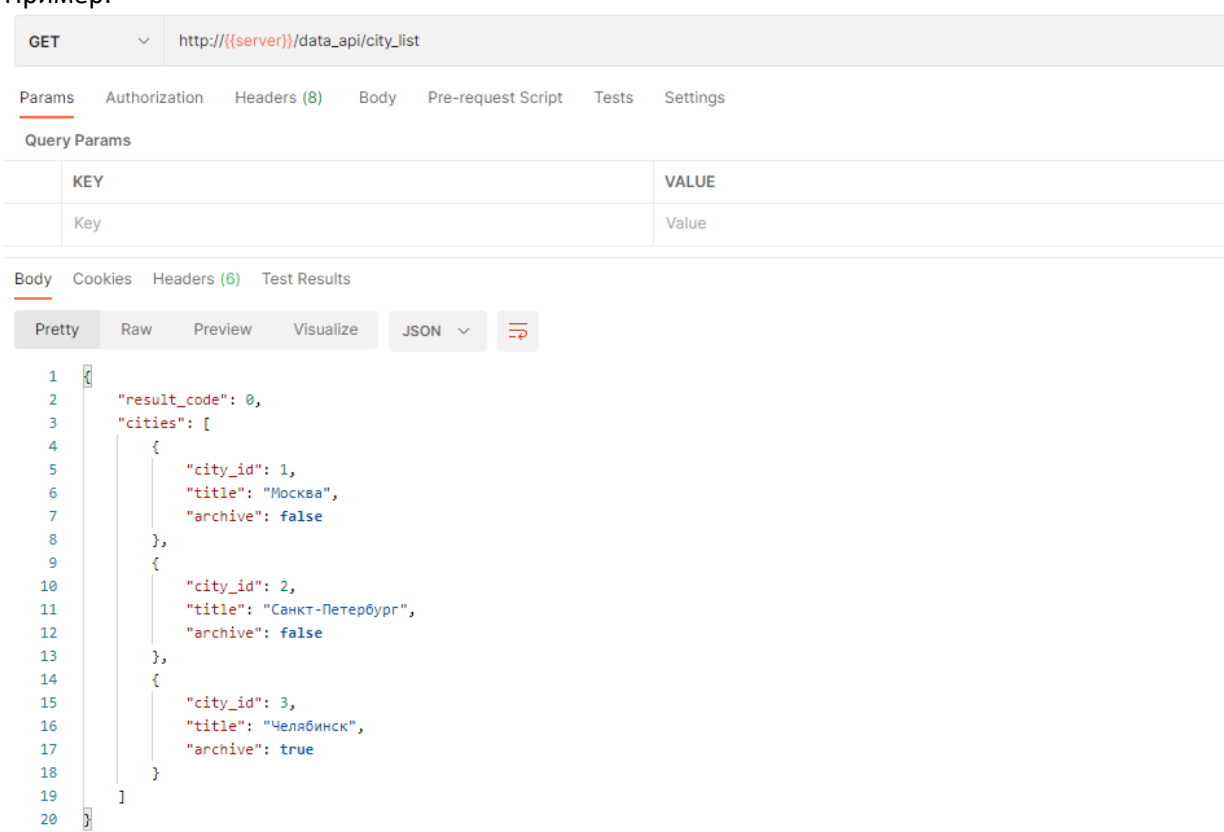


## API-протокол для работы с программой WS.Автопрокат

№	ОПИСАНИЕ
1	<p><b>Запрос на получение списка городов</b></p> <p>Запрос: GET-запрос <i>city_list</i> без параметров</p> <p>Ответ: cities – массив городов</p> <pre>{   city_id – идентификатор города (тип данных – целое число),   title – название (тип данных – текст),   archive – архивность (тип данных – логическое значение. True для отправленных в архив городов) }</pre> <p>В ответе возвращаются все города.</p> <p>Пример:</p>  <pre>{   "result_code": 0,   "cities": [     {       "city_id": 1,       "title": "Москва",       "archive": false     },     {       "city_id": 2,       "title": "Санкт-Петербург",       "archive": false     },     {       "city_id": 3,       "title": "Челябинск",       "archive": true     }   ] }</pre>

## 2 Запрос на получение списка мест выдачи и возврата

Запрос:

GET-запрос *place\_list* без параметров

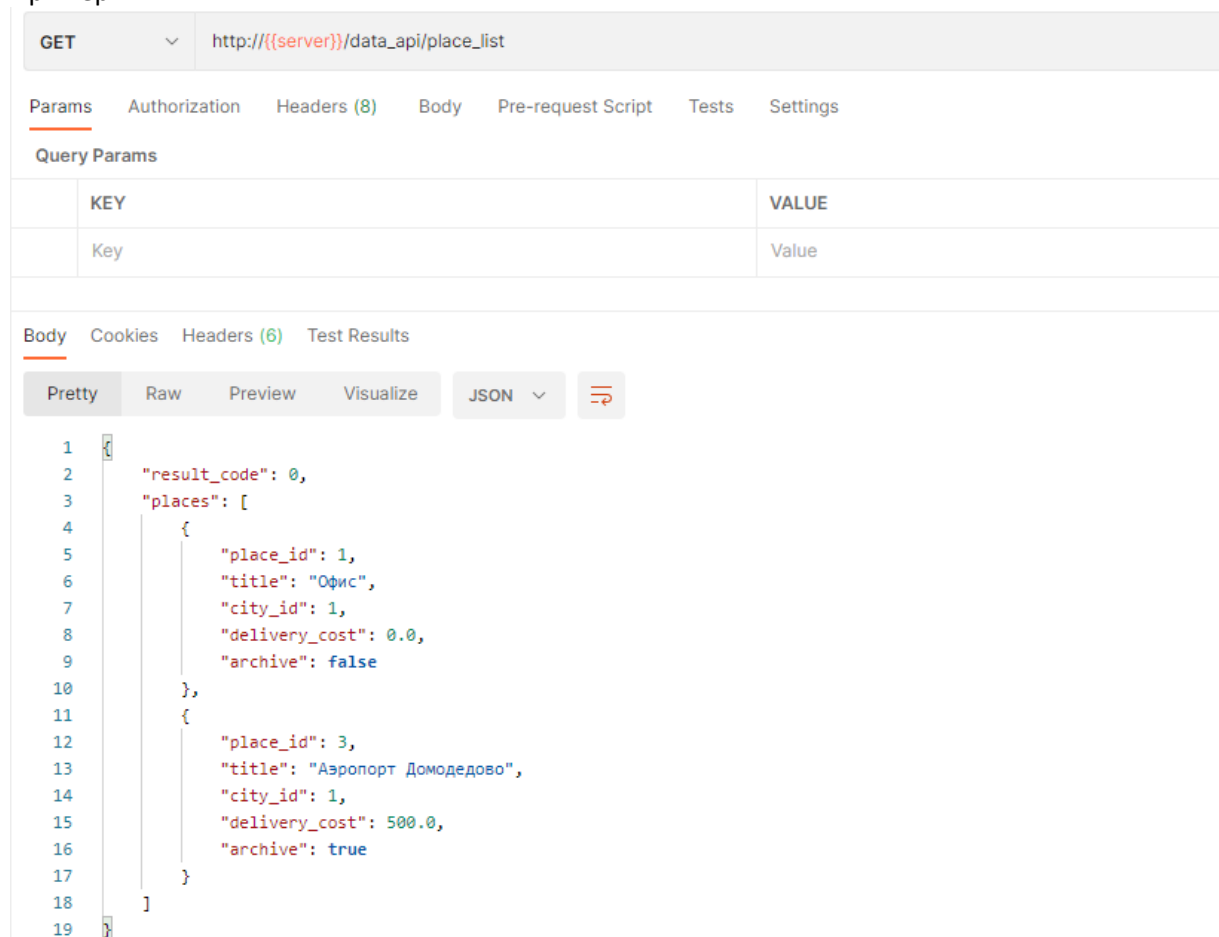
Ответ:

places – массив мест выдачи и возврата

```
{
  place_id – идентификатор города (тип данных – целое число),
  title – название (тип данных – текст),
  city_id – идентификатор города (тип данных – целое число),
  delivery_cost – стоимость доставки/получения автомобиля (тип данных – число),
  archive – архивность (тип данных – логическое значение. True для отправленных в архив мест
выдачи и возврата)
}
```

В ответе возвращаются все места выдачи и возврата.

Пример:



The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** http://{{server}}/data\_api/place\_list
- Params:** Authorization, Headers (8), Body, Pre-request Script, Tests, Settings
- Query Params:** A table with two columns: KEY and VALUE. The first row contains 'Key' and 'Value'.
- Body:** Cookies, Headers (6), Test Results
- Response Format:** Pretty, Raw, Preview, Visualize, JSON (selected)
- Response Body (JSON):**

```
{
  "result_code": 0,
  "places": [
    {
      "place_id": 1,
      "title": "Офис",
      "city_id": 1,
      "delivery_cost": 0.0,
      "archive": false
    },
    {
      "place_id": 3,
      "title": "Аэропорт Домодедово",
      "city_id": 1,
      "delivery_cost": 500.0,
      "archive": true
    }
  ]
}
```

### 3 Запрос на получение списка свободных автомобилей

Запрос:

GET-запрос *car\_free\_list*, параметры передаются через строку запроса:

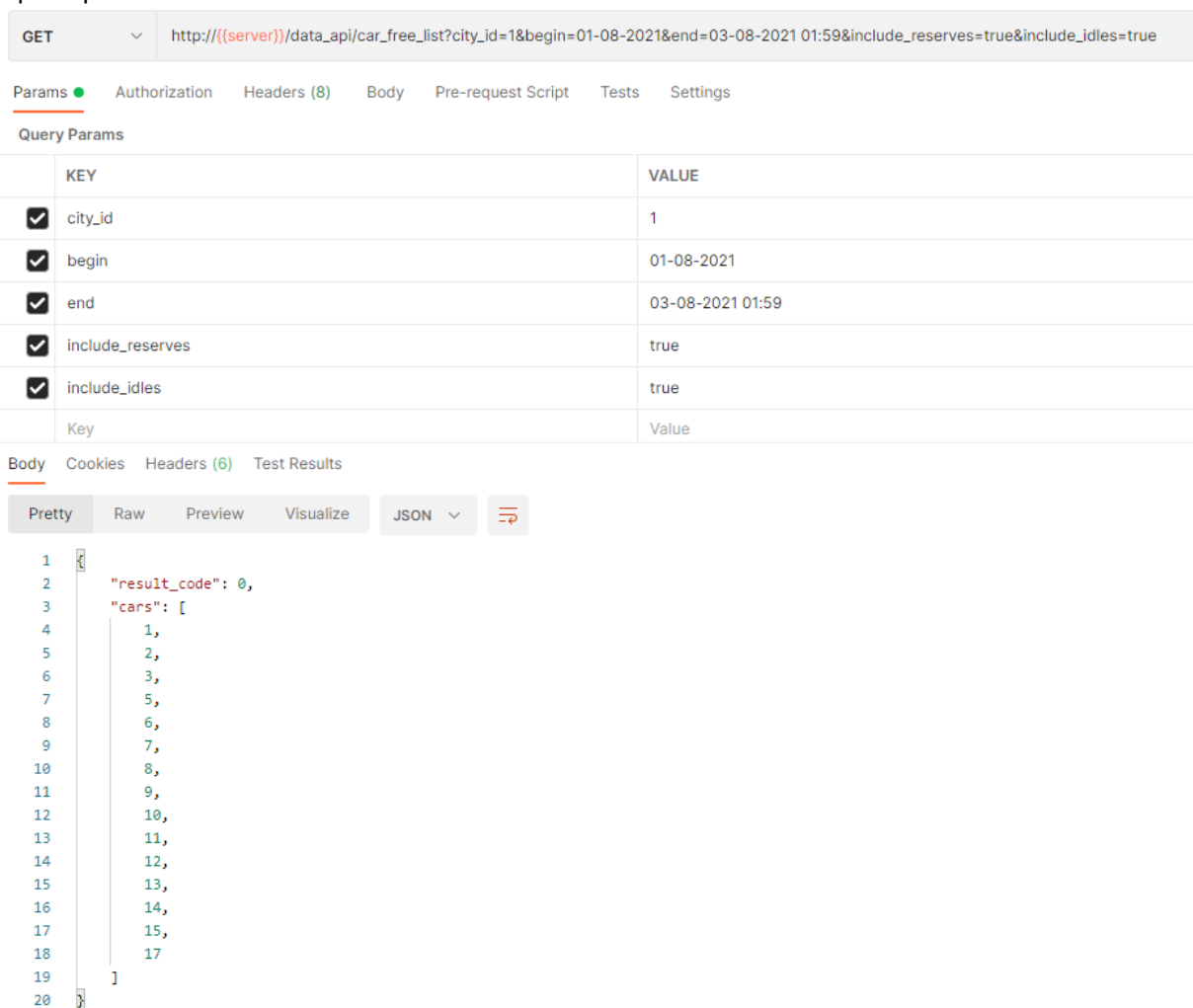
1. city\_id – идентификатор города (тип данных – целое число),
2. begin – начало периода (тип данных – дата+время, ГГГГ-ММ-ДД ЧЧ:ММ),
3. end – окончание периода (тип данных – дата+время, ГГГГ-ММ-ДД ЧЧ:ММ),
4. include\_reserves – флаг «учитывая брони» (тип данных – логическое значение)
5. include\_idles – флаг «учитывая сервисы» (тип данных – логическое значение)

Ответ:

```
{
  cars – массив идентификаторов автомобилей (тип данных – массив целых чисел)
}
```

В ответе возвращаются все автомобили, которые находятся в кластерах выбранного города в течение всего выбранного периода и при этом в выбранном периоде нет неудаленных аренд, броней (при условии include\_reserves) и сервисов (при условии include\_idles).

Пример:



The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** `http://{{server}}/data_api/car_free_list?city_id=1&begin=01-08-2021&end=03-08-2021 01:59&include_reserves=true&include_idles=true`
- Query Params Table:**

KEY	VALUE
city_id	1
begin	01-08-2021
end	03-08-2021 01:59
include_reserves	true
include_idles	true
- Body:**

```
{
  "result_code": 0,
  "cars": [
    1,
    2,
    3,
    5,
    6,
    7,
    8,
    9,
    10,
    11,
    12,
    13,
    14,
    15,
    17
  ]
}
```

#### 4 Запрос на получение списка автомобилей с краткой информацией

Запрос:

GET-запрос *car\_list* без параметров

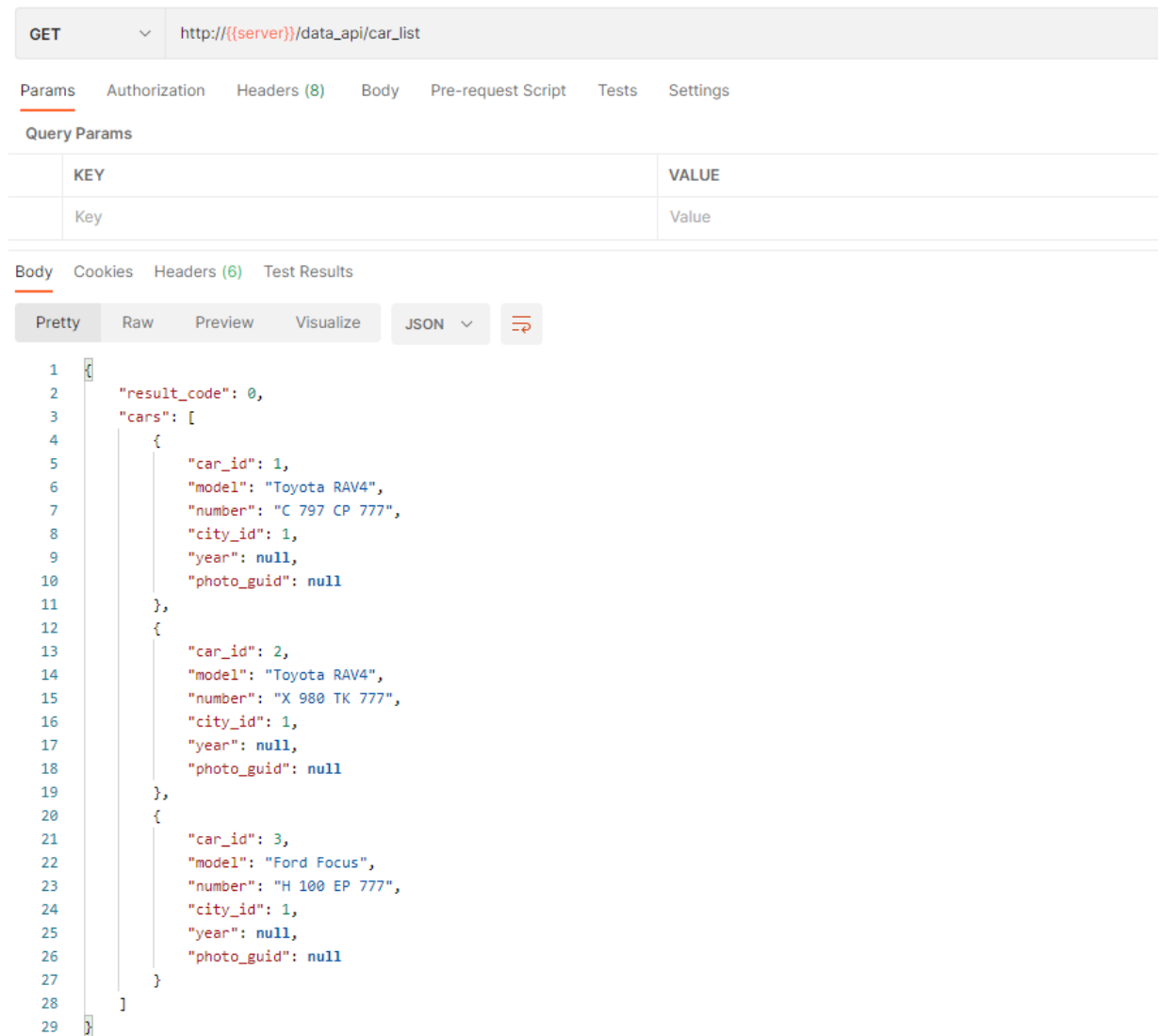
Ответ:

*cars* – массив автомобилей

```
{
  car_id – идентификатор автомобиля (тип данных – целое число),
  model – марка и модель (тип данных – текст),
  number – гос. номер (тип данных – текст),
  city_id – идентификатор города (тип данных – целое число),
  year – год выпуска (тип данных – целое число),
  photo_guid – guid фотографии (тип данных – текст)
}
```

В ответе возвращаются все автомобили, которые в момент обращения находятся в кластере.

Пример:



The screenshot shows a REST client interface. The top bar indicates a GET request to the URL `http://{{server}}/data_api/car_list`. Below the URL bar, there are tabs for Params, Authorization, Headers (8), Body, Pre-request Script, Tests, and Settings. The 'Query Params' section is empty. The 'Body' tab is selected, showing a JSON response in 'Pretty' format. The response is a JSON object with a 'result\_code' of 0 and a 'cars' array containing three car objects.

```
{
  "result_code": 0,
  "cars": [
    {
      "car_id": 1,
      "model": "Toyota RAV4",
      "number": "C 797 CP 777",
      "city_id": 1,
      "year": null,
      "photo_guid": null
    },
    {
      "car_id": 2,
      "model": "Toyota RAV4",
      "number": "X 980 TK 777",
      "city_id": 1,
      "year": null,
      "photo_guid": null
    },
    {
      "car_id": 3,
      "model": "Ford Focus",
      "number": "H 100 EP 777",
      "city_id": 1,
      "year": null,
      "photo_guid": null
    }
  ]
}
```

## 5 Запрос на получение списка тарифов

Запрос:

GET-запрос **tariff\_list**, параметры передаются через строку запроса:

1. **car\_id** – идентификатор автомобиля (тип данных – целое число), необязательное поле

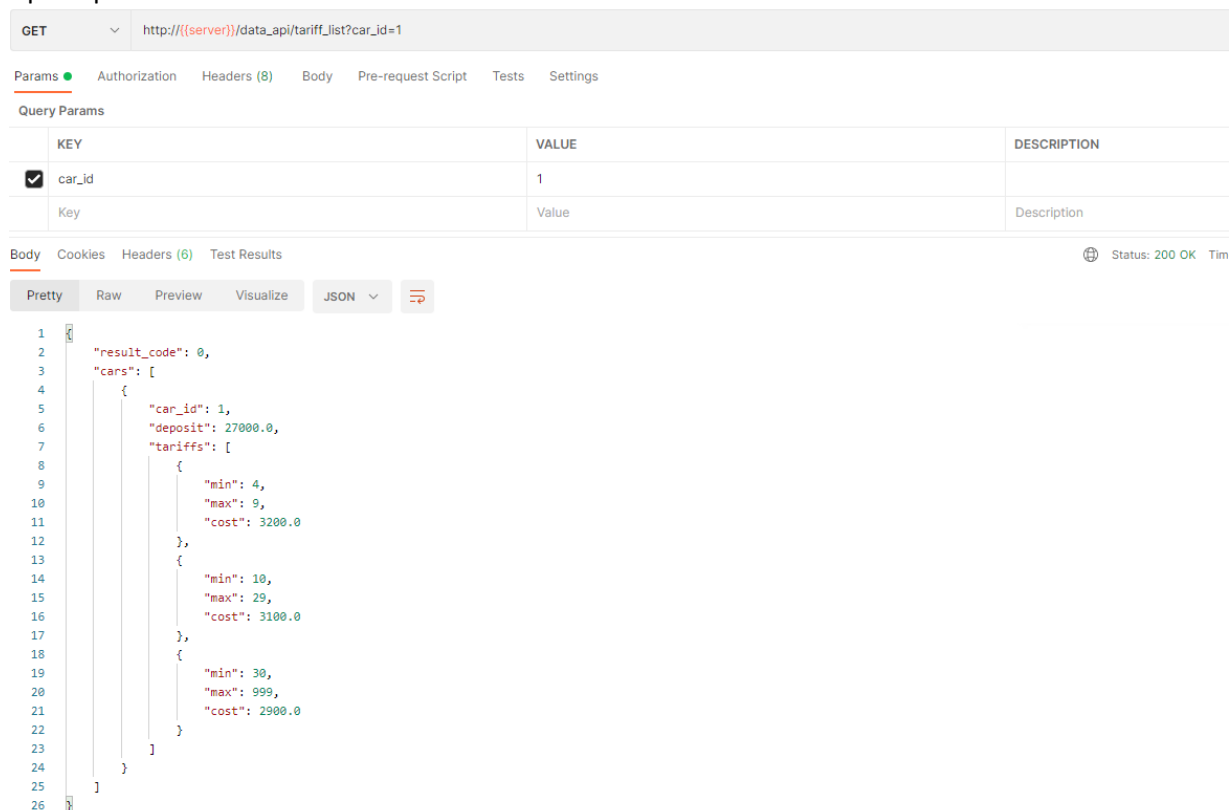
Ответ:

**cars** – массив автомобилей

```
{
  car_id – идентификатор автомобиля (тип данных – целое число),
  deposit – стоимость залога (тип данных – число),
  tariffs – массив тарифов
  {
    min – минимальное кол-во дней в периоде (тип данных – целое число),
    max – максимальное количество дней в периоде (тип данных – целое число),
    cost – стоимость аренды за один день (тип данных – число)
  }
}
```

В ответе возвращаются тарифы в тарифной сетке, помеченной галочкой «Тарифная сетка для сайта», для указанного автомобиля, если идентификатор автомобиля передан, либо, если идентификатор автомобиля не передан, для всех автомобилей, которые в момент обращения находятся в кластере.

Пример:



The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** http://(server)/data\_api/tariff\_list?car\_id=1
- Params:** car\_id (checked)
- Status:** 200 OK
- Body (JSON):**

```
{
  "result_code": 0,
  "cars": [
    {
      "car_id": 1,
      "deposit": 27000.0,
      "tariffs": [
        {
          "min": 4,
          "max": 9,
          "cost": 3200.0
        },
        {
          "min": 10,
          "max": 29,
          "cost": 3100.0
        },
        {
          "min": 30,
          "max": 999,
          "cost": 2900.0
        }
      ]
    }
  ]
}
```

## 6 Запрос на получение списка периодов, в которых автомобиль свободен

Запрос:

GET-запрос **car\_period\_list**, параметры передаются через строку запроса:

1. car\_id – идентификатор автомобиля (тип данных – целое число),
2. begin – начало периода (тип данных – дата+время, ГГГГ-ММ-ДД ЧЧ:ММ),
3. end – окончание периода (тип данных – дата+время, ГГГГ-ММ-ДД ЧЧ:ММ),
4. include\_reserves – флаг «учитывая брони» (тип данных – логическое значение)
5. include\_idles – флаг «учитывая сервисы» (тип данных – логическое значение)

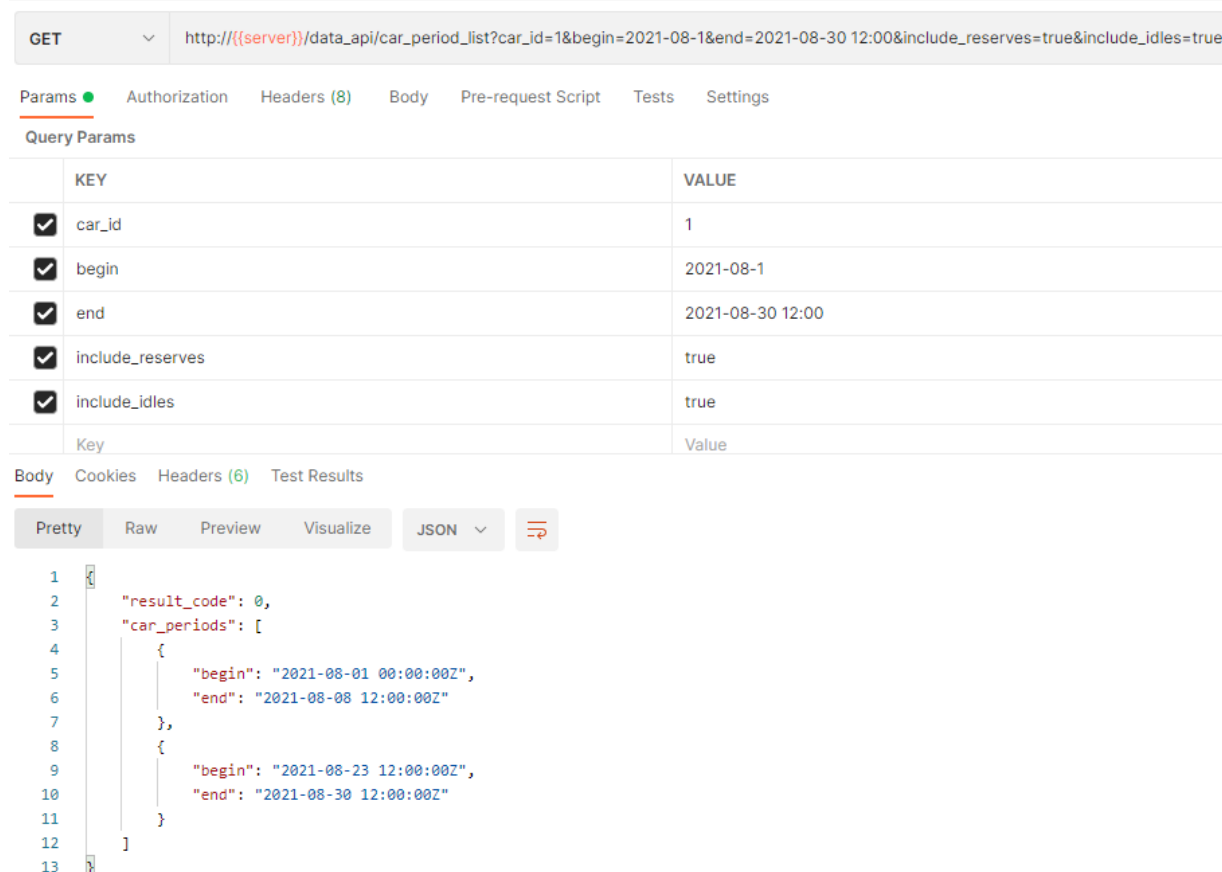
Ответ:

car\_periods – массив периодов

```
{
  begin – начало периода (тип данных – дата+время, ГГГГ-ММ-ДД ЧЧ:ММ),
  end – окончание периода (тип данных – дата+время, ГГГГ-ММ-ДД ЧЧ:ММ)
}
```

В ответе возвращаются все периоды автомобиля, в которые нет неудаленных аренд, броней (при условии include\_reserves) и сервисов (при условии include\_idles).

Пример:



The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** `http://{{server}}/data_api/car_period_list?car_id=1&begin=2021-08-1&end=2021-08-30 12:00&include_reserves=true&include_idles=true`
- Params:**

KEY	VALUE
<input checked="" type="checkbox"/> car_id	1
<input checked="" type="checkbox"/> begin	2021-08-1
<input checked="" type="checkbox"/> end	2021-08-30 12:00
<input checked="" type="checkbox"/> include_reserves	true
<input checked="" type="checkbox"/> include_idles	true
- Body:**

```
{
  "result_code": 0,
  "car_periods": [
    {
      "begin": "2021-08-01 00:00:00Z",
      "end": "2021-08-08 12:00:00Z"
    },
    {
      "begin": "2021-08-23 12:00:00Z",
      "end": "2021-08-30 12:00:00Z"
    }
  ]
}
```

## 7 Запрос на получение списка типов услуг

Запрос:

GET-запрос **service\_type\_list** без параметров

Ответ:

service\_types – массив типов услуг

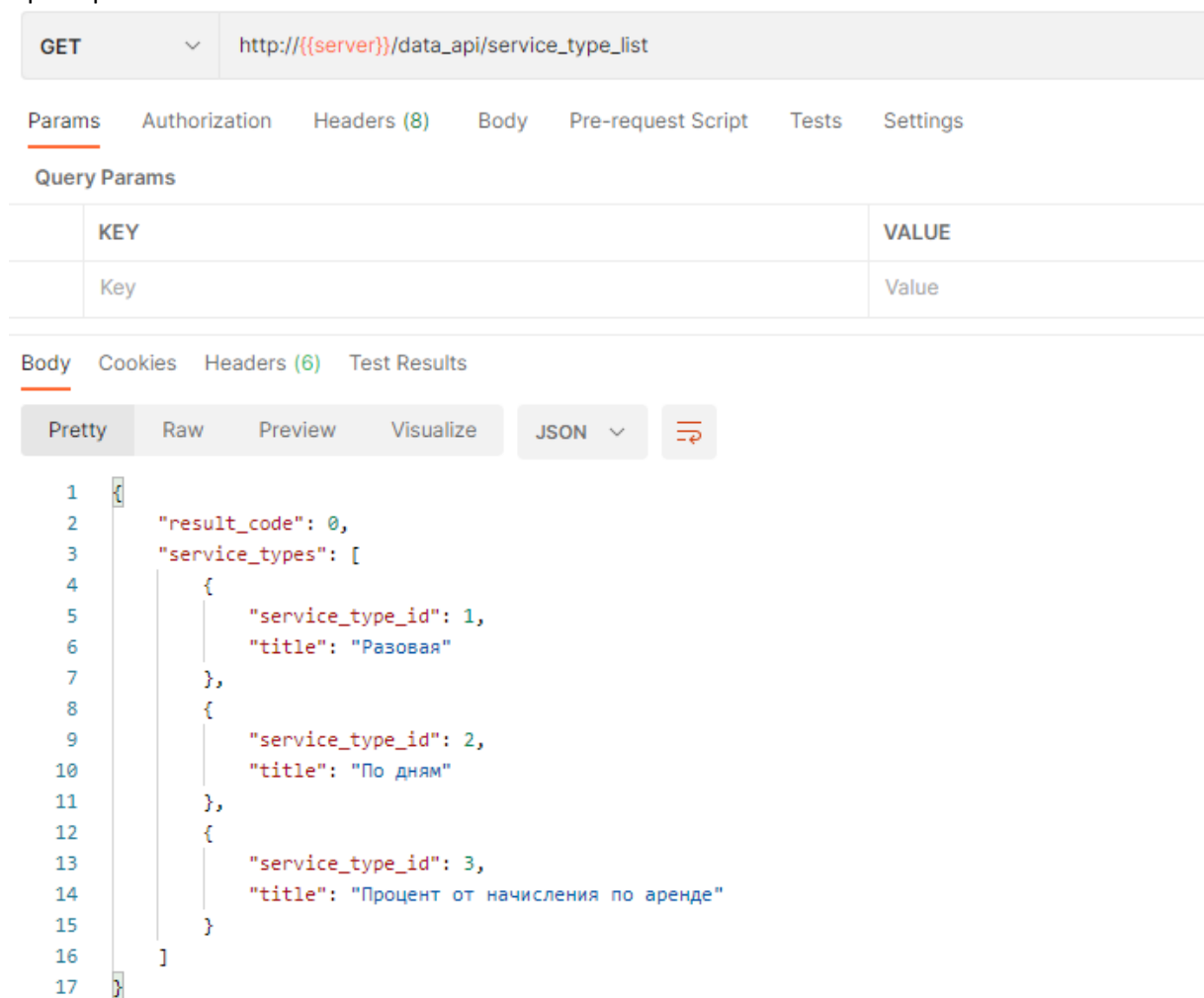
```
{
  service_type_id – идентификатор типа услуги (тип данных – целое число),
  title – название (тип данных – текст)
}
```

В ответе возвращаются все типы услуг.

По умолчанию в программе 3 типа услуг:

1. service\_type\_id = 1. Разовая. Для таких услуг стоимость указывается в основной валюте программы. Количество услуг не связано с количеством дней аренды.
2. service\_type\_id = 2. По дням. Для таких услуг стоимость указывается в основной валюте программы. Количество услуг равно количеству дней аренды.
3. service\_type\_id = 3. Процент от начисления по аренде. Для таких услуг стоимость указывается в процентах от стоимости тарифа.

Пример:



The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** http://{{server}}/data\_api/service\_type\_list
- Query Params:** A table with two columns: KEY and VALUE. The first row shows 'Key' and 'Value'.
- Body:** The response is displayed in JSON format, showing a result code of 0 and a list of service types.

```
{
  "result_code": 0,
  "service_types": [
    {
      "service_type_id": 1,
      "title": "Разовая"
    },
    {
      "service_type_id": 2,
      "title": "По дням"
    },
    {
      "service_type_id": 3,
      "title": "Процент от начисления по аренде"
    }
  ]
}
```

## 8 Запрос на получение списка услуг

Запрос:

GET-запрос *service\_list* без параметров

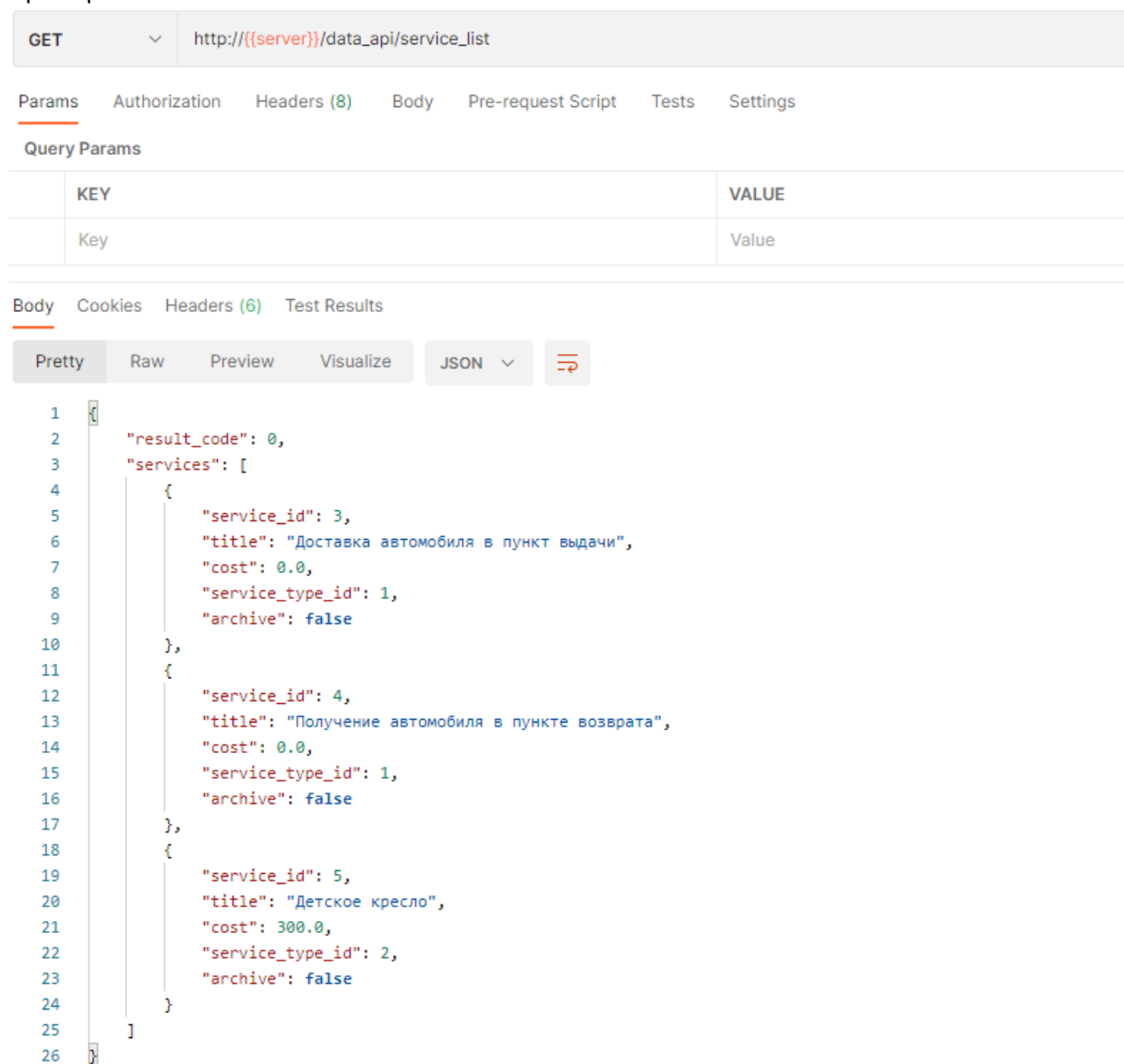
Ответ:

services – массив услуг

```
{
  service_id – идентификатор услуги (тип данных – целое число),
  title – название (тип данных – текст),
  cost – стоимость (тип данных – число),
  service_type_id – идентификатор типа услуги (тип данных – целое число),
  archive – архивность (тип данных – логическое значение. True для отправленных в архив
услуг)
}
```

В ответе возвращаются все услуги, кроме системных услуг «Перепробег 1 км» и «Повреждение».

Пример:



The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** `http://{{server}}/data_api/service_list`
- Params:** Query Params table with columns KEY and VALUE. The table is empty.
- Body:** JSON response in Pretty view.
 

```
1 {
2   "result_code": 0,
3   "services": [
4     {
5       "service_id": 3,
6       "title": "Доставка автомобиля в пункт выдачи",
7       "cost": 0.0,
8       "service_type_id": 1,
9       "archive": false
10    },
11    {
12      "service_id": 4,
13      "title": "Получение автомобиля в пункте возврата",
14      "cost": 0.0,
15      "service_type_id": 1,
16      "archive": false
17    },
18    {
19      "service_id": 5,
20      "title": "Детское кресло",
21      "cost": 300.0,
22      "service_type_id": 2,
23      "archive": false
24    }
25  ]
26 }
```



## 9 Запрос на получение стоимости заявки

Запрос:

GET-запрос **bid\_cost**, параметры передаются через строку запроса:

1. car\_id – идентификатор автомобиля (тип данных – целое число),
2. begin – начало периода (тип данных – дата+время, ГГГГ-ММ-ДД ЧЧ:ММ),
3. end – окончание периода (тип данных – дата+время, ГГГГ-ММ-ДД ЧЧ:ММ),
4. begin\_place\_id – идентификатор места выдачи (тип данных – целое число)
5. end\_place\_id – идентификатор места возврата (тип данных – целое число),
6. services – массив идентификаторов услуг (тип данных – массив целых чисел),  
необязательное поле

Ответ:

```
{
  cost – стоимость заявки (тип данных – число),
  prepay – предоплата (тип данных – число),
  deposit – залог (тип данных – число),
  error_message – сообщение об ошибке (тип данных – текст)
}
```

В ответе возвращается стоимость заявки на аренду указанного автомобиля в указанном периоде для тарифной сетки, помеченной галочкой «Тарифная сетка для сайта», включая указанные услуги, а также услуги по доставке автомобиля в пункт выдачи и получении автомобиля в пункте возврата, если в настройках мест выдачи/возврата задана стоимость; размер предоплаты с учетом настройки «Процент стоимости для предоплаты по заявке с сайта» тарифной сетки; депозит.

Варианты сообщения об ошибке:

«incorrect\_car» – если нет автомобиля с указанным идентификатором;

«incorrect\_dates» – если дата начала периода больше или равна дате окончания;

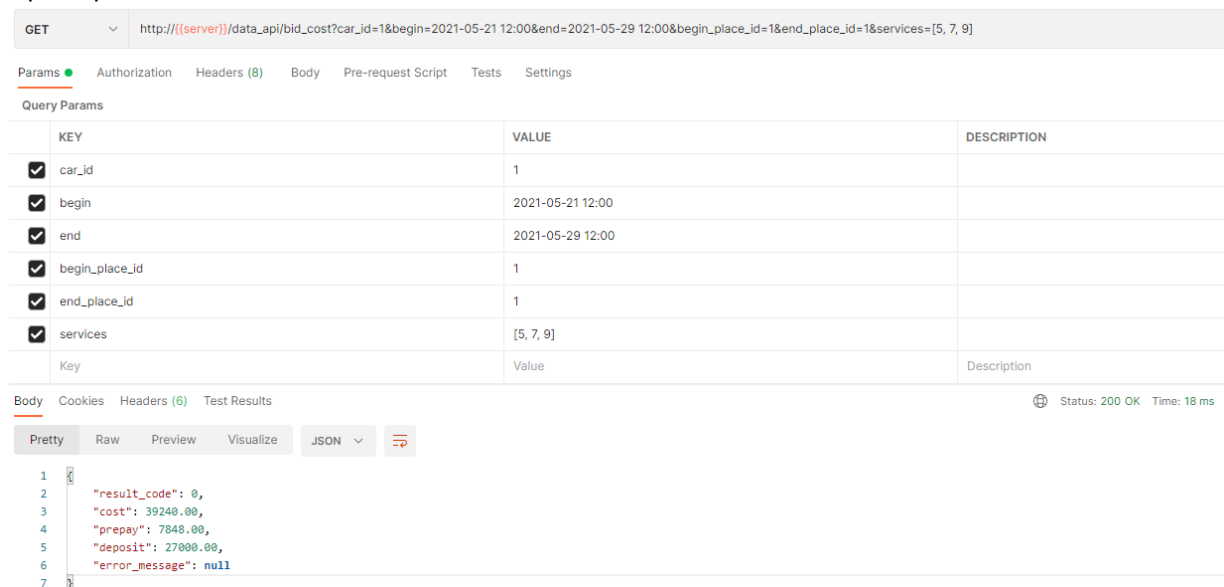
«incorrect\_start\_place» – если нет места выдачи/возврата с указанным идентификатором;

«incorrect\_end\_place» – если нет места выдачи/возврата с указанным идентификатором;

«incorrect\_services» – если нет хотя бы одной услуги в массиве указанных идентификаторов;

«no\_charge\_type» – если нет тарифной сетки, помеченной галочкой «Тарифная сетка для сайта».

Пример:



GET [http://\(server\)/data\\_api/bid\\_cost?car\\_id=1&begin=2021-05-21 12:00&end=2021-05-29 12:00&begin\\_place\\_id=1&end\\_place\\_id=1&services=\[5, 7, 9\]](http://(server)/data_api/bid_cost?car_id=1&begin=2021-05-21 12:00&end=2021-05-29 12:00&begin_place_id=1&end_place_id=1&services=[5, 7, 9])

Params Authorization Headers (8) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> car_id	1	
<input checked="" type="checkbox"/> begin	2021-05-21 12:00	
<input checked="" type="checkbox"/> end	2021-05-29 12:00	
<input checked="" type="checkbox"/> begin_place_id	1	
<input checked="" type="checkbox"/> end_place_id	1	
<input checked="" type="checkbox"/> services	[5, 7, 9]	
Key	Value	Description

Body Cookies Headers (6) Test Results Status: 200 OK Time: 18 ms

Pretty Raw Preview Visualize JSON

```
1 {
2   "result_code": 0,
3   "cost": 39240.00,
4   "prepay": 7848.00,
5   "deposit": 27000.00,
6   "error_message": null
7 }
```

## 10 Запрос на создание заявки

Запрос:

PUT -запрос **bid\_create**, параметры из опубликованных полей формы (multipart/form-data):

1. fio – ФИО клиента (тип данных – текст),
2. phone – телефон клиента (тип данных – текст),
3. car\_id – идентификатор автомобиля (тип данных – целое число),
4. begin – начало периода (тип данных – дата+время, ГГГГ-ММ-ДД ЧЧ:ММ),
5. end – окончание периода (тип данных – дата+время, ГГГГ-ММ-ДД ЧЧ:ММ),
6. begin\_place\_id – идентификатор места выдачи (тип данных – целое число)
7. end\_place\_id – идентификатор места возврата (тип данных – целое число),
8. services – массив идентификаторов услуг (тип данных – массив целых чисел),  
необязательное поле,
9. prepayment – размер предоплаты (тип данных – число), необязательное поле,
- 10.files – файлы клиента, каждый файл добавляется отдельным параметром «files»

Ответ:

```
{
  bid_id – идентификатор заявки (тип данных – целое число),
  bid_number – номер заявки (тип данных – целое число),
  error_message – сообщение об ошибке (тип данных – текст)
}
```

Результатом выполнения метода является создание нового клиента, если в базе не будет обнаружен клиент с переданными ФИО и номером телефона, прикрепление файлов к новому или существующему клиенту и создание заявки.

Варианты сообщения об ошибке:

«incorrect\_car» – если нет автомобиля с указанным идентификатором;  
 «incorrect\_dates» – если дата начала периода больше или равна дате окончания;  
 «incorrect\_start\_place» – если нет места выдачи/возврата с указанным идентификатором;  
 «incorrect\_end\_place» – если нет места выдачи/возврата с указанным идентификатором;  
 «incorrect\_services» – если нет хотя бы одной услуги в массиве указанных идентификаторов;  
 «no\_charge\_type» – если нет тарифной сетки, помеченной галочкой «Тарифная сетка для сайта»;  
 «no\_account» – если нет счета, помеченного галочкой «Счет для сайта», при наличии предоплаты.

Пример 1:

PUT ▼ http://{{server}}/data\_api/bid\_create

Params Authorization Headers (10) **Body** Pre-request Script Tests Settings

☐ none ☒ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

	KEY	VALUE	CONTENT TYPE
<input checked="" type="checkbox"/>	fio	Иванов Иван Иванович	Auto
<input checked="" type="checkbox"/>	phone	+79821111111	Auto
<input checked="" type="checkbox"/>	car_id	1	Auto
<input checked="" type="checkbox"/>	begin	2021-05-21 12:00	Auto
<input checked="" type="checkbox"/>	end	2021-05-29 12:00	Auto
<input checked="" type="checkbox"/>	begin_place_id	2	Auto
<input checked="" type="checkbox"/>	end_place_id	2	Auto
<input checked="" type="checkbox"/>	services	[5, 7, 9]	Auto
<input checked="" type="checkbox"/>	prepayment	500	Auto
<input checked="" type="checkbox"/>	files	Скан паспорт.jpg <span>×</span>	Auto
<input checked="" type="checkbox"/>	files	Скан паспорт, прописка.jpg <span>×</span>	Auto
	Key	Value	Auto

Body Cookies Headers (6) Test Results

Pretty Raw Preview Visualize JSON ▼ ≡

```

1 {
2   "result_code": 0,
3   "bid_id": 13,
4   "bid_number": 8,
5   "error_message": null
6 }
```

## Пример 2:

PUT ▼ http://{{server}}/data\_api/bid\_create

Params Authorization Headers (10) **Body** Pre-request Script Tests Settings

☐ none ☒ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

	KEY	VALUE	CONTENT TYPE
<input checked="" type="checkbox"/>	fio	Иванов Иван Иванович	Auto
<input checked="" type="checkbox"/>	phone	+79821111111	Auto
<input checked="" type="checkbox"/>	car_id	100	Auto
<input checked="" type="checkbox"/>	begin	2021-05-21 12:00	Auto

Body Cookies Headers (6) Test Results

Pretty Raw Preview Visualize JSON ▼ ≡

```

1 {
2   "result_code": 0,
3   "bid_id": null,
4   "bid_number": null,
5   "error_message": "incorrect_car"
6 }
```

## 11 Запрос на получение статуса заявки

Запрос:

GET-запрос **bid\_status**, параметры передаются через строку запроса:

1. phone – телефон клиента (тип данных – текст), только цифры,
2. bid\_number – номер заявки (тип данных – текст)

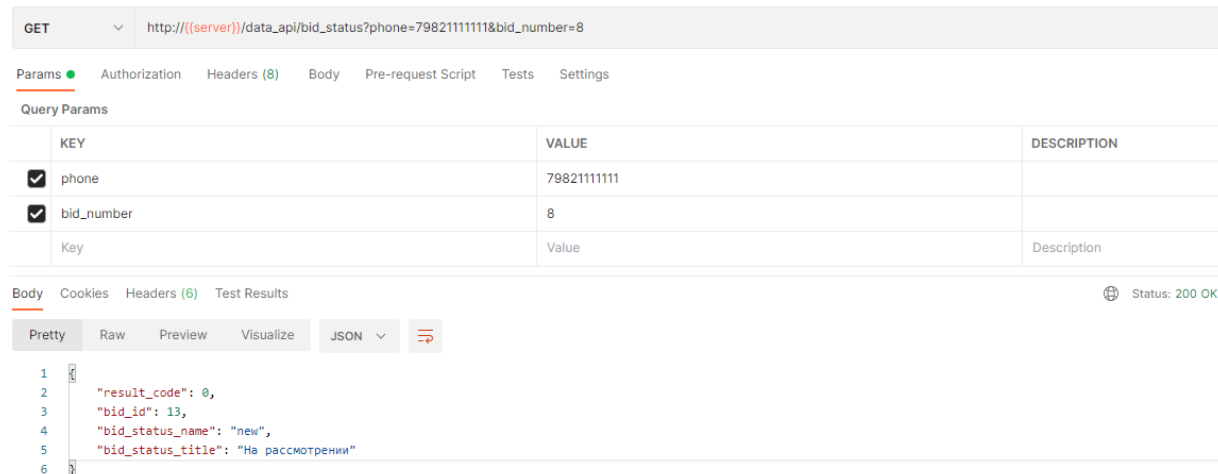
Ответ:

```
{
  bid_id – идентификатор заявки (тип данных – целое число),
  bid_status_name – статус заявки (тип данных – текст),
  bid_status_title – описание статуса заявки (тип данных – текст)
}
```

Варианты статусов заявок:

«not_found»	«Заявка не найдена»
«new»	«На рассмотрении»
«accepted_as_reserve»	«Создана бронь»
«accepted_as_rent»	«Создана аренда»
«rejected»	«Заявка отклонена»
«deleted»	«Заявка удалена»

Пример 1:



GET [http://\(server\)/data\\_api/bid\\_status?phone=7982111111&bid\\_number=8](http://(server)/data_api/bid_status?phone=7982111111&bid_number=8)

Params Authorization Headers (8) Body Pre-request Script Tests Settings

Query Params

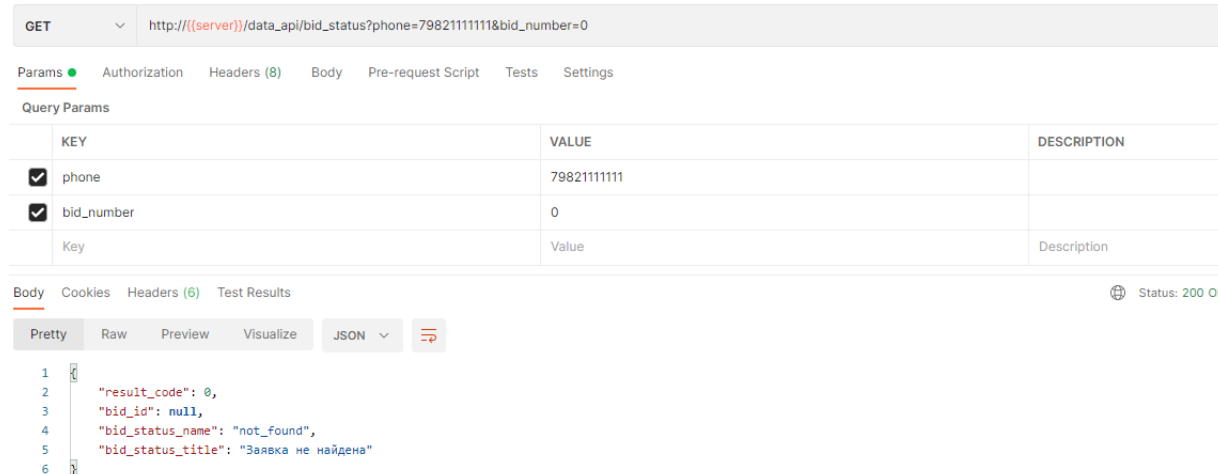
KEY	VALUE	DESCRIPTION
phone	7982111111	
bid_number	8	

Body Cookies Headers (6) Test Results Status: 200 OK

Pretty Raw Preview Visualize JSON

```
1 {
2   "result_code": 0,
3   "bid_id": 13,
4   "bid_status_name": "new",
5   "bid_status_title": "На рассмотрении"
6 }
```

Пример 2:



GET [http://\(server\)/data\\_api/bid\\_status?phone=7982111111&bid\\_number=0](http://(server)/data_api/bid_status?phone=7982111111&bid_number=0)

Params Authorization Headers (8) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE	DESCRIPTION
phone	7982111111	
bid_number	0	

Body Cookies Headers (6) Test Results Status: 200 OK

Pretty Raw Preview Visualize JSON

```
1 {
2   "result_code": 0,
3   "bid_id": null,
4   "bid_status_name": "not_found",
5   "bid_status_title": "Заявка не найдена"
6 }
```

## 12 Запрос на создание оплаты по заявке

Запрос:

PUT-запрос **bid\_payment\_create**, параметры из опубликованных полей формы (multipart/form-data):

1. bid\_id – идентификатор заявки (тип данных – целое число),
2. summ – сумма оплаты (тип данных – число),
3. transaction\_id – идентификатор транзакции (тип данных – текст), необходим для предотвращения дублирования оплат

Ответ:

```
{
  payment_id – идентификатор оплаты (тип данных – целое число),
  error_message – сообщение об ошибке (тип данных – текст)
}
```

Результатом выполнения метода является создание оплаты в указанной заявке и добавление этой оплаты в аренду, если по указанной заявке есть неудаленная аренда. В случае, когда аренда есть, но она удалена, связь между заявкой и арендой удаляется и оплата добавляется только в заявку.

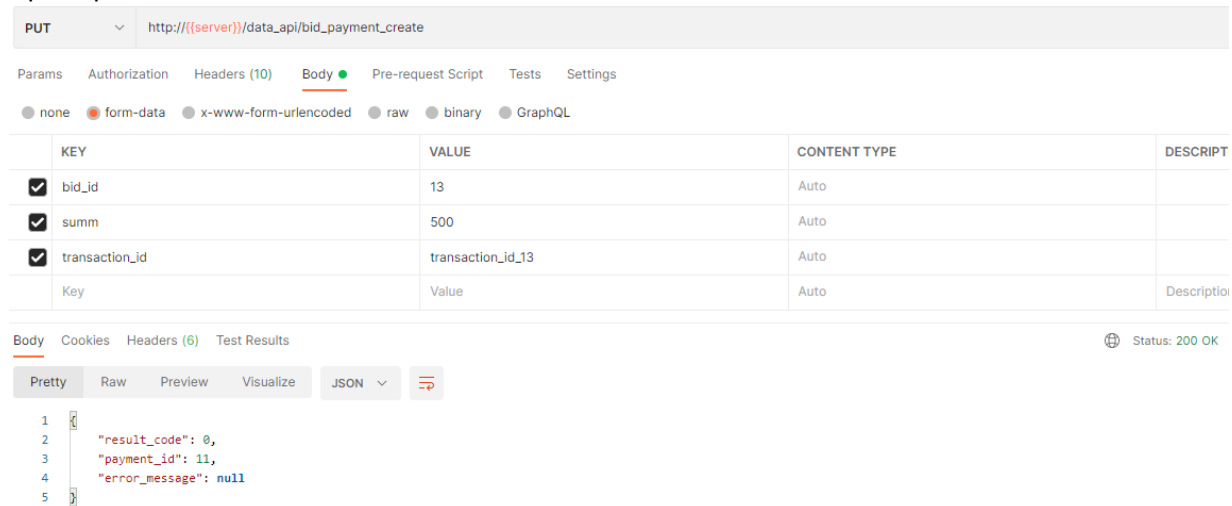
Варианты сообщения об ошибке:

«no\_bid» – если нет заявки с указанным идентификатором;

«no\_account» – если нет счета, помеченного галочкой «Счет для сайта»

«transaction\_id\_already\_used» – если с указанным идентификатором транзакции уже добавлена оплата.

Пример 1:



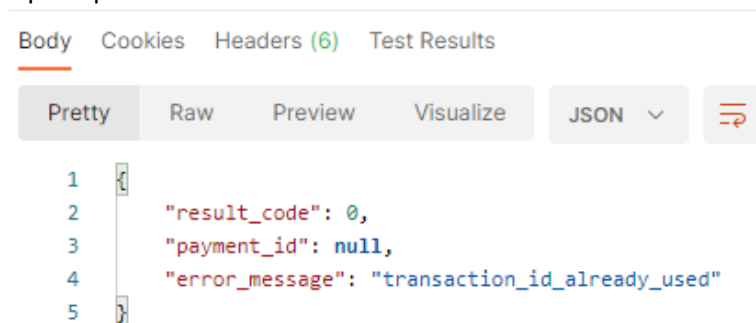
The screenshot shows a REST client interface with the following details:

- Method:** PUT
- URL:** http://{{server}}/data\_api/bid\_payment\_create
- Body Type:** form-data
- Body Data:**

KEY	VALUE	CONTENT TYPE	DESCRIPTION
bid_id	13	Auto	
summ	500	Auto	
transaction_id	transaction_id_13	Auto	
Key	Value	Auto	Description
- Test Results:** Status: 200 OK
- Response Body (JSON):**

```
{
  "result_code": 0,
  "payment_id": 11,
  "error_message": null
}
```

Пример 2:



The screenshot shows a REST client interface with the following details:

- Body Type:** JSON
- Response Body (JSON):**

```
{
  "result_code": 0,
  "payment_id": null,
  "error_message": "transaction_id_already_used"
}
```

## Механизм аутентификации

1	Авторизация. Получение токена
---	-------------------------------

Запрос:

POST-запрос `/api/v1/tokens/signin`, ожидается json-параметр со следующими полями:

```
{
  "UserName" : логин пользователя,
  "PasswordHash" : хэш пароля (Sha512),
  "LongToken" : необязательное поле. Ожидается логическое значение. Если поле не указано, то значение - false. Используется для получения токенов с большим временем жизни.
}
```

Ответ:

```
{
  "accessToken" : токен, которым нужно подписывать запросы,
  "refreshToken": токен, необходимый для обновления основного токена,
  "expires": кол-во секунд с 1970-01-01. Время жизни accessToken. Если LongToken имеет значение false, то время жизни равно 10 минутам. Если true, то примерно 15 лет
}
```

Пример:

POST

http://(server)/api/v1/tokens/signin

Send

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

Cookies

Beautiful

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

```
1 {
2   ... "UserName": "USER_FOR_SITE",
3   ... "PasswordHash": "3c9909afec25354d55daee21590bb26e38d53f2173b8d3dc3eee4c047e7ab1c1eb8b85103e3be7b0613b31bb5c9c36214dc9f14a42fd7a2fdb84856bca5c44c2",
4   ... "LongToken": true
5 }
```

2	Обновление токена
---	-------------------

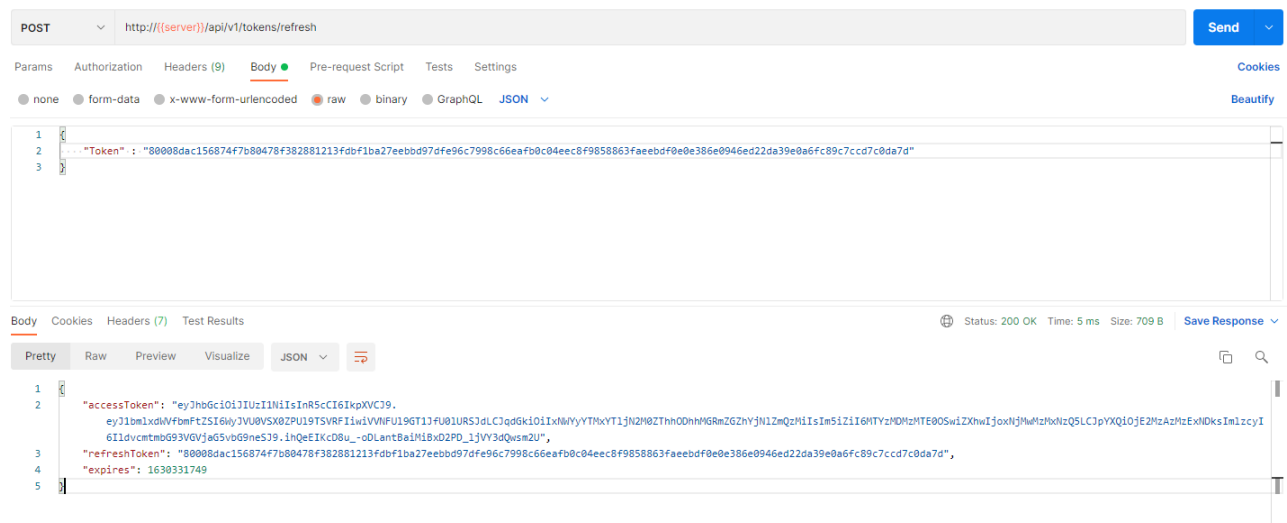
Запрос:

POST-запрос `/api/v1/tokens/refresh`, ожидается json-параметр со следующими полями:

```
{
  "Token": токен, необходимый для обновления основного токена
}
```

Ответ:

```
{
  "accessToken" : токен, которым нужно подписывать запросы,
  "refreshToken": токен, необходимый для обновления основного токена,
  "expires": кол-во секунд с 1970-01-01. Время жизни accessToken. Если LongToken имеет значение false, то время жизни равно 10 минутам. Если true, то примерно 15 лет
}
```



3	Применение токена при отправке запросов
---	---

Все запросы (в том числе на получение файлов) должны быть подписаны полученным токеном. Для этого в заголовке запроса необходимо прописывать:

**Authorization:** "Bearer <accessToken>"

Пример:

	Key	Value	Description
☰ <input checked="" type="checkbox"/>	Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1bmVudWVmbmFtZSI6WyJXU19HVUVTVClldlTX0dVRVNUIl0slmp0aSa16lUyZDdmM2ZhYzdmODRIZGZiImZvNjZQ3YzlwZmM2NmZjliwibmJmIjoxNjE2NjU3Njl0LCJleHAiOjE2MTY2NTgyMjQslmlhdC16MTYxNjY1ZS5yNCwiaXNzIjoiriV29ya2Zsb3dUZWNobm9sb2d5In0.qhmjBiYtK5NpV-0n_pKmN-6jN8NcUF3T2sFhw9TdjnA	Description
	New key		
	Response		

## Прочее

### 1 Получение файлов

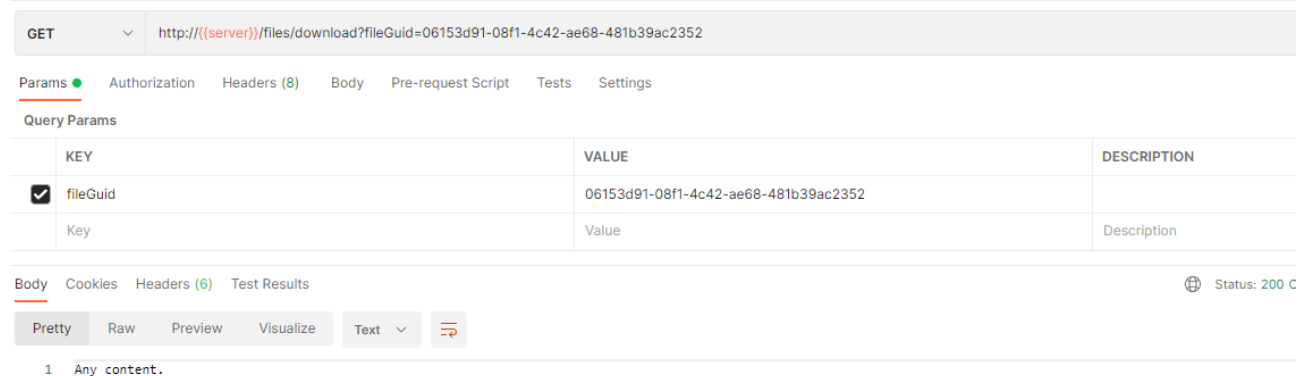
Запрос:

GET-запрос **/files/download**, параметры передаются через строку запроса:

1. fileGuid – идентификатор файла (тип данных – текст),

Ответ – файл.

Пример:



The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** `http://{{server}}/files/download?fileGuid=06153d91-08f1-4c42-ae68-481b39ac2352`
- Params:** Authorization, Headers (8), Body, Pre-request Script, Tests, Settings
- Query Params:**

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> fileGuid	06153d91-08f1-4c42-ae68-481b39ac2352	
Key	Value	Description
- Body:** Cookies, Headers (6), Test Results
- Status:** 200 C
- Response Format:** Pretty, Raw, Preview, Visualize, Text
- Response Content:** 1 Any content.



## Механизм работы с датами

Даты со временем (в формате даты) между сервером и клиентом всегда передаются в UTC. Это означает, что все поля типа «дата+время» сервер перед отправкой преобразует в UTC. При получении полей типа «дата+время» сервер обрабатывает их как даты в UTC и преобразует в свой часовой пояс.

Дата и время по отдельности не преобразуются и передаются/принимаются как есть.

Считается, что любой отправленный или принятый объект типа «Дата+время» хранит дату и время в UTC.