REST API QRManager

Версия 3.2.1 - 27.01.2023

Версия 3.2 - 24.01.2023

Версия 3.1 - 28.11.2022

Версия 3.0 - 18.11.2022

Версия 2.9 - 14.11.2022

Версия 2.8 - 10.11.2022

Версия 2.7 - 19.09.2022

Версия 2.6 - 11.05.2022

Версия 2.5 - 16.12.2021

Версия 2.4 - 26.11.2021

Версия 2.3 - 12.11.2021

Версия 2.2 - 26.10.2021

Версия 2.1 - 06.08.2021

Версия 2.0 - 15.07.2021

🌍 QRManager REST API для онлайн-магазинов и других внешних клиентов (2.0)

Общая информация

С помощью данного API онлайн-магазины смогут получать и отображать QR-коды на оплату заказов. Например, QR-код можно сгенерировать в корзине онлайн-магазина или в квитанции.

Для приема СБП-платежей Вашей организацией оставьте заявку на нашем сайте.

Как работают QR-коды. API генерирует QR-коды в двух вариантах: изображением или ссылкой. Покупатель может навести камеру смартфона на изображение QR-кода и ему откроется его мобильный банк по умолчанию. При переходе по ссылке так же откроется мобильный банк. В нём покупатель подтверждает перевод средств.

Пример QR-кода изображением

Пример QR-кода ссылкой

https://qr.nspk.ru/AS10003P3RH0LJ2A9ROO038L6NT5RU1M?type=01&bank=000000000001&sum=10000&cur=RUB&crc=F3D0

По всем вопросам работы с API можно обращаться на почту post@qrm.ooo

Термины и сокращения

QRM - платформа QRManager, сервис и мобильное приложение, которое позволяет принимать безналичную оплату без POS-терминала, на обычном смартфоне по QR-коду.

СБП - Система Быстрых Платежей, сервис, который позволяет физическим лицам мгновенно (в режиме 24/7) совершать межбанковские переводы по номеру мобильного телефона или с использованием QR-кодов себе или другим лицам и компаниям.

POS терминал - это электронное программно-техническое устройство для приёма к оплате платёжных карт. Также под POS-терминалом часто подразумевают весь программно-аппаратный комплекс, который установлен на рабочем месте кассира.

QR-терминал - это виртуальный POS-терминал, зарегистрированный в СБП для принятия платежей с применением QR-кода. Можно приравнять онлайн-магазин к QR-терминалу, как к единственной высокопроизводительной кассе в физическом магазине. Также можно иметь много QR-терминалов в рамках одного магазина.

ОФД - Оператор Фискальных Данных - юридическое лицо, созданное специально для осуществления приёма, обработки, хранения и передачи фискальных данных в ФНС России.

ЭФД - Электронный Фискальный Документ или электронный чек.

Организация клиента - ЮЛ или ИП, организация продавца, который использует платформу QR-Manager для принятия безналичных платежей по QR-кодам через СБП.

Аутентификация

Для имплементации и тестирования взаимодействия с QRManager REST API в тестовой среде воспользуйтесь данным ключом API:

🔑 **Тестовый ключ: lDvAIB7E.K02zjE1o7P1ZiwDQi5yC4EBrP71Fr8Js**

Адрес сервиса

https://app.devwapiserv.qrm.ooo/

Это URL тестовой среды для разработки. Продакшн-URL для проведения операций с реальными деньгами выдаётся по запросу на почту dev@qrm.ooo после успешной реализации всех методов из этой документации. Тема письма на dev@qrm.ooo обязательно должна быть такой: “Запрос продакшн-URL REST API QRManager”.

Описание пользовательских сценариев

1. **Как получать QR-коды по REST API QR-Manager**
    a. Зайти в приложение QR-Manager и зарегистрироваться как владелец компании (Следуйте инструкции по регистрации в QR-Manager, которая есть в приложении или обратитесь в службу поддержки QRM)
    b. Получить merchant ID и зарегистрировать QR-терминал. В его настройках выбрать режим работы "REST API"
    c. Открыть в приложении зарегистрированный QR-терминал и нажать кнопку "Сгенерировать ключ API". API ключ отобразится в приложении
    d. Данный ключ нужно сохранить в защищенном хранилище API ключей на сервере онлайн-магазина
    e. Начать делать запросы на QR-коды с API ключом в заголовках.

    Т.о., один API-ключ соответствует одному QR-терминалу со своим MCC (или назначением платежей). Если нужно принимать плату на различные расчётные счета или за продукцию разных типов и с различным НДС, то рекомендуется зарегистрировать несколько QR-терминалов и в каждом сгенерировать свой API-ключ.

2. **Как покупатель оплачивает по QR в онлайн-магазине**
    a. Покупатель заходит в свою корзину в онлайн-магазине, в которой набраны какие-либо товары или услуги, и нажимает кнопку "Оплатить по QR-коду"
    b. Получив сигнал от клиента, сервер онлайн магазина запрашивает в QR-Manager QR-код на итоговую сумму покупки, включив в заголовок запроса API ключ. Если ключ валиден, QR-Manager авторизует клиент и генерирует для магазина QR-код
    c. Онлайн-магазин забирает по указанному URL изображение QR-кода и отображает его для покупателя в корзине. Рекомендуется отобразить покупателю не только изображение, но и кликабельную платёжную ссылку

СБП, которая также приходит в ответе на запрос QR-кода. Это нужно для удобства покупателей, которые совершают оплату с того же устройства, на котором выбирают "Оплату по QR".

d. Покупатель наводит камеру смартфона на экран компьютера и на смартфоне открывается приложение онлайн-банка по выбору покупателя, либо по умолчанию стоящее у него в системе. Действие может занять от секунды до нескольких минут

e. Покупатель подтверждает платёж и СБП заботится о том, чтобы деньги со счёта покупателя были переведены на счёт компании

f. Чтобы получить подтверждение об успешном платеже, либо об ошибке, онлайн-магазин устанавливает с сервером QRM соединение по веб-сокету. По нему клиент получает сообщение о смене статуса платежа на "paid"

g. Когда подтверждение об удачно совершенном платеже пришло, онлайн-магазин может отобразить сообщение об успешном платеже в корзине покупателя

3. **Как в онлайн-магазине подключить фискализацию через QR-Manager**

   a. Зарегистрироваться в приложении QR-Manager

   b. В приложении зарегистрировать QR-терминал

   c. На экране настройки QR-терминала выбрать опцию, которая включает фискализацию операций, проведённых через данный QR-терминал. Также нужно выбрать опцию, которая переводит его в онлайн-режим работы

   d. Сохранить QR-терминал с этими настройками

   e. На экране с подробным описанием QR-терминала выбрать опцию "Сгенерировать API ключ"

   f. Скопировать полученный ключ и использовать при разработке.

Теперь все покупки, совершенные с использованием QR-кодов данного QR-терминала, будут регистрироваться в налоговой службе, а покупатель будет получать электронный чек.

> ⚠️ Решение о том, производит ли Ваша организация фискализацию через своего ОФД, или Вы подключаетесь к ОФД через платформу QRM принимает бухгалтерия, владелец организации или иное ответственное лицо. QRM сотрудничает с ОФД АТОЛ и OFD.ru.

> 📌 Даже если Вы не подключили фискализацию, Вашим клиентам всегда будут доступны детализированные сводки товаров/услуг, которые они приобрели (детали заказа) по ссылке www.qrm.ooo/p/{number}/, где {number} - номер операции.
> Пример деталей заказа без номенклатуры: ***https://www.qrm.ooo/d/7404-8204-1521-7112/***
> Пример деталей заказа с номенклатурой: ***https://www.qrm.ooo/d/6569-3450-5180-1089/***

История изменений

Дата	Описание изменений
15.07.2021	создана минимальная версия QRM REST API 2.0. По сравнению с 1.0 изменены способы получения API-ключа, QR-кода и статуса об его оплате. Изменен параграф "Особенности ранней версии 1.0 QRM REST API". Убрано описание пользовательского сценария "Как понять, что QR оплачен и товар можно реализовать", т.к. нуждается в переработке.
06.08.2021	изменено название заголовка запросов: X-Api-Key. Дополнен список терминов и сокращений. Добавлен пользовательский сценарий по подключению фискализации через QR-Manager. Обновлены особенности ранней версии 1.0 QRM REST API. Добавлено описание эндпойнта для получения фискального документа по операции. Добавлен тестовый API ключ. Добавлены тестовые запрос и ответ на генерацию QR-кода.
15.09.2021	изменена команда на отмену операции в веб-сокете. Поменялся адрес сервиса и, следовательно, тестовые данные.

21.10.2021	обновлены особенности ранней версии данной документации. Обновлен тестовый API-ключ. Обновлена схема тела запроса QR-кода. Обновлен список кодов статуса операции в веб-сокете. Обновлены схемы ошибок на запрос получение ЭФД.
12.11.2021	изменен способ открытия веб-сокета, добавлено пояснение, в каких случаях покупатели получают детализацию своих заказов.
30.11.2021	добавлено поле payment_purpose в запрос на получение QR-кода, получение фискального чека разделено на два запроса, статусы оплаты QR-кодов теперь на русском, добавлен метод REST API для получения статуса QR-кода.
16.12.2021	расширено описание фискализации, добавлен ОФД OFD.ru, документация к нему, тестовые данные для ОФД АТОЛ и OFD.ru. Добавлен callback по получению статуса ЭФД и самого ЭФД.
11.05.2022	добавлены методы REST API по работе с кассовыми ссылками и возвратом средств; изменены ключи API и URL тестовой среды.
19.09.2022	добавлен метод формирования ранее добавленных товаров, отображение этих товаров в сведеньях о заказе и получения списка товаров, пример: “***https://www.qrm.ooo/d/6569-3450-5180-1089/***”, добавлено “назначение платежа” в кассовую ссылку
10.11.2022	обновлены “Сведения о заказе”. В методе запроса статуса по веб-сокету в ответе добавлено “operation_uuid”, добавлен пример реализации запросов по веб-сокету
14.11.2022	добавлены методы для взаимодействия с программой лояльности BenefittY
18.11.2022	добавлен пользовательский сценарий для работы с веб-сокетом, изменено описание команды “cancel”, удалена команда “disconnect”
28.11.2022	удален метод отправки кода подтверждения для списания баллов
24.01.2023	В ответе на метод “Активация кассовой ссылки” исправлено с “operationId”, на “operation_Id”, так же удален “qr-ttl” - время жизни берется из настроек QR-терминала В методе “Регистрация Кассовой ссылки” добавлено поле “qrc_id”
27.01.2023	Добавлена информация об изменении платежной ссылки, для удобства оплаты

Особенности ранней версии 2.4 QRM REST API

Присутствуют временные заглушки и механизмы для более удобной разработки.

1. В тестовой зоне для разработки, представленной в этой документации, по QR-коду не могут быть переведены деньги, но при наведении камеры смартфона на QR-код и переходе по ссылке должен открываться мобильный банк по умолчанию. Также можно зайти в приложение любого мобильного банка и отсканировать QR-код. В любом случае мобильный банк выдаст сообщение наподобие представленного на Рис.1. QR-коды станут валидными, если изменить адрес сервиса на продакшн-url.

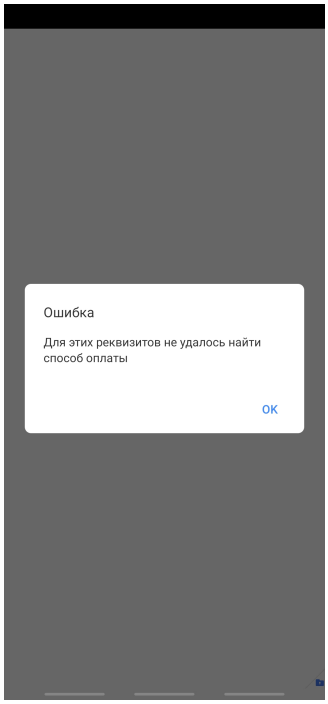


Рис. 1

Описание API

1. Получение API ключа

Чтобы получить доступ к методам QRManager REST API в продуктовой среде, владельцу онлайн-магазина или ответственному лицу необходимо зарегистрироваться в мобильном приложении QRManager. В нем нужно сгенерировать ключ API. Воспользуйтесь инструкцией "Получение ключа API для QR-терминала", которая находится здесь. Рекомендуется сохранить ключ в защищенном хранилище ключей на своём сервере.

2. Получение QR-кода

2.1 Получение QR-кода

🔑 **Тестовый ключ: IDvAIB7E.K02zjE1o7P1ZiwDQi5yC4EBrP71Fr8Js**

🌐 **POST** /operations/qr-code/

HEADER PARAMETERS

X-Api-Key (required, string)

REQUEST BODY SCHEMA

application/json

REQUEST BODY

```
{
  "sum": 1000,
  "qr_size": 400,
  "payment_purpose": "Иванов И.И. Договор №345567356324, плата за обучение"
}
```

Пояснения к полям

Aa Название поля	Обязательность	Тип	Описание	Диапазон допустимых значений
sum	обяз.	integer	Сумма за всю покупку в копейках	0 ... 1 000 000 00
qr_size	необяз.	integer	Задаёт и высоту и ширину изображения QR-кода в пикселях. Если клиент отправляет данный параметр в запросе, то в ответе придут и платёжная ссылка от СБП и готовое изображение QR-кода, содержащее ссылку от СБП	100 ... 1000
payment_purpose	необяз.	string	Короткий текст, который описывает цель/предмет платежа. Фиксируется в СБП. Покупатель может увидеть этот текст в подробной информации о платеже в своём банковском приложении/клиенте. Например, в истории операций по карте. Продавец или владелец бизнеса увидят этот текст в истории операций в приложении QRManager.	До 140 символов

⚠️ **Время жизни QR-кода или ttl применяется из настроек QR-терминала, на котором код создан. Отдельно для каждой операции ttl задать нельзя.**

⚠️ Есть 2 способа доставить ЭФД или детали заказа плательщику:

1. В запросе на фискализацию (см. ниже) указать email плательщика
2. Вставить ссылку на детали заказа в payment_purpose.

Второе QRManager делает по умолчанию. Но если Вы заполняете поле "payment_purpose" своим текстом, то он заменяет наш, в котором есть ссылка на детали заказа. В этом случае вам обязательно нужно осуществлять фискализацию через QRManager и указать email плательщика в запросе на фискализацию. Конечно, если Вам важно, чтобы соблюдался 54-ФЗ и если вы не выдаёте покупателе печатные чеки.

RESPONSES

▼ 200 Успешный запрос

Тело ответа

```
{
  "results": {
    "operation_id": "8d585d08-96c6-4005-abb9-aa99f822019c",
    "number": "4316-4040-0201-6359",
    "qr_link": "https://qr.nspk.ru/7921408844781707150427388936443 5?type=01&bank=12345&sum=10000&cur=RUB&crc=F3D0",
    "qr_img": "http://app.dewwapiserv.qrm.ooo/media/qr_codes/qr_code_7921408844781707150427388936443 5.png"
  }
}
```

Пояснения к полям

Aa Название	Описание	Тип
qr_img	URL изображения QR-кода заданного размера, хранящегося на сервере QRM до истечения заданного ttl или факта оплаты	URL
qr_link	Платёжная ссылка от СБП	URL
number	Уникальный номер заказа в формате XXXX-XXXX-XXXX-XXXX из букв и цифр. Необходим для фискализации данной операции	string
operation_id	UUID операции. Необходим для получения статуса оплаты QR-кода по веб-сокету	string

▼ 400 Запрос завершился ошибкой

1. Если не получен ответ из банка то через 5 секунд клиент получит сообщение об ошибке

   Тело ответа

   ```
   {
     "non_field_errors": [
       "Истекло время ожидания QR-кода"
     ]
   }
   ```

2. Ошибки сериализации (возможны на каждое из полей REQUEST BODY. В данном случае вместо <field> может быть sum или qr_size).

   Формат тела ответа

   ```
   {
     "<field>": [
       "<message>"
     ],

   }
   ```

3. Если запрошен статус платежа с несуществующим UUID

   Тело ответа

```
{
  "non_field_errors": [
    "QRT не обнаружен с uuid: <uuid>"
  ]
}
```

4. Ошибка произошла на стороне СБП или банка

   Тело ответа

```
{
  "detail": "<Ошибка, полученная из банка или СБП>"
}
```

Тестовые данные

▼ **Пример запроса**

```
curl -X POST "https://app.devwapiserv.qrm.ooo/operations/qr-code/" -H  "accept: application/json" -H  "X-Api-Key: ViVuB0oA.Fs0p
QYpduk99zxZoeZR6gVhFx8itg4yP" -H  "Content-Type: application/json" -H  "X-CSRFToken: XeBqlIvv91UiIZYHZWJrKBEYEpLSs8BmwZBlRQAgSU
JcBVKWduFHuN55JFVUfftS" -d "{  \"sum\": 10000,  \"qr_size\": 600}"
```

▼ **Ожидаемый ответ**

```
{
  "results": {
    "operation_id": "8d585d08-96c6-4005-abb9-aa99f822019c",
    "number": "4316-4040-0201-6359",
    "qr_link": "https://qr.nspk.ru/792140884478170715042738893644 35?type=01&bank=12345&sum=10000&cur=RUB&crc=F3D0",
    "qr_img": "http://app.devwapiserv.qrm.ooo/media/qr_codes/qr_code_792140884478170715042738893644 35.png"
  }
}
```

2.2 Получение QR-кода с номенклатурой

🔑 **Тестовый ключ: GZJK4okP.w0gCTfpBU7q3lO3oyTZIBu9QGIV3GM9p**

🌐 **POST** /operations/qr-code/

HEADER PARAMETERS

X-Api-Key (required, string)

REQUEST BODY SCHEMA

application/json

REQUEST BODY

```
{
  "sum": 1000,
  "qr_size": 400,
  "nomenclature": [
    {
      "product_id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
      "count": 0,
      "price": 0
    }
  ]
}
```

Пояснения к полям

Aa Название поля	Обязательность	Тип	Описание	Диапазон допустимых значений
sum	обяз.	integer	Сумма за всю покупку в копейках	0 ... 1 000 000 00
qr_size	необяз.	integer	Задаёт и высоту и ширину изображения QR-кода в пикселях. Если клиент отправляет данный параметр в запросе, то в ответе придут и платёжная ссылка от СБП и готовое изображение QR-кода, содержащее ссылку от СБП	100 ... 1000
product_id	обяз.	string	Id продукта, ранее добавленного в терминал	
count	обяз.	integer	Количество товара	1 … 1000
price	обяз.	integer	Цена товара в копейках	1 … 1 000 000 00

⚠️ **Время жизни QR-кода или ttl применяется из настроек QR-терминала, на котором код создан. Отдельно для каждой операции ttl задать нельзя.**

⚠️ В этом случае назначение платежа будет браться назначение платежа из свойств QR-терминала.
В приложении банка, клиент-покупатель увидит, пример: ***“ИП Иванов. Оплата заказа. Правила списания НДС внутри заказа.”https://www.qrm.ooo/d/6569-3450-5180-1089/”***.

Тестовые данные

▼ **Пример запроса**

```
curl -X POST "https://app.devwapiserv.qrm.ooo/operations/qr-code/" -H  "accept: application/json" -H  "X-Api-Key: bpkNMc19.gCrEPynUpwBsiujn7ZiYeTCmtgJkXVj8" -H  "Content-Type: application/json" -H  "X-CSRFToken: FQi2HlmfvjWKs1WAXpEfXMIf6KmSb9bZNBwCJdlKHYBfhHg95D9GRmP3DrKCARjN" -d "{  \"sum\": 91500,  \"qr_size\": 100,  \"nomenclature\": [    {      \"product_id\": \"e7d5af05-49f2-45d5-b89e-fdd3ef918502\",      \"count\": 1,      \"price\": 91500    }  ]}"
```

▼ **Ожидаемый ответ**

```
{
  "results": {
    "operation_id": "42760272-d792-4311-85b5-50cc755173a3",
    "number": "6627-0789-5628-4186",
    "qr_link": "https://qr.nspk.ru/260007127373170735557438028579 51?type=01&bank=12345&sum=91500&cur=RUB&crc=F3D0",
    "qr_img": "https://app.devwapiserv.qrm.ooo/media/qr_codes/qr_code_260007127373170735557438028579 51.png"
  }
}
```

Изменение платежной ссылки

⚠️ **При переходе по платежной ссылки у клиентов встречаются проблемы связанные с переходом в приложение не того банка**

Решение:

Изменение платежной ссылки с формата:

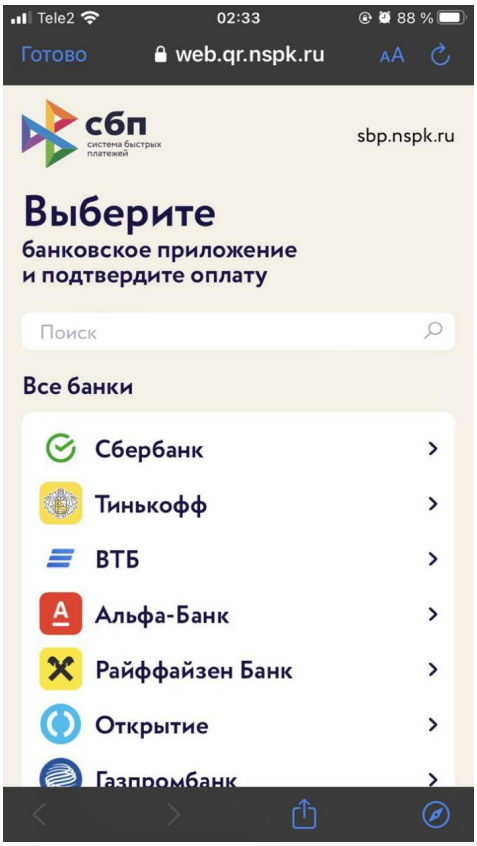
```
https://qr.nspk.ru/792140884478170715042738893644 35?type=01&bank=12345&sum=10000&cur=RUB&crc=F3D0
```

до формата:

```
https://web.qr.nspk.ru/7921408844781707150427388936443 5?type=01&bank=12345&sum=10000&cur=RUB&crc=F3D0
```

Таким образом Ваш клиент перейдет на веб-страницу с выбором банка, который нужно использовать при оплате

Далее производится переход в приложение выбранного банка



3. Получение списка номенклатуры

Получение списка id товаров, ранее добавленных в терминал, для дальнейшей регистрации QR-код

GET /users/qrt/products/

HEADER PARAMETERS

X-Api-Key (required, string)

RESPONSES

▼ 200 Успешный запрос

Тело ответа

```
{
    "id": "f1fccfff-06a9-44c2-8c07-7622f20b0e43",
    "name": "Продукт №1",
    "price": 123,
    "isdeleted": false
}
```

Тестовые данные

▼ **Пример запроса**

```
curl -X GET "https://app.devwapiserv.qrm.ooo/users/qrt/products" -H  "accept: application/json" -H  "X-Api-Key: bpkNMc19.gCrEPy
nUpwBsiujn7ZiYeTCmtgJkXVj8" -H  "X-CSRFToken: g8QNhCg1rnzZktCDbjsA29ty3p05NiTLxRSHNVZZsfcHLj6Khf7VTTQ1a4DLfgb4"
```

▼ **Ожидаемый ответ**

```
{
    "id": "f1fccfff-06a9-44c2-8c07-7622f20b0e43",
    "name": "Продукт №1",
    "price": 123,
    "isdeleted": false
}
```

4. Получение статуса оплаты

Жизненный цикл полученного QR-кода: после создания его становится невозможно оплатить в 2 случаях:

1. после успешно совершенной оплаты,
2. после истечения времени его жизни, которое задаётся в настройках QR-терминала.

Команды “деактивировать QR-код” нет.

4.1 По веб-сокету

Пользовательский сценарий:

1. Формирование QR-кода
2. Коннект по веб-сокету
3. Отправка команды по запросу статуса
4. Ожидание ответа по веб-сокету
5. Отмена запроса по операции, в зависимости от ответа
6. Закрытие веб-сокета

Шаги для получения статуса:

1. Клиент инициирует web-socket запросом по адресу wss://<адрес сервиса>/ws/qr-code?api_key=<api key>. Происходит открытие веб-сокета и устанавливается постоянное соединение.
2. Клиент отправляет сообщение на получения статуса qr-кода.

   ▼ Запрос

   ```
   {
        "command": "get_qr_status",
         "body": {
                "operation_uuid": <operation_uuid>
         }
   }
   ```

   ▼ Ответ

   По истечению времени опроса статуса или изменения статуса сервер отдаёт ответ:

   ```
   {
         "results": {
                "operation_status_code": <код>,
                "operation_status_msg": <описание>
                "operation_uuid": <operation_uuid>
         }

   }
   ```

   ▼ Список статусов операции

Enum в Operation

Aa Статус	Описание	Код
Время ожидания истекло	Таймаут ожидания ответа от банка	0
В процессе	Оплата по операции не проводилась	4
Оплачено	Платёж совершен	5
Отозван	Платеж не прошёл	6
Возврат	Возврат денежных средств	7
Просрочено	Время жизни QR-кода истекло	8

▼ Ошибки

```
{
      "errors": "Данный запрос запрещен, операция в статусе <operation.status>" (Разрешено только для операций в статусе CR
EATED)
}
```

```
{
       "errors": "Не верны входные параметры: <описание ошибки сериализатора>"
}
```

```
{
       "errors": "Операция не обнаружена с uuid: <operation_uuid>"
}
```

```
{
       "errors": error(we didn't receive status from third party service, so there may be problems)
}
```

3. Чтобы отменить операцию по которой был раннее сделан запрос можно воспользоваться следующей командой:

   Данной командой советуем пользоваться при таких сценариях:
   1. Операция не будет оплачена
   2. При получении конечного статуса операции:
      2.1 - “Время ожидания истекло” = “0”
      2.2 - “Оплачено” = “5”
      2.3 - “Отменен” = “6”
      2.4 - “Возврат” = “7”
      2.5 - “Просрочено” = “8”

   ▼ Запрос

```
{
     "command": "cancel",
      "body": {
             "operation_uuid": <operation_uuid>
      }
}
```

   В ответе вернется текущий статус операции, который был зафиксирован в БД

   ▼ Ответ

```
{
      "results": {
             "operation_status_code": 4
              "operation_status_msg": "В процессе"
      }
}
```

▼ **Пример URL веб-сокета с тестовым API-ключом**

```
wss://app.devwapiserv.qrm.ooo/ws/qr-code?api_key=lDvAIB7E.K02zjE1o7P1ZiwDQi5yC4EBrP71Fr8Js
```

4. При обрыве интернет-соединения

   Клиент должен повторно открыть вебсокет и повторить последнюю выполненную команду или отправить команду cancel.

Пример реализации запросов по веб-сокету

```
import 'dart:async';
import 'dart:convert';

import 'package:qrm_frontend/api_client.dart';
import 'package:web_socket_channel/io.dart';
import 'package:web_socket_channel/web_socket_channel.dart';

enum QrOperationStatus { paid, aborted, time_is_up }

const _connectionErrorText = "Не удалось установить соединение с сервером";

const _commandLabel = "command";
const _connectedLabel = "connected";
const _cancelLabel = "cancel";
const _getQrStatusLabel = "get_qr_status";
const _operationStatusCodelabel = "operation_status_code";
const _resultsLabel = "results";
const _errorsLabel = "errors";
const _operationIdLabel = "operation_uuid";
const _bodyLabel = "body";

class QrWebSocket {
  IOWebSocketChannel connection;
  final String apiKey;
  final String operationId;
  final Function({bool isSuccess, String error}) onListen;

  QrWebSocket(this.apiKey, this.operationId, this.onListen);

  bool isClosed = false;

  // Команда запрос на отмену соединения по вебсокету
  String get _jsonForCancelRequest => jsonEncode({
        _commandLabel: _cancelLabel,
        _bodyLabel: {_operationIdLabel: this.operationId}
      });
  //Команда запрос на получение статуса операции
  //Один раз за соединение отправляем и ждем ответа
  String get _jsonForOperationStatusRequest => jsonEncode({
        _commandLabel: _getQrStatusLabel,
        _bodyLabel: {_operationIdLabel: this.operationId}
      });

  //Создание соединения и запуск общения
  void connectAndStartWebSocket() {
    if (!isClosed) {
      connection = IOWebSocketChannel.connect(Uri.parse(
          'wss://app.devwapiserv.qrm.ooo/ws/qr-code?api_key=$apiKey'));
      start();
    }
  }

  //Отправка запроса на получение статуса операции
  void _addOperationStatusRequest() {
    connection.sink.add(_jsonForOperationStatusRequest);
  }

  //Если вебсокет не был закрыт, то закрываем его
  void close() {
    if (!isClosed) {
      isClosed = true;
      connection.sink.add(_jsonForCancelRequest);
      connection.sink.close();
    }
  }

  //Старт общения
  void start() {
    //Отправка запроса статуса
    _addOperationStatusRequest();
```

```
    connection.stream.listen(
        //Функция обработки сообщений от вебсокета
        _onWebSocketListen,
        //При завершении вебсокета, если он был закрыт не по причине 1000, то заново открываем соединение
        onDone: () {
      //1000 - код нормального закрытия вебсокета
      if (connection.closeCode != null && connection.closeCode != 1000) {
        Future.delayed(Duration(seconds: 1), connectAndStartWebSocket);
      }
    },
        //Обработка ошибки
        //если ошибка критическая - нет сокета по такому урлу, то пробрасываем ошибку дальше
        //иначе пробуем заново открыть соединение
        onError: (e) {
      Object error = e;
      if (error is WebSocketChannelException &&
          error.message != null &&
          error.message.contains("not upgraded to websocket")) {
        isClosed = true;
        onListen(error: "Ошибка соединения с сервером.");
      } else
        Future.delayed(Duration(seconds: 1), connectAndStartWebSocket);
    });
  }

  //Обработка сообщений, полученных с вебсокета
  void _onWebSocketListen(message) {
    //Если сообщение существует, то происходит его декодирование
    if (message != null && !isClosed) {
      final _message = jsonDecode(message);
      final results = _message[_resultsLabel];
      //Если в результате декодирования в json содержится поле result, то происходит обработка результата
      if (results != null) {
        _handleResult(results);
      } else
      //Если в результате декодирования в json НЕ содержится поле errors, то отправляется еще один запрос на получение статуса опер
ации
      if (_message[_errorsLabel] == null)
        _addOperationStatusRequest();
      else {
        //Если в результате декодирования в json содержится поле errors, то происходит обработка ошибки
        onListen(error: _message[_errorsLabel]);
      }
    }
  }

  void _handleResult(results) {
    //Если результат представляем собой стрингу, то это может быть только в одном случае - сервер отправил информацию об успешном п
одключении
    if (results is String) {
      if (results.toLowerCase() == _connectedLabel)
        _addOperationStatusRequest();
      else {
        onListen(error: _connectionErrorText);
      }
    } else
    //Если результат представляет собой словарь, то проверям поле, содержащее в себе статус код операции и обрабатываем этот статус
    if (results[_operationStatusCodelabel] != null) {
      close();
      int statusNumber = results[_operationStatusCodelabel];
      switch (statusNumber) {
        //Оплачено
        case 5:
          onListen(isSuccess: true);
          break;
        //Платеж не прошел
        case 6:
          onListen(error: "Платёж не прошёл");
          break;
        //Таймаут
        case 0:
          onListen(error: "Таймаут ожидания ответа от банка");
          break;
        default:
          onListen(error: "Получен неизвестный статус");
          break;
      }
    }
    //Пришло непредвиденное сообщение - отправляем запрос на статус еще раз
    else {
      _addOperationStatusRequest();
    }
  }
}
```

4.2 По REST API

Предпочтительным остаётся способ получения статуса по веб-сокету для уменьшения количества запросов на сервер. Данный метод существует для тех клиентов, которые по какой-то технической причине не могут реализовать веб-сокет.

GET /operations/{id}/qr-status/

HEADER PARAMETERS

X-Api-Key (required, string)

PATH PARAMETERS

id (required, string) - UUID операции, который был получен вместе с QR-кодом в результате запроса **POST** /operations/qr-code/.

RESPONSES

▼ 200 Успешный запрос

Тело ответа

```
{
      "results": {
            "operation_status_code": <код>,
            "operation_status_msg": <статус>
      }

}
```

▼ Список статусов операции

Enum в Operation

Aa Статус	Описание	Код
Создан	Операция создана	3
В процессе	Ожидание оплаты	4
Оплачено	Платёж совершен	5
Отменен	Платеж не прошёл	6
Просрочено	Время жизни QR-кода истек	8
Время ожидания истекло	Таймаут ожидания ответа от банка	0

▼ 400 Запрос завершился ошибкой

```
{
"detail": "Описание ошибки."
}
```

Тестовые данные

▼ **Пример запроса**

```
curl -X GET "https://app.devwapiserv.qrm.ooo/operations/310852c0-945e-4e59-aabe-bda80d8b408e/qr-status/" -H  "accept: application/json" -H  "X-Api-Key: lDvAIB7E.K02zjE1o7P1ZiwDQi5yC4EBrP71Fr8Js" -H  "X-CSRFToken: hMD9gr7XeSDaI9aaJJmOW8DZr3qgLXXN9rQxTfUrOUwzTj12GhaD9TV58ybFrx1S"
```

▼ **Ожидаемый ответ**

```
{
  "results": {
    "operation_status_code": 5,
    "operation_status_msg": "Оплачено"
```

```
  }
}
```

5. Получение фискального чека

Описание

Данные эндпойнты необходимо использовать, только если ответственное лицо согласилось с условиями формирования фискального документа с использованием возможностей ООО "КуАрМенеджер" - Агента ОФД при создании соответствующего QR-терминала в мобильном приложении QRManager. Вся ответственность за правильное указание сведений и своевременную отправку лежит на юридическом лице, принимающем средства от Покупателей.

Получение ЭФД двухэтапное: сначала нужно отправить данные покупки на регистрацию ЭФД, а позже по callback придёт статус формирования ЭФД. Либо, если на клиенте нет возможности получать callback - сделать запрос на статус. При успешной фискализации вместе со статусом придёт ссылка на ЭФД, уже зафиксированный в ОФД и ФНС. Структура данных для фискализации зависит от того, клиентом какого ОФД является ваша компания. В настоящий момент доступны АТОЛ и OFD.ru.

5.1. Регистрация ЭФД

Описание

Эндпойнт позволяет зарегистрировать операцию с указанным номером в ОФД. В результате запроса данные QR-терминала и покупки будут переданы на валидацию в ОФД. Возвращает сообщение об успешной отправке чека на регистрацию.

Все данные из покупки будут не только попадут в фискальный документ и будут отправлены покупателю на почту, но их также можно будет найти в удобном виде на сайте QRManager по адресу

```
www.qrm.ooo/p/{number}/
```

где {number} - номер операции, полученный вместе с QR-кодом в результате запроса **POST** /operations/qr-code/. Пример чека: https://qrm.ooo/d/6550-6158-3901-4696/

⚠️ Запрос необходимо выполнять НЕ РАНЕЕ получения статуса paid по веб-сокету или ответа на запрос по REST API, т.к. нет смысла регистрировать в ОФД покупку, оплата которой ещё не произошла.

Эндпойнт

🌐 **POST** /operations/{number}/receipt/

HEADER PARAMETERS

X-Api-Key (required, string), Content-Type (application/json)

PATH PARAMETERS

number (required, string) - номер операции, который был получен вместе с QR-кодом в результате запроса **POST** /operations/qr-code/.

REQUEST BODY SCHEMA

application/json

REQUEST BODY

```
{
    // JSON с данными о покупке, которые запрашивает ОФД
}
```

Пояснения к полям

Структура данных для фискализации зависит от того, клиентом какого ОФД является ваша компания. Она подробно описана в документации каждого ОФД ниже.

▼ АТОЛ

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/d863255c-02d9-4a19-8c54-8af8a1d98aef/____.pdf

▼ OFD.ru

Актуальная документация с пункта "**3.1. Формирование кассового чека**" и до пункта "**4. Описание ошибок**".

Ferma для разработчиков - API сервиса для интернет-магазина | OFD.ru

В документе приводятся технические сведения о программном интерфейсе приложений (API) предоставляющем возможность регистрировать онлайн-кассы и инициировать генерацию кассовых документов посредством информационной системы (ИС) "Ferma". Кассы в сервисе

https://ofd.ru/razrabotchikam/ferma#%D1%84%D0%BE%D1%80%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%BA%D0%B0%D1%81%D1%81%D0%BE%D0%B2%D0%BE%D0%B3%D0%BE_%D1%87%D0%B5%D0%BA%D0%B0

⚠️ Ниже описаны поля, которые отличаются от документации ОФД. Они первичны для протокола REST API QRM и отменяют (либо дополняют) описание тех же полей в документации ОФД.

Специфические для QRM REST API пояснения к полям 1 (1)

Aa Название поля	Описание	Пример значения
external_id или InvoiceId	В это поле обязательно нужно поместить number данной операции, полученный вместе с QR-кодом в результате запроса **POST** /operations/qr-code/. Формат number: "XXXX-XXXX-XXXX-XXXX". Без указания этого поля ЭФД сформирован не будет.	5352-4777-6325-1692
payment_address или BillAddress	Поле будет перезаписано на сервере QRM вне зависимости от того, какое значение будет помещено в него в теле запроса. В поле будет помещен физический адрес QR-терминала, с которым тот был зарегистрирован в банке, обслуживающим организацию клиента.	"Кофейня BestCoffee: Москва, Московская, 1, офис 1"
vats.sum	Если вы клиент АТОЛ, рекомендуем рассчитывать сумму НДС по формуле round(operation.sum / (100 + qrt.vat) * qrt.vat, 0), где operation.sum - сумма за позицию в покупке, qrt.vat - целое число, обозначающее ставку НДС. 0, 10 или 20.	

RESPONSES

▼ 200 Успешный запрос

Тело ответа

```
{
  "results": {
    "status": "success"
  }
}
```

▼ 400 Запрос завершился ошибкой

Тело ответа

```
{
  "detail": "<сообщение из поля text ответа об ошибке из АТОЛ>"
}
```

Перечень возможных ошибок:

```
"Ошибка формирования электронного чека. Операция не обнаружена."
"Ошибка формирования электронного чека. Операция уже имеет зарегистрированный чек"
"Ошибка формирования электронного чека. Неверно настроен ОФД."
"Ошибка формирования электронного чека. Ошибка доступа к сервису ОФД."
"Ошибка формирования электронного чека. Истекло время ожидания сервиса ОФД."
"Ошибка формирования электронного чека. Ошибка соединения с ОФД."
"Ошибка формирования электронного чека, полученная с сервиса ОФД: текст ошибки"
"Ошибка формирования электронного чека. Итоговая сумма фискализации не совпадает с суммой заказа."
"Ошибка сервиса: текст ошибки"
```

▼ 404 Операция ещё не оплачена

```
{
  "detail": "Operation number: {number} не обнаружена"
}
```

Где {number} - номер операции вида "XXXX-XXXX-XXXX-XXXX"

Тестовые данные

Чтобы протестировать фискализацию были созданы дополнительные QR-терминалы, подключенные каждый к своему ОФД. С этими ключами необходимо создать новые операции и перевести их в состояние "Оплачено" запросом **GET** /operations/{id}/qr-status/ (Явный запрос статуса операции необходим для тестирования фискализации только в тестовой среде. В продуктивной среде операция будет оплачена, когда Вы отсканируете сформированный QR-код и оплатите его).

🔑 **Тестовый ключ QRT под АТОЛ: 2lVRjLhR.2zoJKwE0zNWMRjSWO0DybEcKB7Q80iWk**

⚠️ **Сумма в АТОЛ передается в рублях**

Примеры тел запросов для клиентов АТОЛ

▼ Вариант 1. Запрос с минимальными данными о покупке

⚠️ Внимание, ИНН компании для тестовых запросов АТОЛ всегда должен быть 5544332219.

```
{
  "external_id": "номер вашей операции",
  "timestamp": "10.11.2022 20:10:00",
  "receipt": {
    "client": {
      "email": "user@mail.ru"
    },
    "company": {
      "sno": "osn",
      "email": "user@mail.ru",
      "inn": "5544332219",
      "payment_address": "г Москва, Ленинградское шоссе, д 34 к 2, кв 50"
    },
    "items": [
      {
        "name": "тест",
        "price": 500,
        "quantity": 1,
        "sum": 500,
        "payment_object": "commodity"
      }
    ],
    "payments": [
      {
        "type": 1,
        "sum": 500
      }
    ],
    "vats": [
      {
```

```
        "type": "vat20",
        "sum": 80
      }
    ],
    "total": 500,
    "cashier": "Василий Цветков"
  }
}
```

```
curl -X POST "https://app.devwapiserv.qrm.ooo/operations/7786-0666-4734-9487/receipt/" -H  "accept: application/json" -H
 "X-Api-Key: 2IVRjLhR.2zoJKwE0zNWMRjSWO0DybEcKB7Q80iWk" -H  "Content-Type: application/json" -H  "X-CSRFToken: xeHbtmLJu7T6
WRPr7pddDqAOsZQ6nI6EvfLQXSNV7KvSVcX6ncFNTFo1xU9YLwOy" -d "{    \"external_id\": \"7786-0666-4734-9487\",  \"timestamp\":
 \"10.11.2022 20:10:00\",  \"receipt\": {    \"client\": {      \"email\": \"user@mail.ru\"    },    \"company\": {
 \"sno\": \"osn\",      \"email\": \"user@mail.ru\",      \"inn\": \"5544332219\",      \"payment_address\": \"г Москва, Ле
нинградское шоссе, д 34 к 2, кв 50\"    },    \"items\": [      {        \"name\": \"тест\",        \"price\": 500,
 \"quantity\": 1,        \"sum\": 500,        \"payment_object\": \"commodity\"      }    ],    \"payments\": [      {
\"type\": 1,        \"sum\": 500      }    ],    \"vats\": [      {        \"type\": \"vat20\",        \"sum\": 80      }
],    \"total\": 500,    \"cashier\": \"Василий Цветков\"  }}"
```

Полученный тестовый ЭФД: https://consumer.1-ofd-test.ru/ticket?t=20221110T2018&s=500.00&fn=9999078900006242&i=92450&fp=1019638328&n=1

▼ Вариант 2. Запрос с подробным описанием позиций покупки из документации АТОЛ

Является примером структуры. Использованный в запросе, может выдать ошибку валидации содержания некоторых полей.

▼ Вариант 3. Запрос с максимальными данными о покупке

Является примером структуры. Использованный в запросе, может выдать ошибку валидации содержания некоторых полей.

🔑 **Тестовый ключ QRT под OFD.ru: wql0hiV4.rZLWWkLR68PS8Lwfe3CWHsgZobGhgICq**

⚠️ **Сумма в OFD.ru передается в рублях**

Примеры тел запросов для клиентов OFD.ru

▼ Вариант 1. Запрос с минимальными данными о покупке

```
{
  "Request": {
    "Inn": "5544332219",
    "Type": "Income",
    "InvoiceId": "номер вашей операции",
    "LocalDate": "2022-11-10T19:51:24",
    "CustomerReceipt": {
      "TaxationSystem": "Common",
      "Email": "user@mail.ru",
      "PaymentType": 1,
      "KktFA": false,
      "BillAddress": "г Москва, Ленинградское шоссе, д 34 к 2, кв 50",
      "Items": [
        {
          "Label": "Ветеринарные услуги",
          "Price": 500,
          "Quantity": 1,
          "Amount": 500,
          "Vat": "Vat20",
          "PaymentMethod": 1
        }
      ],
      "PaymentItems": [
        {
          "PaymentType": 1,
          "Sum": 500
        }
      ]
    },
    "Cashier": {
      "Name": "Василий Цветков"
    }
```

```
  }
}
```

```
curl -X POST "https://app.devwapiserv.qrm.ooo/operations/4154-2308-1431-6192/receipt/" -H  "accept: application/json" -H  "X-Ap
i-Key: wqI0hiV4.rZLWWkLR68PS8Lwfe3CWHsgZobGhgICq" -H  "Content-Type: application/json" -H  "X-CSRFToken: xeHbtmLJu7T6WRPr7pddDq
AOsZQ6nI6EvfLQXSNV7KvSVcX6ncFNTFo1xU9YLwOy" -d "{    \"Request\": {    \"Inn\": \"5544332219\",    \"Type\": \"Income\",    \"I
nvoiceId\": \"4154-2308-1431-6192\",    \"LocalDate\": \"2022-11-10T19:51:24\",    \"CustomerReceipt\": {      \"TaxationSystem
\": \"Common\",      \"Email\": \"user@mail.ru\",      \"PaymentType\": 1,      \"KktFA\": false,      \"BillAddress\": \"г Моc
ква, Ленинградское шоссе, д 34 к 2, кв 50\",      \"Items\": [        {          \"Label\": \"Ветеринарные услуги\",
 \"Price\": 500,          \"Quantity\": 1,          \"Amount\": 400,          \"Vat\": \"Vat20\",          \"PaymentMethod\": 1
}      ],      \"PaymentItems\": [        {          \"PaymentType\": 1,          \"Sum\": 500        }      ]    },    \"Cashi
er\": {      \"Name\": \"Василий Цветков\"    }  }}"
```

Полученный тестовый ЭФД: https://check-demo.ofd.ru/rec/9999078902001735/187579/757868555

5.2. Получение статуса и ссылки на ЭФД

Статус заказа и ЭФД возможно получить как с помощью callback с нашего сервера, так и вручную сделав REST API вызов.

Callback

Описание

Это рекомендованный способ получения статуса и ЭФД и обычно, с ним клиент может быстрее получить ответ. В JSON-структуре, которую вы отправляете на фискализацию в АТОЛ или OFD.ru есть поле "callback_url" или "CallbackUrl". Для получения статуса по callback подставьте URL, на который будете ожидать ответ, в соотвествующее поле.

Эндпойнт

Описание

Эндпойнт возвращает статус обработки фискального документа в ОФД и URL этого документа, если он был успешно сформирован.

Требуется некоторое время для формирования чека в ОФД, поэтому рекомендуем делать запрос 300 миллисекундами позже запроса на регистрацию ОФД. Важно, что информация о статусе в OFD.ru имеет срок годности (порядка суток), после которого перестает быть доступна.

🌐 **GET** /operations/{number}/receipt/status/

HEADER PARAMETERS

X-Api-Key (required, string)

PATH PARAMETERS

number (required, string) - номер операции, который был получен вместе с QR-кодом в результате запроса **POST** /operations/qr-code/.

REQUEST BODY SCHEMA

application/json

RESPONSES

▼ 200 Успешный запрос

Тело ответа

```
{
  "status" : "done",
  "ofd_receipt_url": "https://consumer.1-ofd-test.ru/ticket?t=20211124T2050&s=87419.00&fn=9999078900006279&i=52590&fp=2561555433&r
  "error" : ""
}
```

Обработка ошибок в поле error состоит в том, что необходимо их исправлять и повторно отправлять запросы **POST** /operations/<number>/receipt/, пока не вернётся ответ об успешной фискализации данной операции и URL ЭФД. Если после нескольких попыток добиться успешной фискализации так и не получилось, обратитесь в QRManager по почте dev@qrm.ooo или на горячую линию поддержки, т.к. в противном случае операция не будет зафиксирована в налоговой службе.

Пояснения к полям

Aa Название поля	Тип	Описание	Примеры значений
status	string	Статус	wait, done, fail. 1) wait - ЭФД всё ещё валидируется или в очереди на обработку. 2) done - ЭФД успешно зарегистрирован в налоговой. При таком статусе поле ofd_receipt_url не пустое, а error - пустое. 3) fail - ЭФД не был зарегистрирован. При таком статусе поле error не пустое, а ofd_receipt_url - пустое
ofd_receipt_url	string (URL)	Ссылка на ЭФД	https://check-demo.ofd.ru/rec/9999078902001735/80980/4278894019
error	string	Описание ошибки валидации данных покупки из ОФД	[-3975] Некорректное значение параметров команды ФН

▼ 400 Запрос завершился ошибкой

Тело ответа

```
{
  "detail": "<Описание ошибки>"
}
```

Перечень возможных ошибок:

```
"Ошибка формирования электронного чека. Операция не обнаружена."
"Ошибка формирования электронного чека. Операция уже имеет зарегистрированный чек"
"Ошибка формирования электронного чека. Неверно настроен ОФД."
"Ошибка формирования электронного чека. Ошибка доступа к сервису ОФД."
"Ошибка формирования электронного чека. Истекло время ожидания сервиса ОФД."
"Ошибка формирования электронного чека. Ошибка соединения с ОФД."
"Ошибка формирования электронного чека, полученная с сервиса ОФД: текст ошибки"
"Ошибка формирования электронного чека. Итоговая сумма фискализации не совпадает с суммой заказа."
"Ошибка сервиса: текст ошибки"
```

🌍 Кассовые ссылки СБП

Кассовая ссылка СБП - это разновидность статического QR-кода с возможностью перезаписи параметров платежа. Это альтернатива одноразовым динамическим QR-кодам.

Как применять кассовую ссылку в бизнесе?

Компания может распечатать изображение QR-кода, по которому при покупке будет задаваться сумма платежа и другие парамеры. Бизнесу больше не нужно инвестировать в устройства для отображения динамического QR-кода или создавать ссылки и распечатывать QR-код для каждого товара по отдельности.

Как используется кассовая ссылка?

QR код (QR изображение кассовой ссылки) всегда наклеен на кассе. Используется реальная наклейка, либо распечатка. Изображение кода всегда одинаковое. Состояние ссылки по умолчанию – «деактивирована».

При выборе способа оплаты «СБП», клиентское приложение обращается к методу «активация кассовой ссылки». В период активности кассовой ссылки её можно сфотографировать смартфоном с последующим открытием банковского приложения плательщика, либо сфотографировать приложением СБПэй.

⚠️ Принимается только одна оплата активированной кассовой ссылки

Из банковского приложения или приложения СБПэй производится оплата.

Клиентскому приложению поступают сведения об изменении статуса QR кода.

Клиентское приложение обращается к методу «деактивация кассовой ссылки».

Заказ закрывается.

Последовательность запросов REST API

1. Регистрация кассовой ссылки (Кассовая ссылка регистрируется только один раз для каждой точки!)

   💡 **POST** /operations/cash-qr/

2. Активация кассовой ссылки

   💡 **POST** /operations/cash-qr/{cashQrId}/activate/

3. Запрос статуса платежа

   По websocket

   💡 wss://{URL сервиса QRM}/ws/qr-code/?api_key={api key}

   ИЛИ по REST API

   💡 **GET** /operations/{id}/cash-qr/status/

4. Деактивация кассовой ссылки

   💡 **DELETE** /operations/cash-qr/{cashQrId}/deactivate/

⚠️ Для реализации и тестирования методов кассовой ссылки в тестовой среде необходимо использовать другой API key:
🔑 7m9EkxnW.vJ9ROl7Bl7pjKxp1e1n9Gw9IyV6M46M1

⚠️ В версиях QRM REST API 2.6 и ниже невозможно оплатить операцию, которая будет создана при активации кассовой ссылки в тестовой среде. Можно проверить её статус, который должен вернуть значение “Создан”. Для полноценного тестирования, пожалуйста, воспользуйтесь в продуктовой среде ключом API, который Вы сгенерировали в приложении QRManager.

1. Регистрация Кассовой ссылки СБП

🌐 **POST** /operations/cash-qr/

HEADER PARAMETERS

X-Api-Key (required, string)

REQUEST BODY SCHEMA

application/json

REQUEST BODY

```
{
    "size": 400
    "qrc_id": "AD10007F3VC9B6VF9O08QVO10VSQRKJM"
}
```

Пояснения к полям (4)

Aa Название поля	Обязательность	Тип	Описание	Диапазон допустимых значений
size	необяз.	integer	Размер QR-кода в px. Если оставить тело запроса пустым, то в ответе вернётся QR-код 400x400px.	Min = 100px, Max =1000px
qrc_id	необяз.	string	Идентификатор Кассовой ссылки, выданной в банке, если его нету, то заполнять ничего не нужно	1 … 140

RESPONSES

▼ 200 Успешный запрос

Тело ответа

```
{
    "cashQrId": "AS1000670LSS7DN18SJQDNP4B05KLJL2",
    "cashQrLink": "https://qr.nspk.ru/AS1000670LSS7DN18SJQDNP4B05KLJL2?type=01&bank=100000000001&sum=10000&cur=RUB&crc=C08B",
    "cashQrCode": "https://wapiserv.qrm.ooo/media/qr_codes/qr_code_6a14291f-df22-4541-8a17-34752907f12b.png",
    "status": "CREATED"
}
```

Пояснения к полям

Название поля	Тип	Описание
cashQrId	string	UUID кассовой ссылки в БД QRM.
cashQrIdURL		изображения QR-кода заданного размера, хранящегося на сервере QRM до истечения заданного ttl или факта оплаты
cashQrLink	URL	Кассовая ссылка от СБП
status	string	Статус регистрации Платежной или Информационной ссылки СБП

▼ 400 Запрос завершился ошибкой

Тело ответа

Ошибка, связанная полями тела запроса:

```
{
  "size": [
    "Size must be >= 100 and <= 1000"
  ]
}
```

Ошибка, не связанная полями тела запроса:

```
{
  "non_field_errors": [
    "Текст ошибки"
  ]
}
```

Прочие ошибки:

```
{
  "detail":"Текст с описанием ошибки"
}
```

Тестовые данные

▼ **Пример запроса**

```
curl -X POST "https://app.devwapiserv.qrm.ooo/operations/cash-qr/" -H  "accept: application/json" -H  "X-Api-Key: 7m9EkxnW.vJ9R
Ol7Bl7pjKxp1e1n9Gw9IyV6M46M1" -H  "Content-Type: application/json" -H  "X-CSRFToken: RHD0nU9a1anhjsd0grBnFuA776lgMdNZ0y8IyAsUUI
rRiVbOBfM4jZ5nn0xVcGzW" -d "{  \"size\": 300}"
```

▼ **Ожидаемый ответ**

```
{
  "results": {
    "cashQrId": "BS1R003CN9P4NJ0F8QQA41JQRG1U4U44",
    "cashQrLink": "https://qr.nspk.ru/BS1R003CN9P4NJ0F8QQA41JQRG1U4U44?type=01&bank=1crt88888881&crc=24BB",
    "cashQrCode": "https://app.devwapiserv.qrm.ooo/media/qr_codes/qr_code_BS1R003CN9P4NJ0F8QQA41JQRG1U4U44.png",
    "status": "CREATED"
  }
}
```

2. Активация Кассовой ссылки СБП

⚠️ **Время жизни Кассовой ссылки минимум 5 минут, максимум 20 минут, данное значение берется из настроек QR-терминала**

🌐 **POST/operations/cash-qr/{cashQrId}/activate/**

HEADER PARAMETERS

X-Api-Key (required, string)

REQUEST BODY SCHEMA

application/json

PATH PARAMETERS

{cashQrId} - ID кассовой ссылки полученный в результате запроса **POST** /operations/cash-qr/

REQUEST BODY

```
{
  "sum": 600,
  "payment_purpose": "Оплата по договору №123454"
}
```

Пояснения к полям (3)

Aa Название поля	Обязательность	Тип	Описание	Значение по умолчанию	Диапазон допустимых значений
sum	обяз.	integer	Сумма за всю покупку в копейках		0 ... 1 000 000 00
payment_purpose	необяз.	string	Назначение платежа		1 … 140

RESPONSES

▼ 200 Успешный запрос

Тело ответа

```
{
    "cashQrParamsId": "AS1000670LSS7DN18SJQDNP4B05KLJL2",
    "number": "23DF-3456-G3GG-634K",
    "operation_Id": "3fa85f64-5717-4562-b3fc-2c963f66afa6"
}
```

Название поля	Тип	Описание
cashQrParamsId	string	Идентификатор активных значений параметров Кассовой ссылки СБП
number	string	Уникальный номер заказа в формате XXXX-XXXX-XXXX-XXXX из букв и цифр. Необходим для фискализации данной операции
operation_Id	string	UUID операции. Необходим для получения статуса оплаты QR-кода по веб-сокету

▼ 400 Запрос завершился ошибкой

Ошибка, связанная полями тела запроса:

```
{
  "sum": [
    "Описание ошибки"
  ],
  "qrTtl": [
    "Описание ошибки"
  ]
}
```

Ошибка, не связанная полями тела запроса:

```
{
  "non_field_errors": [
    "Текст ошибки"
  ]
}
```

Прочие ошибки:

```
{
  "detail":"Текст с описанием ошибки"
}
```

Тестовые данные

▼ **Пример запроса**

```
curl -X POST "https://app.devwapiserv.qrm.ooo/operations/cash-qr/BS1R003CN9P4NJ0F8QQA41JQRG1U4U44/activate/" -H  "accept: appli
cation/json" -H  "X-Api-Key: 7m9EkxnW.vJ9ROl7Bl7pjKxp1e1n9Gw9IyV6M46M1" -H  "Content-Type: application/json" -H  "X-CSRFToken:
 E08HCjoVmCvfQWEYImQi7BvS3EfjStdoOKlctQ1XMF9392DHQDMAfR6azYoyDsMZ" -d "{  \"sum\": 600}"
```

▼ **Ожидаемый ответ**

```
{
  "results": {
    "cashQrParamsId": "BP10007TBU1RQ11M9IC8648U21STFNNU",
    "number": "3106-5144-5361-8625",
    "operation_Id": "470e3479-0144-4fe0-9268-1675cf2d1960"
  }
}
```

3. Запрос статуса платежа

🌐 **GET/operations/{id}/cash-qr/status/**

HEADER PARAMETERS

X-Api-Key (required, string)

REQUEST BODY SCHEMA

application/json

PATH PARAMETERS

{id}(operation_Id) - ID кассовой ссылки полученный в результате запроса **GET** /operations/cash-qr/

RESPONSES

▼ 200 Успешный запрос

Тело ответа

```
{
  "results": {
    "status": 6
  }
}
```

Название поля	Тип	Описание статуса
status	string	“Создано” - 3 “Платеж прошел” - 5 ”Платеж не прошел” - 6 ”Отменен” - 7 ”Просрочено” - 8 ”Таймаут ожидания ответа от банка” - 0

4. Деактивация Кассовой ссылки СБП

⚠️ При успешной оплате кассовой ссылки, она деактивируется автоматически!

🌐 **DELETE/operations/cash-qr/{cashQrId}/deactivate/**

HEADER PARAMETERS

X-Api-Key (required, string)

REQUEST BODY SCHEMA

application/json

PATH PARAMETERS

{cashQrId} - ID кассовой ссылки полученный в результате запроса **GET** /operations/cash-qr/

RESPONSES

▼ 200 Успешный запрос

```
{
  "results": {
    "status": "OK"
  }
}
```

▼ 400 Запрос завершился ошибкой

Тело ответа

```
{
  "detail":"Текст с описанием ошибки"
}
```

Тестовые данные

▼ **Пример запроса**

```
curl -X DELETE "https://app.devwapiserv.qrm.ooo/operations/cash-qr/BS1R003CN9P4NJ0F8QQA41JQRG1U4U44/deactivate/" -H  "accept: application/json" -H  "X-Api-Key: 7m9EkxnW.vJ9ROl7Bl7pjKxp1e1n9Gw9IyV6M46M1" -H  "X-CSRFToken: E08HCjoVmCvfQWEYImQi7BvS3EfjStdoOKlctQ1XMF9392DHQDMAfR6azYoyDsMZ"
```

▼ **Ожидаемый ответ**

```
{
  "results": {
    "status": "OK"
  }
}
```

🌎 Возврат средств по операции

Позволяет вернуть частично или полностью сумму покупки на счёт покупателя.

⚠️ Для реализации и тестирования методов возврата в тестовой среде необходимо использовать другой API key:
🔑 7m9EkxnW.vJ9ROl7Bl7pjKxp1e1n9Gw9IyV6M46M1
По этому ключу нужно сформировать QR-код в методе “POST /operations/qr-code/”
на сумму 1000180коп., после чего выполнить запрос на возврат, по этой операции.

1. Запрос на возврат по Операции

🌐 **POST** /operations/refund/

HEADER PARAMETERS

X-Api-Key (required, string)

REQUEST BODY SCHEMA

application/json

REQUEST BODY

```
{
  "operation_id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "amount": 1000180
}
```

Название поля	Тип	Описание
operation_id	string	ID операции, полученный в результате запроса QR-кода или кассовой ссылки
amount	int	Сумма в копейках, которую нужно вернуть. Должна быть ≤ суммы возвращаемой операции. **В тестовой сумма возврата всегда будет 1000180коп.**

RESPONSES

▼ 200 Успешный запрос

Тело ответа

```
{
   "results":{
      "refundRequestId":"BR10000B34KU3CDJ95U9T42NQ7DN0VA0"
```

```
   }
}
```

Название поля	Тип	Описание
refundRequestId	string	ID запроса на возврат, который необходимо использовать в запросе статуса возврата.

▼ 400 Запрос завершился ошибкой

Тело ответа

Ошибка, связанная полями тела запроса:

```
{
  "operation_id": [
    "Описание ошибки"
  ],
  "amount": [
    "Описание ошибки"
  ]
}
```

Ошибка, не связанная полями тела запроса:

```
{
  "non_field_errors": [
    "Текст ошибки"
  ]
}
```

Прочие ошибки:

```
{
  "detail":"Текст с описанием ошибки"
}
```

Тестовые данные

▼ **Пример запроса**

```
curl -X POST "https://app.devwapiserv.qrm.ooo/operations/refund/" -H  "accept: application/json" -H  "X-Api-Key: 7m9EkxnW.vJ9RO
l7Bl7pjKxp1e1n9Gw9IyV6M46M1" -H  "Content-Type: application/json" -H  "X-CSRFToken: CUs1Vo6rmGuQzr4TtqvYxU3DgE3vf9eJJY18ARambFh
7NhWaOlOGXh29aTqwGfAu" -d "{  \"operation_id\": \"3c58bb1b-eba7-4821-b286-2cb2d7ffea5c\",  \"amount\": 1000180}"
```

▼ **Ожидаемый ответ**

```
{
   "results":{
      "refundRequestId":"BR10000TGJDMIT0397FAKQIH4V3M8E1F"
   }
}
```

2. Статус запроса на возврат средств

🌐 **GET** /operations/{id}/refund-status/{refundRequestId}/

HEADER PARAMETERS

X-Api-Key (required, string)

REQUEST BODY SCHEMA

application/json

PATH PARAMETERS

{refundRequestId} - ID запроса на возврат полученный в результате запроса **POST** /operations/refund/

RESPONSES

▼ 200 Успешный запрос

Тело ответа

```
{
   "results":{
      "status": "success",
      "message":"Согласие Банка Плательщика на проведение платежа"
   }
}
```

Название поля	Тип	Описание	Допустимые значения
status	string	Результат операции	success, fail, received
message	string	Ответ СБП	

▼ 400 Запрос завершился ошибкой

Тело ответа

```
{
  "detail":"Текст с описанием ошибки"
}
```

▼ 404 Неверный id операции или id запроса на возврат

Тело ответа

```
{
  "detail":"Страница не найдена."
}
```

Тестовые данные

▼ **Пример запроса**

```
curl -X GET "https://app.devwapiserv.qrm.ooo/operations/3c58bb1b-eba7-4821-b286-2cb2d7ffea5c/refund-status/BR10000TGJDMIT0397FA
KQIH4V3M8E1F/" -H  "accept: application/json" -H  "X-Api-Key: 7m9EkxnW.vJ9ROl7Bl7pjKxp1e1n9Gw9IyV6M46M1" -H  "X-CSRFToken: CUs1
Vo6rmGuQzr4TtqvYxU3DgE3vf9eJJY18ARambFh7NhWaOlOGXh29aTqwGfAu"
```

▼ **Ожидаемый ответ**

```
{
   "results":{
      "status":"Success",
      "message":"Успешно"
   }
}
```

🌍 Программа лояльности BenefittY

BenefittY - это кросс-платформенный рекомендательный сервис, который помогает вашим клиентам освободиться от пластиковых карт, пользоваться скидками и отзывами друзей, а также информирует о всех скидках, акциях и событиях вашей организации!

После регистрации в программе лояльности и подключении торговых точек, можно пользоваться функциями начисления/списания/скидки вашим клиентам при оплате по СБП.

1. Поиск/регистрация клиента в программе лояльности

1.1 Поиск клиента в программе лояльности

🔑 **Тестовый ключ: fia98afA.YoBPKERQJTukRFZk75WvFqVvmAECVyqw**

🌐 **POST /loyalty/search/**

HEADER PARAMETERS

X-Api-Key (required, string)

REQUEST BODY SCHEMA

application/json

REQUEST BODY

```
{
  "phone": "+79000000000",
  "sum": 0
}
```

Пояснения к полям (2)

Aa Название поля	Обязательность	Тип	Описание	Диапазон допустимых значений
phone	обяз.	string	Номер телефона клиента	
sum	обяз.	integer	Сумма за всю покупку в копейках	0 ... 1 000 000 00

RESPONSES

⚠️ **Данный ответ вернется в случае, если клиент зарегистрирован в программе лояльности**

▼ 200 Успешный запрос

Тело ответа

```
{
  "results": {
    "client_id": 256672,
    "discount_card_id": 129719,
    "card_level": "BenefittY Start",
    "max_discount": 25,
    "max_points": 1,
    "all_points": 492,
    "accrual_points": 0,
    "accrual_available": true,
    "write_off_available": true,
    "needs_code": false
  }
}
```

Пояснения к полям

Aa Название	Описание	Тип
client_id	id клиента	integer
discount_card_id	id скидочной карты	integer
card_level	Уровень карты, в соответствии с настройками программы лояльности	string
max_discount	Максимально допустимая скидка по присланной в запросе сумме	string
max_points	Максимально доступное количество баллов для списания	integer
all_points	Общее количество баллов клиента	integer

Aa Название	Описание	Тип
accrual_points	Количество баллов, которые будут начислены, если покупатель выберет начисление	integer
accrual_available	Возможность начисления баллов	boolean
write_off_available	Возможность списания баллов	boolean
needs_code	Необходимость кода подтверждения для списания баллов	boolean

Тестовые данные

▼ **Пример запроса**

```
curl -X POST "https://app.devwapiserv.qrm.ooo/loyalty/search/" -H  "accept: application/json" -H  "X-Api-Key: fia98afA.YoBPKERQ
JTukRFZk75WvFqVvmAECVyqw" -H  "Content-Type: application/json" -H  "X-CSRFToken: YxCMPAsvvKjU3WlMyMPFBtksJhEdNnnJ6hrm0QIclYJ3eV
cYYLRuzmnvwmA03R4J" -d "{  \"phone\": \"+79000000000\",  \"sum\": 100000}"
```

▼ **Ожидаемый ответ**

```
{
  "results": {
    "client_id": 256672,
    "discount_card_id": 129719,
    "card_level": "BenefittY Start",
    "max_discount": 25,
    "max_points": 1,
    "all_points": 492,
    "accrual_points": 0,
    "accrual_available": true,
    "write_off_available": true,
    "needs_code": true
  }
}
```

⚠️ **Данный ответ вернется в случае, если клиент не зарегистрирован в программе лояльности**

▼ 200 Успешный запрос

Тело ответа

```
{
  "results": {
    "confirmation_code": true
  }
}
```

Пояснения к полям

Aa Название	Описание	Тип
confirmation_code	отправление кода подтверждения	boolean

Тестовые данные

▼ **Пример запроса**

```
curl -X POST "https://app.devwapiserv.qrm.ooo/loyalty/search/" -H  "accept: application/json" -H  "X-Api-Key: fia98afA.YoBPKERQ
JTukRFZk75WvFqVvmAECVyqw" -H  "Content-Type: application/json" -H  "X-CSRFToken: YxCMPAsvvKjU3WlMyMPFBtksJhEdNnnJ6hrm0QIclYJ3eV
cYYLRuzmnvwmA03R4J" -d "{  \"phone\": \"+79000000000\",  \"sum\": 100000}"
```

▼ **Ожидаемый ответ**

```
{
  "results": {
    "confirmation_code": true
  }
}
```

2. Формирование QR-кода с программой лояльности

⚠️ В зависимости от выбранной при регистрации программы лояльности, Вашим клиентам будут доступны разные виды бонусов:
1. списание/начисление
2. скидка
3. списание/начисление + скидка

⚠️ **Важно!**
Поле “**loyalty_type”** - является определяющим фактором при создании QR-кода с программой лояльности**.**
Типы лояльности**:**
”0” - **списание**
“1” - **начисление**
”2” - **скидка**
”0” **- списание + скидка**
”1” **- начисление + скидка**

⚠️ При формировании QR-кода в поле:
”**sum**” - указывается конечная сумма(с учетом всех скидок)
”**base_amount**” - указывается изначальная сумма
Если сумма остается неизменной, то поле ”**base_amount**” можно не указывать

2.1 Формирование QR-кода (начисление)

🌐 **POST /operations/qr-code/**

HEADER PARAMETERS

X-Api-Key (required, string)

REQUEST BODY SCHEMA

application/json

REQUEST BODY

```
{
  "sum": 10000,
  "qr_size": 400,
  "payment_purpose": "Назначение платежа",
  "loyalty": {
    "phone": "+79000000000",
    "client_id": 257065,
    "loyalty_type": 1
  }
}
```

Пояснения к полям (6)

Aa Название поля	Обязательность	Тип	Описание

Aa Название поля	Обязательность	Тип	Описание
sum	обяз.	string	Сумма за всю покупку в копейках
qr_size	необяз.	integer	Размер QR-кода в пикселях
payment_purpose	необяз.	string	Назначение платежа
phone	обяз.	string	Номер телефона клиента
client_id	обяз.	integer	id клиента
loyalty_type	обяз.	integer	Тип лояльности: 0-списание, 1-начисление, 2-скидка

▼ 200 Успешный запрос

Тело ответа

```
{
  "results": {
    "operation_id": "3df4cff6-0392-4f39-8055-92ba48f22898",
    "number": "8760-1393-8085-6214",
    "qr_link": "https://qr.nspk.ru/811796256201983190178592005565 81?type=01&bank=12345&sum=10000&cur=RUB&crc=F3D0",
    "qr_img": "https://app.devwapiserv.qrm.ooo/media/qr_codes/qr_code_811796256201983190178592005565 81.png"
  }
}
```

Пояснения к полям

Aa Название	Описание	Тип
qr_img	URL изображения QR-кода заданного размера, хранящегося на сервере QRM до истечения заданного ttl или факта оплаты	URL
qr_link	Платёжная ссылка от СБП	URL
number	Уникальный номер заказа в формате XXXX-XXXX-XXXX-XXXX из букв и цифр. Необходим для фискализации данной операции	string
operation_id	UUID операции. Необходим для получения статуса оплаты QR-кода по веб-сокету	string

Тестовые данные

▼ **Пример запроса**

```
curl -X POST "https://app.devwapiserv.qrm.ooo/operations/qr-code/" -H  "accept: application/json" -H  "X-Api-Key: fia98afA.YoBP
KERQJTukRFZk75WvFqVvmAECVyqw" -H  "Content-Type: application/json" -H  "X-CSRFToken: cjdwVxtZBv0m1WyFShTPs4yg73kXkQYs2ApUC1t7lZ
qDdale2UphpuJ4oc6xcCRP" -d "{  \"sum\": 10000,  \"qr_size\": 400,  \"payment_purpose\": \"Назначение платежа\",  \"loyalty\": {
\"phone\": \"+79000000000\",    \"client_id\": 257065,    \"loyalty_type\": 1  }}"
```

▼ **Ожидаемый ответ**

```
{
  "results": {
    "operation_id": "3df4cff6-0392-4f39-8055-92ba48f22898",
    "number": "8760-1393-8085-6214",
    "qr_link": "https://qr.nspk.ru/811796256201983190178592005565 81?type=01&bank=12345&sum=10000&cur=RUB&crc=F3D0",
    "qr_img": "https://app.devwapiserv.qrm.ooo/media/qr_codes/qr_code_811796256201983190178592005565 81.png"
  }
}
```

2.2 Формирование QR-кода (списание)

🌐 **POST /operations/qr-code/**

HEADER PARAMETERS

X-Api-Key (required, string)

REQUEST BODY SCHEMA

application/json

REQUEST BODY

```
{
  "sum": 90000,
  "qr_size": 400,
  "payment_purpose": "Назначение платежа",
  "loyalty": {
    "phone": "+79000000000",
    "client_id": 257065,
    "loyalty_type": 0,
    "loyalty_amount": 100,
    "base_amount": 100000
  }
}
```

Пояснения к полям (10)

Aa Название поля	Обязательность	Тип	Описание
sum	обяз.	string	Сумма за всю покупку в копейках
qr_size	необяз.	integer	Размер QR-кода в пикселях
payment_purpose	необяз.	string	Назначение платежа
phone	обяз.	string	Номер телефона клиента
client_id	обяз.	integer	id клиента
loyalty_type	обяз.	integer	Тип лояльности(0-списание, 1-начисление, 2-скидка)
loyalty_amount	обяз.	integer	Сумма баллов для списания
base_amount	обяз.	integer	Сумма товаров без скидки

▼ 200 Успешный запрос

Тело ответа

```
{
  "results": {
    "operation_id": "ef2d9b6b-d704-45d1-bdac-c3cebbaa2930",
    "number": "5915-1021-2094-6201",
    "qr_link": "https://qr.nspk.ru/3993870490107199334103638366530 4?type=01&bank=12345&sum=90000&cur=RUB&crc=F3D0",
    "qr_img": "https://app.devwapiserv.qrm.ooo/media/qr_codes/qr_code_3993870490107199334103638366530 4.png"
  }
}
```

Пояснения к полям

Aa Название	Описание	Тип
qr_img	URL изображения QR-кода заданного размера, хранящегося на сервере QRM до истечения заданного ttl или факта оплаты	URL
qr_link	Платёжная ссылка от СБП	URL
number	Уникальный номер заказа в формате XXXX-XXXX-XXXX-XXXX из букв и цифр. Необходим для фискализации данной операции	string
operation_id	UUID операции. Необходим для получения статуса оплаты QR-кода по веб-сокету	string

Тестовые данные

▼ **Пример запроса**

```
curl -X POST "https://app.devwapiserv.qrm.ooo/operations/qr-code/" -H  "accept: application/json" -H  "X-Api-Key: fia98afA.YoBP
KERQJTukRFZk75WvFqVvmAECVyqw" -H  "Content-Type: application/json" -H  "X-CSRFToken: CEhoRv6ny5VLcJVtzhRO1TbDqtHJgu74sBqPgYPbpq
```

```
j1KTW0unITUDiIZZFHIioc" -d "{  \"sum\": 90000,  \"qr_size\": 400,  \"payment_purpose\": \"Назначение платежа\",  \"loyalty\": {
\"phone\": \"+79000000000\",    \"client_id\": 257065,    \"loyalty_type\": 0,    \"loyalty_amount\": 100,    \"base_amount\":
 100000   }}"
```

▼ **Ожидаемый ответ**

```
{
  "results": {
    "operation_id": "ef2d9b6b-d704-45d1-bdac-c3cebbaa2930",
    "number": "5915-1021-2094-6201",
    "qr_link": "https://qr.nspk.ru/3993870490107199334103638366530 4?type=01&bank=12345&sum=90000&cur=RUB&crc=F3D0",
    "qr_img": "https://app.devwapiserv.qrm.ooo/media/qr_codes/qr_code_399387049010719933410363836653 04.png"
  }
}
```

2.3 Формирование QR-кода (скидка)

POST /operations/qr-code/

HEADER PARAMETERS

X-Api-Key (required, string)

REQUEST BODY SCHEMA

application/json

REQUEST BODY

```
{
  "sum": 95000,
  "qr_size": 400,
  "payment_purpose": "Назначение платежа",
  "loyalty": {
    "phone": "+79000000000",
    "client_id": 257065,
    "loyalty_type": 2,
    "base_amount": 100000
  }
}
```

Пояснения к полям (8)

Aa Название поля	Обязательность	Тип	Описание
sum	обяз.	string	Сумма за всю покупку в копейках
qr_size	необяз.	integer	Размер QR-кода в пикселях
payment_purpose	необяз.	string	Назначение платежа
phone	обяз.	string	Номер телефона клиента
client_id	обяз.	integer	id клиента
loyalty_type	обяз.	integer	Тип лояльности: 0-списание, 1-начисление, 2-скидка
loyalty_amount	обяз.	integer	Сумма баллов для списания
base_amount	обяз.	integer	Сумма товаров без скидки

▼ 200 Успешный запрос

Тело ответа

```
{
  "results": {
    "operation_id": "1cb9117f-a8b8-4309-8d34-918160dd970d",
    "number": "4083-8240-8339-0344",
    "qr_link": "https://qr.nspk.ru/830497213993550008046609214208 11?type=01&bank=12345&sum=95000&cur=RUB&crc=F3D0",
```

```
    "qr_img": "https://app.devwapiserv.qrm.ooo/media/qr_codes/qr_code_830497213993550008046609214208 11.png"
  }
}
```


Пояснения к полям

Aa Название	Описание	Тип
qr_img	URL изображения QR-кода заданного размера, хранящегося на сервере QRM до истечения заданного ttl или факта оплаты	URL
qr_link	Платёжная ссылка от СБП	URL
number	Уникальный номер заказа в формате XXXX-XXXX-XXXX-XXXX из букв и цифр. Необходим для фискализации данной операции	string
operation_id	UUID операции. Необходим для получения статуса оплаты QR-кода по веб-сокету	string

Тестовые данные

▼ **Пример запроса**

```
curl -X POST "https://app.devwapiserv.qrm.ooo/operations/qr-code/" -H  "accept: application/json" -H  "X-Api-Key: fia98afA.YoBP
KERQJTukRFZk75WvFqVvmAECVyqw" -H  "Content-Type: application/json" -H  "X-CSRFToken: tg2nsfWNJquRkoMOSDKNuhnMNPE6fAx2QegFnFi7HL
yxoEwRzMz5tjKMMH8GOJSo" -d "{  \"sum\": 95000,  \"qr_size\": 400,  \"payment_purpose\": \"Назначение платежа\",  \"loyalty\": {
\"phone\": \"+79000000000\",    \"client_id\": 257065,    \"loyalty_type\": 2,    \"base_amount\": 100000  }}"
```

▼ **Ожидаемый ответ**


```
{
  "results": {
    "operation_id": "1cb9117f-a8b8-4309-8d34-918160dd970d",
    "number": "4083-8240-8339-0344",
    "qr_link": "https://qr.nspk.ru/830497213993550008046609214208 11?type=01&bank=12345&sum=95000&cur=RUB&crc=F3D0",
    "qr_img": "https://app.devwapiserv.qrm.ooo/media/qr_codes/qr_code_830497213993550008046609214208 11.png"
  }
}
```