ООО "ЦЕНТР ПРОГРАММНЫХ ИННОВАЦИЙ"

RetailProxy

API для подключения сторонних систем

Оглавление

Общие понятия
Установка RetailProxy
Подключение к API
API Endpoint
Авторизация
- Параметры Digest Auth
- Детальный процесс
  1. Инициализация запроса
  2. Расчет клиентской части
     - **2.1. Исходные данные:**
     - **2.2. Расчет HA1:**
     - **2.3. Расчет HA2:**
     - **2.4. Генерация клиентских параметров:**
     - **2.5. Расчет итогового Response:**
  3. Повторный запрос с авторизацией
  4. Серверная проверка
  5. Логирование на стороне сервера

Параметры запросов на получение данных
Возвращаемые данные
Лицензия
Пользователи
Прайсы
Группы продаж
Товары
Скидки
Кассовые сервера
- Подключение TitanPOS Mobile
  - Подключение TitanPOS Mobile к RetailProxy
- Подключение CashServer

Кассовые документы
- Платежный документ
- Сменный итог продаж (Z-отчёт)
- Документы внесения, изъятия и выдачи
- Документ аннулирования

Общие понятия

RetailProxy - это веб-сервер, принимающий HTTP-запросы от клиентов, обрабатывающий их и отправляющий ответы с запрошенными данными. Позволяет связать товароучетную систему с мобильной кассой TitanPOS Mobile. Предоставляет функционал загрузки и выгрузки следующих данных: лицензий, кассовых серверов, кассовых документов, групп продаж, справочника товаров, скидок, кассиров. Представляет собой файл RetailProxy.exe, который необходимо установить в среде Windows как службу, с указанием порта для приема HTTP запросов, для обеспечения автоматического запуска и работы в фоновом режиме.

Для работы RetailProxy требуется установка СУБД MongoDB.
В качестве хранилища данных используется MongoDB - распределенная база данных общего назначения. Рекомендуемая версия для установки - не ниже 4.2. При установке СУБД локально не требуются дополнительные действия, установка производится стандартным для системы образом по умолчанию.

Установка RetailProxy

Установите RetailProxy в качестве службы. Предварительно необходима установка MongoDB.

```
@echo off

sc create RetailProxy binPath= "%~dp0RetailProxy.exe -PORT=3333 -DEBUG=true"
start= auto
sc config RetailProxy depend= MongoDB
net start RetailProxy
```

Подключение к API

Точки подключения к API выглядят следующим образом

`<BaseUrl>/api/<DataType>` - для подключения к RetailProxy API

Где:

BaseUrl - адрес сервера, например http://localhost:3333
DataType - запрашиваемый тип данных, он же endpoint, например: Goods, SalesGroup, Discount, CashDocuments

Пример строки подключения:

```
http://localhost:3333/api/CashServers
```

Все параметры являюстя регистрозависимыми.

Запрос типа GET используется получение данных, запрос типа POST - для отправки данных.

API Endpoint

Endpoint / DataType	Описание	Методы
Licenses	Изменения ключа для подключения стороннего ПО	GET, POST
User	Добавление кассиров, пользователей	GET, POST
CashServers	Добавление кассовых серверов	GET, POST
CashDocuments	Кассовые документы	GET
Prices	Прайсы	GET, POST
SalesGroup	Группы продаж	GET, POST
Goods	Товары	GET, POST
Discount	Скидки	GET, POST

Авторизация

RetailProxy API использует **Digest Authentication** (RFC 2617) для защиты доступа к ресурсам. Это метод аутентификации, при котором пароль никогда не передается по сети в открытом виде, а используется хэширование MD5 для подтверждения знаний пароля.

Параметры Digest Auth

Параметр	Описание	Пример
`realm`	Область аутентификации	"RetailProxy"
`nonce`	Уникальная строка сервера	Генерируется сервером
`qop`	Качество защиты	"auth" (аутентификация)
`nc`	Счетчик запросов	"00000001" (в hex)
`cnonce`	Клиентский nonce	Генерируется клиентом
`response`	Вычисленный хэш ответа	c669650854abd56e4c5024421fb12061
`opaque`	Непрозрачный токен	Клиент должен вернуть его без изменений

Детальный процесс

1. Инициализация запроса

Клиент → Сервер:

```
GET /api/Goods HTTP/1.1
Host: localhost:3333
User-Agent: PostmanRuntime/7.51.1
Accept: */*
Accept-Encoding: gzip
Connection: close
```

Что здесь важно:

- Нет заголовка `Authorization` - сервер сразу понимает, что нужна аутентификация.
- Тело запроса может содержать данные, но сервер их игнорирует до авторизации.
- Сервер поддерживает сжатие ответа в формате **gzip**, стоит использовать это для уменьшения размера передаваемых данных.
- `Connection`: RetailProxy **всегда закрывает соединения** после обработки запроса, независимо от значения заголовка от клиента.

Сервер → Клиент (ответ 401):

```
HTTP/1.1 401 Unauthorized
Content-Type: text/html
Date: YYYY-MM-DDThh:mm:ss+03:00
Server: RetailProxy/26.01.05-0617
Supported-Features: gzip
WWW-Authenticate: Digest
    realm="RetailProxy",
    qop="auth",
    algorithm="MD5",
    nonce="0h9m7q16jhtpskd9xgsqqm40aeqwo2ctf8",
    opaque="967z9amtu3d6ulukh1ogdxm6bgp8t4wq"
Content-Length: 0
Connection: close
```

2. Расчет клиентской части

Клиент получает challenge и должен вычислить `response` по следующему алгоритму:

2.1. Исходные данные:

```
username = "BackOffice" // стандартное название лицензии
realm = "RetailProxy"
password = "29fae49393a05"  // ваш пароль
method = "GET"
uri = "/api/Goods"
nonce = "0h9m7q16jhtpskd9xgsqqm40aeqwo2ctf8" //прислал сервер для расчетов
opaque="967z9amtu3d6ulukh1ogdxm6bgp8t4wq" //прислал сервер для передачи в
неизменном виде
qop = "auth"
```

2.2. Расчет HA1:

```
HA1 = MD5(username:realm:password)
    = MD5("BackOffice:RetailProxy:29fae49393a05")
```

Результат: `3f7dcc4c4442b5a4bb11c35314b53f2a`

2.3. Расчет HA2:

```
HA2 = MD5(method:uri)
    = MD5("GET:/api/Goods")
```

Результат: `a671dd1299ad6767a29fcf903e34c5c3`

2.4. Генерация клиентских параметров:

```
nc = "00000001"      // Nonce Count (счетчик, увеличивается с каждым запросом)
cnonce = "G4QRCfBh"  // Client Nonce (уникальная строка, генерируется клиентом)
```

2.5. Расчет итогового Response:

```
Response = MD5(HA1:nonce:nc:cnonce:qop:HA2)
    = MD5("3f7dcc4c4442b5a4bb11c35314b53f2a:
          0h9m7q16jhtpskd9xgsqqm40aeqwo2ctf8:
          00000001:
          G4QRCfBh:
          auth:
          a671dd1299ad6767a29fcf903e34c5c3")
```

Результат: `c669650854abd56e4c5024421fb12061`

3. Повторный запрос с авторизацией

Клиент отправляет тот же запрос, но с заголовком `Authorization`:

Клиент → Сервер:

```
GET /api/Goods HTTP/1.1
Host: localhost:3333
User-Agent: PostmanRuntime/7.51.1
Accept: */*
Accept-Encoding: gzip
Connection: close
Authorization: Digest username="BackOffice", realm="RetailProxy",
nonce="0h9m7q16jhtpskd9xgsqqm40aeqwo2ctf8", uri="/api/Goods", algorithm="MD5",
qop=auth, nc=00000001, cnonce="G4QRCfBh",
response="c669650854abd56e4c5024421fb12061",
opaque="967z9amtu3d6ulukh1ogdxm6bgp8t4wq"
```

Важно: Параметр `opaque` возвращается **без изменений**, как получен от сервера.

4. Серверная проверка

Сервер выполняет аналогичные вычисления и проверяет:

```
# 1. Поиск пользователя в БД
"Find one" Collection=Licenses Filter="map[Id:Application BackOffice]"

# 2. Вычисление HA1 (с паролем из БД)
"MD5(username:realm:userpass)" HA1=23a6af47478dbda1d2456c97a709c57b

# 3. Вычисление HA2
"MD5(Method:RequestURI)" HA2=a671dd1299ad6767a29fcf903e34c5c3

# 4. Вычисление ожидаемого Response
"MD5(HA1:nonce:nc:cnonce:qop:HA2)" Response=c669650854abd56e4c5024421fb12061

# 5. Сравнение: вычисленный Response == полученный от клиента
```

При совпадении: `HTTP/1.1 200 OK` с запрошенными данными

```
{"Status":"OK","Data":[{"Barcode":"4811114561112",...]}
```

При несовпадении: `HTTP/1.1 401 Unauthorized`

5. Логирование на стороне сервера

В логах RetailProxy можно отследить:

```
// ПЕРВЫЙ ЗАПРОС: Клиент пробует доступ без авторизации
"Request processed" Code=401 Latency=0s Method=GET RemoteAddr="[::1]:55286"
path=/api/Goods
// ↑ Сервер: "Требуется авторизация!" (отправляет WWW-Authenticate с nonce)

// ВТОРОЙ ЗАПРОС: Клиент повторяет с Digest-заголовком
"Incoming request" Method=GET RemoteAddr="[::1]:55288" path=/api/Goods
// ↑ Клиент: "Вот мои credentials" (новый порт = новое соединение)

// ПРОВЕРКА 1: Ищем лицензию в БД
"Find one" Collection=Licenses Filter="map[Id:Application BackOffice]"
// ↑ Сервер: "Существует ли пользователь BackOffice?"

// ПРОВЕРКА 2: Вычисляем HA1 (проверка пароля)
"MD5(username:realm:userpass)" HA1=23a6af47478dbda1d2456c97a709c57b

// ПРОВЕРКА 3: Вычисляем HA2 (проверка целостности запроса)
"MD5(Method:RequestURI)" HA2=a671dd1299ad6767a29fcf903e34c5c3
// ↑ Сервер: "Клиент не менял запрос? HA2 = MD5(GET:/api/Goods)"

// ПРОВЕРКА 4: Вычисляем Response (итоговая проверка)
"MD5(HA1:nonce:nc:cnonce:qop:HA2)" Response=c669650854abd56e4c5024421fb12061
// ↑ Сервер: "Клиент знает пароль? Response должен совпасть"

// БИЗНЕС-ЛОГИКА: Готовим запрос данных
"Find data" CompanyId= Conditions="[]" Filter="map[]" Limit=0 OrderBy="[]" Skip=0
Type=Goods
// ↑ Сервер: "Аутентификация ОК, готовлю запрос товаров без фильтров"

// ЗАПРОС К БД: Получаем данные
```

```
"Find many" Collection=Goods Filter="map[]" Order="[]"
// ↑ Сервер: "Ищу все товары в MongoDB"

// РЕЗУЛЬТАТ: Успех!
"Request processed" Code=200 Latency=7.3443ms Method=GET RemoteAddr="[::1]:55288"
path=/api/Goods
// ↑ Сервер: "Готово! 200 OK, данные отправлены за 7.3мс"
```

Важно: В логах отображается HA1, вычисленный сервером на основе пароля из БД. Он не совпадает с расчетами клиента.

Параметры запросов на получение данных

Limit - Количество документов в выдаче (максимально - 100)

ClearData - вернуть "чистые данные". Если параметр присутствует, то возвращается только значение поля **Data** (см. ниже)

OrderBy - сортировка, Modified: сортировать по полю Modified (в момент создания или изменения сервер записывает timestamp изменения)

Condition - условия для выбора данных, нельзя использовать одно и тоже поле дважды. Примеры условий:

```
Condition=StatusFlag,$eq,2&Condition=Timestamp,$gte,1782896434443618423
Получить фискализированные документы изменённые после времени в системе UNIX

Condition=Lines.Goods.Barcode,$eq,48109017
Получить документы содержащие товары с указанным штрихкодом

Condition=SerialId,$eq,AVQ24159990026&Condition=Position,$gte,2026-01-
06T10:00:00.000
Получить все документы от токена с серийным номером AVQ24159990026 начиная с
2026-01-06T10:00:00.000
```

Общий синтаксис работы с условиями выбора данных следующий:

Condition=Field,Comparison,Value
Условия объединяются логическим AND (&)
Условия для сравнения:

$eq	**равно**
$ne	**не равно**
$gt, **$gte**	**больше, больше либо равно**
$lt, **$lte**	**меньше, меньше либо равно**

Используются следующие приведения типов данных в уловиях Condition запроса:

Функция	Тип данных	Входные значения	Пример использования

Bool()	boolean	`true`: 1, t, T, true, TRUE, True
		`false`: 0, f, F, false, FALSE, False
			`Condition=Deleted,$eq,Bool(false)`
			`Condition=IsDeposit,$eq,Bool(T)`
			`Condition=Refund,$eq,Bool(T)`

Int()	Int64	Целочисленные значения	`Condition=StatusFlag,$gte,Int(2)`

Digit()	Double	Числа с плавающей точкой	`Condition=Quantity,$eq,Int(100)`
			`Condition=Price,$eq,Digit(199.99)`
			`Condition=TotalSum,$gte,Digit(1000.50)`
			`Condition=DiscountSum,$lt,Digit(15.75)`

Пример запроса для получения платёжных документов, созданных после указанного времени в системе UNIX:

```
http://localhost:3333/api/CashDocuments?
ClearData&OrderBy=Modified&Condition=StatusFlag,$eq,2&Condition=Type,$eq,Check&Co
ndition=Timestamp,$gte,1782896434443618423
```

Пример запроса для получения платёжных документов определённой кассы, созданных за период:

```
http://localhost:3333/api/CashDocuments?
Condition=StatusFlag,$eq,2&Condition=Type,$eq,Check&Condition=SerialId,$eq,AVQ241
70201111&Condition=Position,$gte,2026-07-
01T09:34:37.000&Condition=Timestamp,$lte,1782896434443618423
```

Возвращаемые данные

Коды возврата API:

200 - запрос прошел валидацию
400 - в запросе есть ошибки

Возвращаемые данные
По-умолчанию возвращается следующий json:

```
{
   "Status": "OK/ERR",
   "Data": "Данные",
   "Message": "Сообщение"
}
```

Если в параметрах запроса присутствует параметр **ClearData**, то ответ будет содержать только значение поля Data

Status: OK - запрос обработан успешно. В этом случае будет присутствовать поле Data с количеством записанных объектов или с jsonArray в случае выгрузки, ERR - в процессе обработки запроса возникли ошибки, текст которых будет приведен в поле Message

Data: Массив данных в формате JSON

Message: Текст ошибки

Лицензия

Коллекция Licenses, предназначена для изменения пароля для авторизации по API из внешних программ. После первого подключения можно изменить Password через POST запрос на endpoints /Licenses. `http://localhost:3333/api/Licenses`

```
[
    {
        "Serial": "BackOffice", //в качестве Serial всегда должен быть BackOffice - другие
                                //подключения игнорируются
        "Key": "пароль"
    }
]
```

Пользователи

Прежде всего для добавления кассовых серверов, необходимо создать кассиров, которые будут распределены и закреплены за кассовыми серверами (TitanPOS Mobile). Отправляется POST запрос на endpoint **User**, который заканчивается названием типа данных, либо необходимыми параметрами и(или) условиями запроса, а телом запроса является json, который содержит эти данные, в теле запроса должен передаваться массив элементов.
`http://localhost:3333/api/User`

Примечание: поле `"RoleFlag"` отвечает за права пользователя, по умолчанию для всех пользователей, кроме `"admin"`, значение равно 2 - кассир, 1 - администратор. Кассир имеет права для выполнения любой кассовой операции. Для версии RetailProxy старше 26.01.05: Не изменять пароль пользователя `"admin"`, т.к. при изменении пароля, в поле `"RoleFlag"` установится значение 2. Вернуть значение 1, можно только через прямое взаимодействие с БД, через Mongo Compass или Mongosh.

**Справочная информация о значениях `RoleFlag`*

Значение	Роли
1	Админ
2	Кассир
3	Админ, Кассир
4	Старший кассир

Значение	Роли
5	Админ, Старший кассир
6	Кассир, Старший кассир
7	Админ, Кассир, Старший кассир

Пример отправляемых данных:

```
[
    {
        "Id" : "cashier",       //Id необходимый для привязки к кассовому серверу
        "Deleted" : false,      //признак удаления, логический
        "Name" : "Иванов",      //имя кассира отображаемое на чеке
        "Password" : "1111"     //пароль для входа кассира
    },
...
```

Прайсы

Это документ, с использованием которого мы можем распределить товары или услуги, а также их цены, для реализации в каждом конкретном торговом объекте.

Отправляется запрос на endpoint **Prices**, который заканчивается названием типа данных, либо необходимыми параметрами и(или) условиями запроса, а телом запроса является json, который содержит эти данные, в теле запроса должен передаваться массив элементов.
`http://localhost:3333/api/Prices`

```
{
    "Id": "1",                          //идентификатор
    "Deleted": false,                   //признак удаления, логический
    "Memo": "Пр-т Независимости 58/22", //примечание
    "Name": "Магазин Товары"            //наименование прайса
},
```

Группы продаж

Отправляется запрос на endpoint **SalesGroup**, который заканчивается названием типа данных, либо необходимыми параметрами и(или) условиями запроса, а телом запроса является json, который содержит эти данные, в теле запроса должен передаваться массив элементов.
`http://localhost:3333/api/SalesGroup`

```
[
    {
        "Deleted" : false,              //признак удаления, логический
        "Departament" : 1,              //номер отдела
        "Id" : "1",                     //код группы продаж
        "Name" : "Молочные продукты"    //наименование группы продаж
    }
]
```

Товары

Отправляется запрос на endpoint **Goods**, который заканчивается названием типа данных, либо необходимыми параметрами и(или) условиями запроса, а телом запроса является json, который содержит эти данные, в теле запроса должен передаваться массив элементов. Ключ Id должен быть уникальным.
`http://localhost:3333/api/Goods`

Обязательные поля: Id, Barcode, Price, Deleted.

```
[
    {
    "Id" : "1",                       //идентификатор товара во внешней учетной
системе
    "Barcode" : "4811111111112",      //штрихкод
    "Country" : "Республика Беларусь", //страна происхождения
    "Departament" : 3,                //номер отдела
    "ExpDate" : "2025-01-01T00:00:00", //срок годности
    "FractionFlag" : 1,               //тип товара (0=товар 1=вес 2=услуга
3=аванс)
    "Manufacturer" : "Савушкин",      //производитель
    "MarkType" : 1,                   //маркировка (1=одна маркировка, 2=две
маркировки)
    "Measure" : "ШТ.",                //единица измерения
    "Name" : "Творог2",               //наименование товара
    "Price" : 2.34,                   //цена
    "RatingFlag" : 5,                 //рейтинг, целое число
    "Sale" : false,                   //признак распродажи, логический
    "SalesGroup" : "1",               //код группы продаж
    "VATrate" : 0.25,                 //допустимое значение НДС (см.CSV формат)
    "VendorCode" : "112233",          //артикул товара
    "Deleted" : false                 //признак удаления, логический
    }
]
```

Товар с использованием прайса.

```
[
    {
    "Id": "2",
    "Barcode": "5010677945004",
    "Country": "",
```

```
    "Deleted": false,
    "Departament": 1,
    "ExpDate": "",
    "FractionFlag": 0,
    "FractionMeasure": "",
    "Manufacturer": "",
    "MarkType": 0,
    "Measure": "шт",
    "Name": "Вино ВЕРМУТ MARTINI ROSATO розовое 15% 1л",
    "Price": 13.89,
    "Prices": [
        {
            "Name": "Магазин Товары",//Название прайса (необязательное)
            "Price": 10,             //Специальная цена прайса
            "PriceId": "1",          //Уникальное Id прайса, из коллекции Prices
            "SpecialPrice": true     //false цена будет 13.89, true цена будет 10
        },
        {
            "Name": "Кафе",
            "Price": 0,
            "PriceId": "2",
            "SpecialPrice": false
        },
        {
            "Name": "Ресторан",
            "Price": 0,
            "PriceId": "3",
            "SpecialPrice": false
        }
    ],
    "RatingFlag": 0,
    "Sale": false,
    "SalesGroup": "",
    "VATrate": 0.2,
    "VendorCode": ""
    }
]
```

При использовании десктопной кассы вы также можете добавить собственные информационные поля — они будут сохранены в карточке товара и, при добавлении этого товара в чек, скопированы в соответствующий элемент массива `Lines` кассового документа.

При использовании мобильной кассы в элементы массива `Lines` попадает ограниченный набор полей: `Id`, `Barcode`, `Name`.

Скидки

Отправляется запрос на endpoint **Discount**, который заканчивается названием типа данных, либо необходимыми параметрами и(или) условиями запроса, а телом запроса является json, который содержит эти данные, в теле запроса должен передаваться массив элементов.
`http://localhost:3333/api/Discount`

```
[
    {
        "Deleted": false, //признак удаления, логический
        "Id" : 1, //внутренний идентификатор скидки
        "Name" : "Молочные продукты", //наименование скидки
        "Rate" : 30 //указание % скидки 30/надбавки -30
    }
]
```

Кассовые сервера

Каждый документ в коллекции `CashServers` является кассовым сервером, представляет из себя отдельную сущность, которая независит от других. На один кассовый сервер можно подключить любое кол-во кассовых аппаратов или локальных веб-серверов CashServer. Можно для каждого кассового аппарата создать отдельный кассовый сервер, например для внутренней логики внешней учетной программы для разделения кассовых документов по местам хранения `"Storage"`. При подключении более 1-ой кассы к кассовому серверу кассовые документы можно отличать друг от друга по полю `"SerialId"`, которое всегда содержит серийный номер токена программной кассы.

Подключение TitanPOS Mobile

Добавление кассового сервера необходимо для подключения TitanPOS Mobile к RetailProxy и выгрузки товаров из RetailProxy в TitanPOS Mobile. Доступны POST, GET запрос на endpoint **CashServers** `http://localhost:3333/api/CashServers`, пример:

```
[
    {
        "Id" : "mobile",                                    //серийный № кассового сервера
        "Deleted" : false,                                  //признак удаления, логический
        "Serials" : "AVQ11169990716 AVQ11169990716",        //серийные № токенов, указываются через пробел
        "Users" : "admin cashier cashier2 cashier3",        //доступные пользователи для кассы
        "Key": "1111",                                      //ключ кассового сервера
        "RetailName": "Магазин №1",                         //название магазина печатается в заголовке чека
        "Address": "Минск, 6-й путепроводный переулок, д.7",    //адрес магазина печатается в заголовке чека
        "DefaultCashiers": [
            {
                "CashierId": "cashier",      //Id пользователя
                "SerialId": "AVQ11169990716" //обязательно серийный номер токена установленного в кассовом аппарате
            }
        ],
        //кассир по умолчанию, позволяет привязать нужного кассира к кассе без возможности его замены на самой кассе
```

```
        "WorkingHours": ["09:00-18:00"],                //время работы
отображается на экране и в X, Z отчётах
        "Storage": "1",                                 //Id места хранения,
присваивается ко всем чекам
        "PriceId": "1"                                  //id нужного прайса, для
распределения товаров
    }
]
```

Примечание:
`"DefaultCashiers"`, `"WorkingHours"`, `"Storage"`, `"PriceId"` - опциональные поля, которые можно использовать при необходимости. `"DefaultCashiers"` - позволяет создать несколько пар "Кассир=Номер СКО" `[{},{},{}]`
`"RetailName"`, `"Address"` - поля для текста в шапке чека, можно использовать только одно, можно оба. Перенос строк происходит автоматически в зависимости от количества символов.

Подключение TitanPOS Mobile к RetailProxy

Необходимо открыть TitanPOS Mobile на мобильной кассе перейти в Настройки - Касса и выбрать режим работы RetailServer касса, затем перейти в Настройки - Retail Server, выбрать ProxyServer и ввести необходимые данные.

Пример:
Адрес сервера (RetailProxy): http://192.168.101.115:3333
Серийный № кассового сервера: mobile
Ключ кассового сервера: 1111

Подключение CashServer

Добавление кассового сервера необходимо для подключения CashServer к RetailProxy и обеспечения обмена данными. Взаимодействовать с CashServer можно как через API, описанное в API CashServer, так и через клиентское приложение TitanRetail. Для настройки обмена необходимо на стороне CashServer создать лицензию в коллекции Licenses, доступно только через TitanRetail, Настройки - Лицензии - Меню - Создать.

Параметр	Значение	Описание
Серийный номер	desktop	Уникальный идентификатор лицензии, должен совпадать с Id CashServers
Наименование	Подключение к RetailProxy	Назначение лицензии, произвольное
Действительна до	01.01.2030	Дата окончания действия лицензии, произвольное
Тип лицензии	Лицензия для подключению к серверу TitanRetail	Тип лицензии

Параметр	Значение	Описание
Ключ	1111	Ключ кассового сервера, должен совпадать с Key CashServers
Доп. параметр	http://localhost:3333/RetailProxy	URL для подключения к RetailProxy. /RetailProxy - остаётся неизменной

Пример тела запроса для создания Кассового сервера в RetailProxy:

```
[
    {
        "Id" : "desktop",                                   //серийный № кассового сервера
        "Deleted" : false,                                  //признак удаления, логический
        "Serials" : "AVQ11169990716 AVQ11169990716",        //серийные № токенов, указываются через пробел
        "Users" : "admin cashier cashier2 cashier3",        //доступные пользователи для кассы
        "Key": "1111",                                      //ключ кассового сервера
        "RetailName": "Магазин №1",                         //название магазина
        "Address": "Минск, 6-й путепроводный переулок, д.7",    //адрес магазина
        "Storage": "1",                                     //Id места хранения, присваивается ко всем чекам
        "PriceId": "1"                                      //id нужного прайса, для распределения товаров
    }
]
```

Примечание:

`"Storage"` - обязательное поле, при его отсутствии CashServer не сможет отправить кассовые документы в RetailProxy.
`"PriceId"`, `"RetailName"`, `"Address"` - опциональные поля, которые можно использовать при необходимости.
`"RetailName"`, `"Address"` - справочная информация для внутреннего использования, не используется в TitanPOS. Шапка чека задаётся через конфигурационный файл `TitanPOS.conf` параметрами: `"CustomOrganizationName" : "String"` , `"DefaultHeader" : "String"`, `"ContactInfo" : "String"`, подробнее в руководстве по установке и администрированию TitanPOS.

Кассовые документы

При запросе кассовых документов **CashDocuments** следует ограничивать выборку необходимым периодом. В качестве элементов массива возвращаемых данных будут получены стандартные кассовы документы TitanPOS (платежные, внесения, изъятия, выдачи, возвраты и аннулирования). `http://localhost:3333/api/CashDocuments?Condition=Position,$gt,2026-01-01`
Далее будут преведены только существенные для товарооборота типы документов и существенные поля из них с необходимыми пояснениями.

Платежный документ

Далее приводится пример платежного документа сформированного в TitanPos Mobile.

```
{
    "Status": "OK",
    "Data": [
        {
            "Author": {
                "Id": "cashier",                        //id кассира
                "Name": "Иванов"                        //имя кассира
            },

            "CashServerVersionOnSave": "2.0.2604021004", //версия CashServer на
момент сохранения документа
            "CashServerVersionOnSend": "2.0.2604021004", //версия CashServer на
момент отправки документа

            "Corrections": [],                          //массив, соответствующий
товарам, по которым были
                                                        //осуществлены коррекции
(товары были удалены
                                                        //из предчека). Формат
данных элемента массива
                                                        //идентичен элементу массива
"Lines"

            "Currency": "BYN",                          //валюта расчета
            "Deleted": false,                           //признак удаления документа
            "DiscountSum": 0,                           //общая сумма скидок

            "FrontInfo": {                              //информация о фронтовом ПО
(ПМСР)
                "Name": "TitanPosMobile",               //наименование фронтового ПО
                "Version": "2.0.2604021004"             //версия фронтового ПО
            },

            "Id": "27D425F7B02EB5F8071835E6",           //идентификатор кассового
документа

            "Lines": [                                  //массив товаров
                {
                    "Departament": 0,                   //отнесение к отделу
                    "DiscountSum": 0,                   //сумма скидки по позиции

                    "Goods": {
                        "Barcode": "4812311000215",//штрихкод товара
                        "Id": "d3038977-7306-40a0-85b1-c8bc37cb19e9",  //
идентификатор товара
                        "Name": "Кола"                  //наименование товара
                    },

                    "Price": 9,                         //цена за единицу товара
```

```
                "Quantity": 1,                  //количество товара
                "TotalSum": 9                   //итоговая сумма по позиции
            }
        ],

        "Modified": {
            "T": 1778831849,                    //время изменения документа
            "I": 1                              //номер изменения документа
        },

        "Number": "33",                         //номер платежного документа

        "Payments": [                           //массив описания примененных
видов оплат
            {
                "Sum": 9,                       //сумма оплаты

                "TypeFlag": 1                   //вид платежа:
                                                //1 - наличными
                                                //2 - безналичными
                                                //3 - прочие виды оплат
            }
        ],

        "Position": "2026-05-15T10:47:55.000", //время закрытия чека

        "Refund": false,                        //false - платежный документ
                                                //true - возврат

        "RetailProxyVersionOnSave": "26.01.05-0617", //версия RetailProxy на
момент сохранения

        "SerialId": "AVQ11171227503",           //серийный номер СКО

        "Signature": "27D425F7B02EB5F8071835E6", //УИД кассового документа

        "StatusFlag": 2,                        //состояние документа:
                                                //1 - открыт
                                                //2 - закрыт
                                                //3 - ожидает закрытия на
сервере
        "Storage": {
            "Id": "1"                           //идентификатор склада
        },

        "Timestamp": "1778831849508751200",     //timestamp документа

        "TotalSum": 9,                          //итого платежного документа

        "Type": "Check",                        //тип кассового документа:
                                                //Check - платежный документ

        "_id": "6a06d1e9daa71bec0da8a42b"       //внутренний идентификатор
документа
    }
]
```

```
}
```

Сменный итог продаж (Z-отчёт)

Далее приводится пример Z-отчёт.

```
{
    "AdvanceSum": 0,                                    //общая сумма реализации
средств
                                                        //авансового платежа
    "Author": {
        "Id": "Benya",
        "Name": "Бенедикт М."
    },
    "CancelCount": 1,                                   //число и сумма отмененных
заказов
    "CancelSum": 3.35,
    "CorrectionCount": 3,                               //число и сумма коррекций
заказов
    "CorrectionSum": 11.06,
    "Currency": "BYN",
    "Deleted": false,
    "DepositCount": 1,                                  //количество и сумма
документов внесения
    "DepositSum": 10,
    "DiscountSum": 7.8,                                 //общая сумма скидок
    "FirstSaleNumber": 203,                             //номер первого платежного
документа
                                                        //в смене
    "Id": "a45a2228-e2ec-4764-94be-0346210519b6",
    "IssuingCashCount": 0,                              //количество и сумма
документов выдачи
    "IssuingCashSum": 0,
    "LastSaleNumber": 215,                              //номер последнего платежного
документа
                                                        //в смене
    "MarkSIMoneyBackCount": 0,                          //Кол-во товаров, марк-ых СИ
                                                        //возвращенных за смену
    "MarkSIRollbackCount": 0,                           //аннулированных за смену
    "MarkSISaleCount": 0,                               //реализованных за смену
    "MarkSISaleSum": 0,                                 //сумма товаров, марк-ых СИ

    "MarkUKZMoneyBackCount": 0,                         //Кол-во товаров, марк-ых УКЗ
                                                        //возвращенных за смену
    "MarkUKZRollbackCount": 0,                          //аннулированных за смену
    "MarkUKZSaleCount": 0,                              //реализованных за смену
    "MarkUKZSaleSum": 0,                                //сумма товаров, марк-ых УКЗ
    "Modified": {
        "T": 1739957707,
        "I": 1
    },                                                  //время записи в БД
    "MoneyBackCount": 4,                                //количество документов
возврата
```

```
    "MoneyBackSum": 17,                                 //сумма по документам
возврата
    "Number": "75",                                     //порядковый номер смены
    "Position": "2025-02-19T12:35:07.920",              //время закрытия смены
    "RollbackCount": 0,
    "RollbackSum": 0,
    "SaleCashSum": 48.59,                               //сумма продаж за наличные
    "SaleCashlessSum": 1374.93,                         //сумма продаж за безналичную
оплату
    "SaleCount": 8,                                     //количество документов
продаж
    "SaleOtherSum": 0,                                  //сумма иных способов оплат
    "SaleSum": 1423.52,                                 //общая сумма продаж
    "SerialId": "AVQ11169990664",                       //серийный номер токена
    "ShiftOpenDate": "2025-02-18T18:30:28.000",         //время открытия смены
    "ShiftOpening": false,                              //смена открыта/закрыта
    "Signature": "",
    "StatusFlag": 2,
    "Timestamp": "1739957707922411400",
    "TotalSum": 1423.52,                                //общая сумма продаж
    "Type": "CashShift",                                //тип документа: Z-отчёт
    "WithdrawCount": 1,                                 //количество и сумма
документов изъятия
    "WithdrawSum": 41.59,
    "_id": "67b5a5cb3a80cfb88ddbb18e"
}
```

Документы внесения, изъятия и выдачи

Далее приводится обобщенный пример документов данных типов.

```
{
    "Type" : "MoneyOrder",                  //тип документа (MoneyOrder для
внесения
                                            //и изъятия, IssuingCash для выдачи)
    "Author" : {"Name" : "Кассир2" },       //кассир
    "Currency" : "BYN",                     //код валюты
    "Deleted" : false,                      //признак того, что документ удален
    "IsDeposit" : true,                     //признак операции внесения (иначе
изъятие,
                                            //только для MoneyOrder)
    "Number" : "236",                       //номер документа
    "Position" : "2024-02-12T16:54:19.000", //кассовая дата
    "SerialId" : "AVQ13159990440",          //серийный номер СКО
    "Signature" : "0DEBD50F8412180C07183273",//УИД кассового документа
    "TotalSum" : 12.0
}
```

Документ аннулирования

Далее приводится пример документа аннулирования.

```
{
    "Type" : "Cancellation",                  //тип документа - аннулирование
    "Author" : {"Name" : "Кассир2" },         //кассир
    "Currency" : "BYN",                       //код валюты
    "Deleted" : false,                        //признак того, что документ удален
    "IsDeposit" : true,                       //признак операции внесения (иначе
изъятие,
                                              //только для MoneyOrder)
    "Number" : "236",                         //номер документа
    "Position" : "2024-02-12T16:54:19.000",   //кассовая дата
    "SerialId" : "AVQ13159990440",            //серийный номер СКО
    "Signature" : "0DEBD50F8412180C07183273",//УИД кассового документа
    "TotalSum" : 12.0,
    "CanceledDocNumber" : "235"               //номер документа, который был
аннулирован
}
```