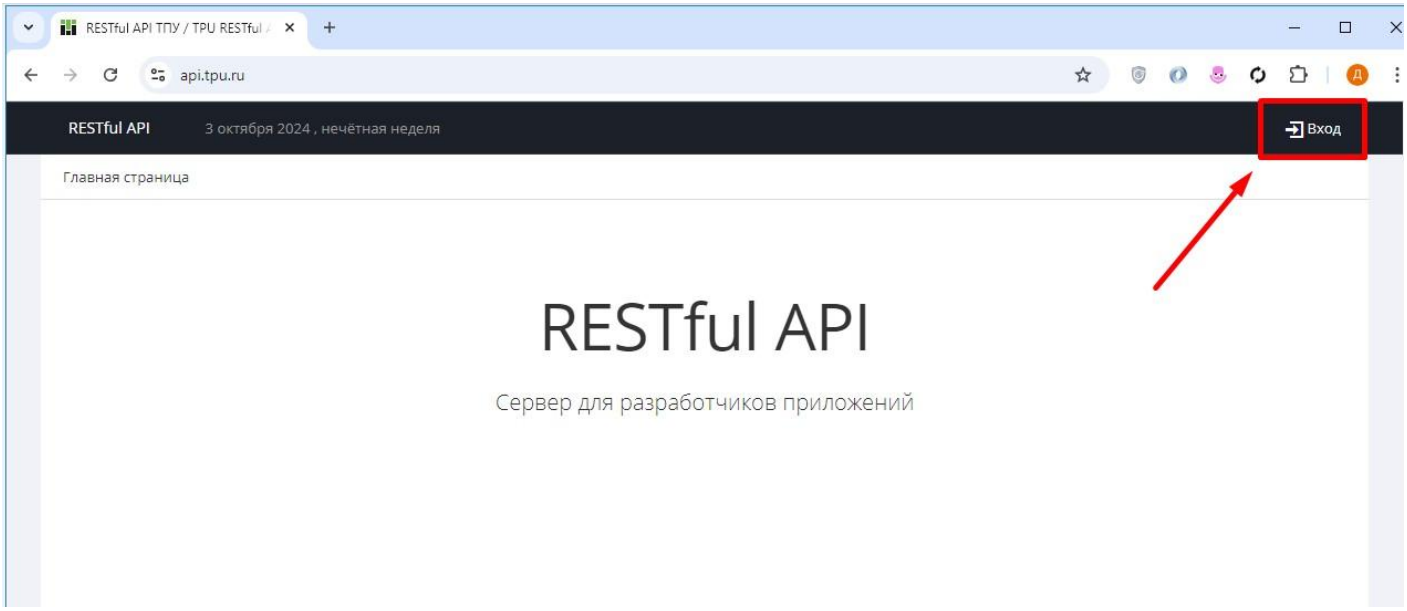
Создание приложения

- Регистрация нового приложения
- Добавление платформы приложения
- Добавлении версии приложения
- Расчет контрольной суммы

Регистрация нового приложения

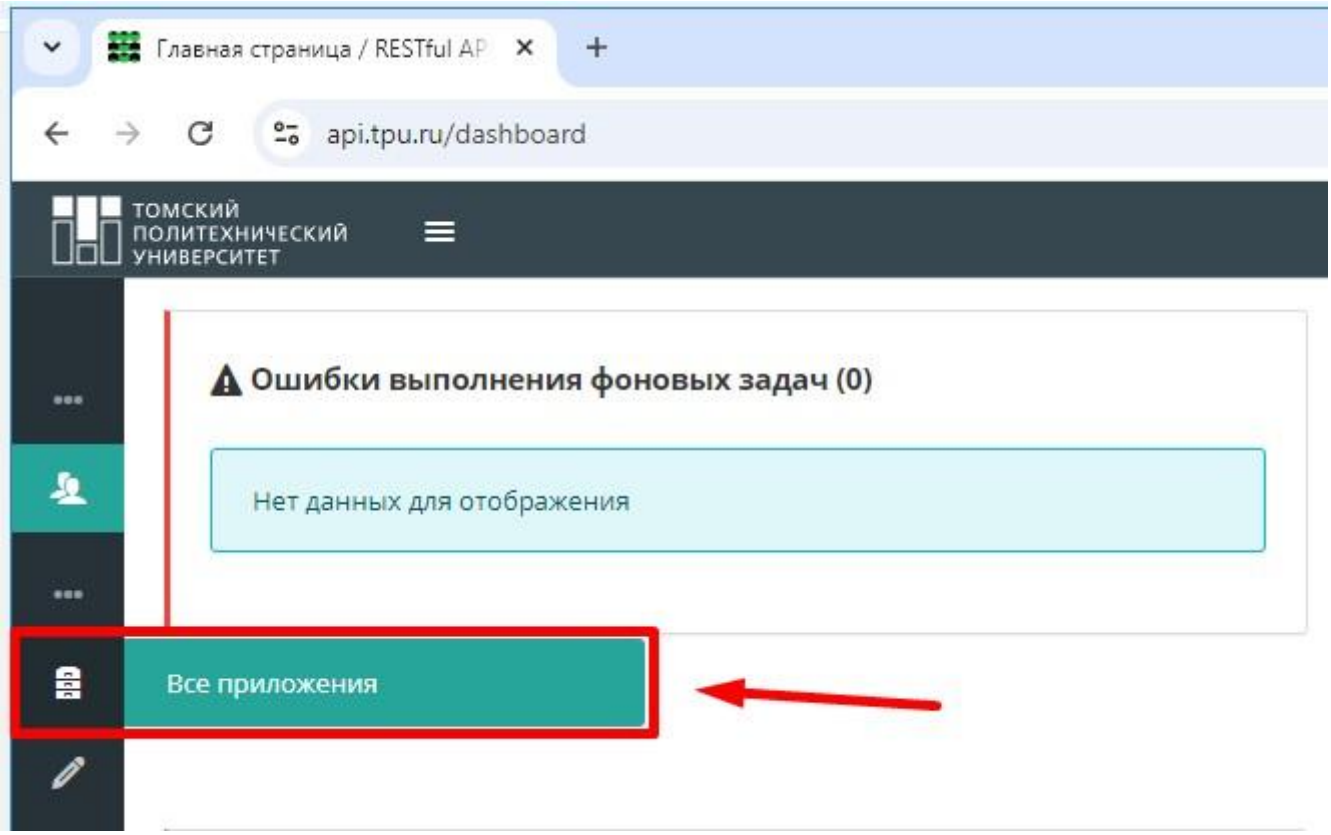
Для начала работы надо авторизоваться в приложение API TPU (https://api.tpu.ru)

1. Перейти на сайт API TPU (https://api.tpu.ru)
2. Нажать на кнопку вход (см. рисунок) следуя инструкциям авторизации ввести свой корпоративный логин и пароль.

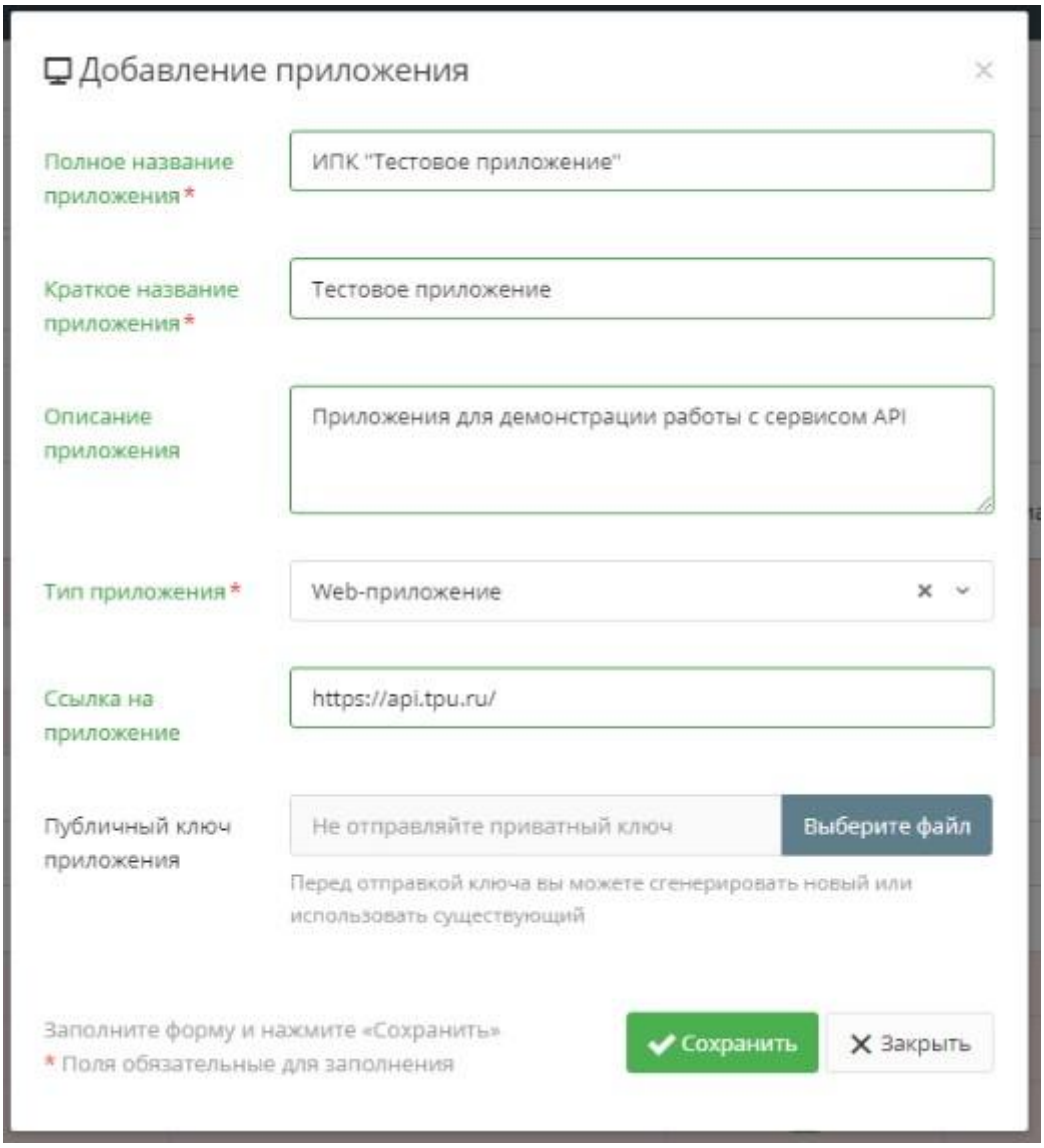


3. После авторизации в боковом меню выбрать раздел "Все приложения"

> В списке будут отображаться приложения, где вы являетесь разработчиком. Если кто-то вас добавит к своему приложению - оно отобразиться у вас в боковом меню.



4. В правом верхнем углу нажмите кнопку "Добавить" и заполните все поля, при необходимости можете сразу указать публичный SSL ключ вашего приложения

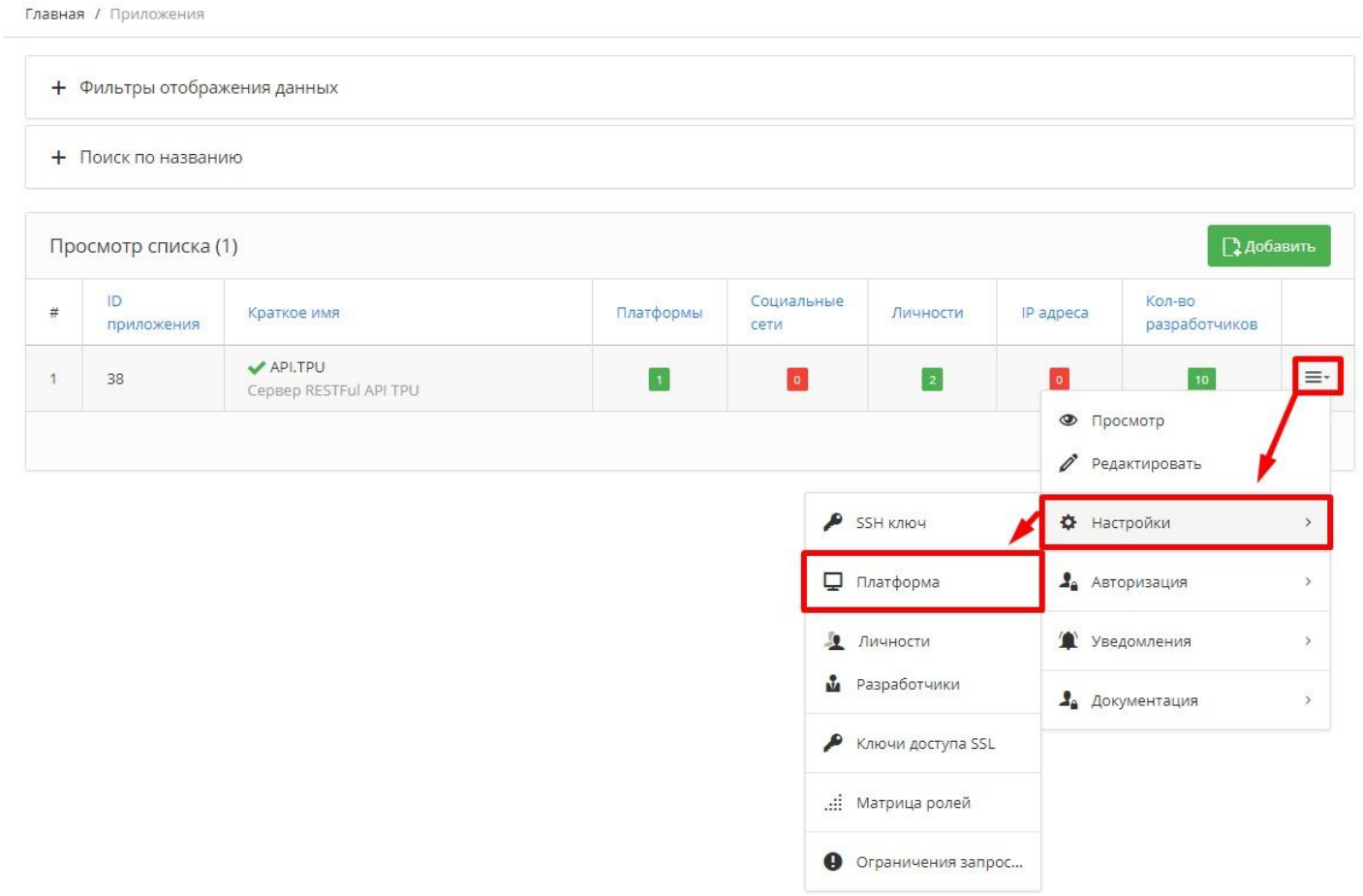


5. После добавления вашего приложения отправьте заявку на почту support@tpu.ru для подтверждения вашего приложения.

Добавление платформы приложения

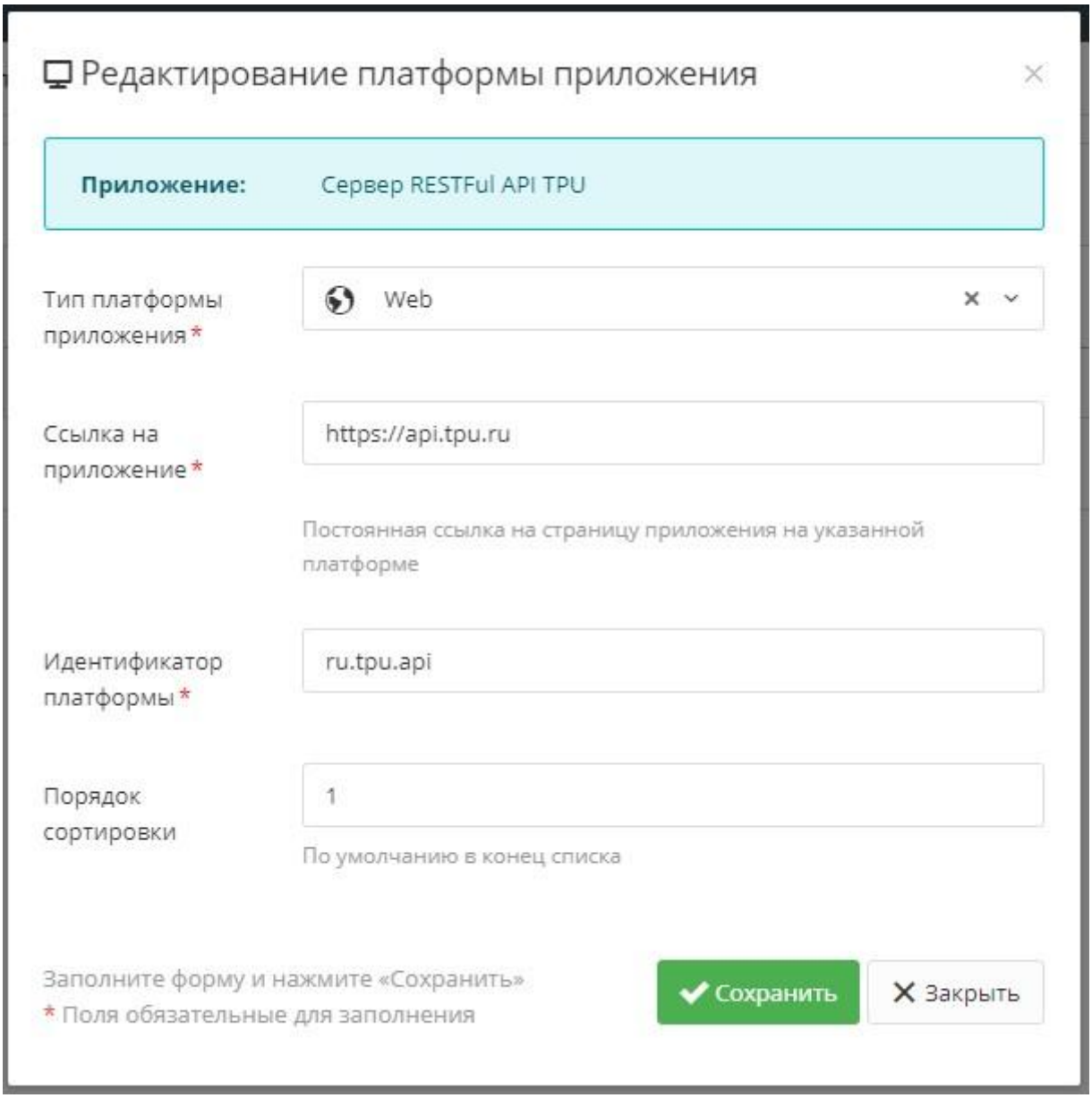
После добавления вашего приложения, требуется добавить платформы и получить API Key для работы с методами API.

Для этого в меню выберите следующие пункты "Настройки" -> "Платформы".



В появившимся окне необходимо выбрать тип платформы приложения и заполнить все поля. Нажать "Сохранить".

> Идентификатор платформы должен содержать только латинские буквы. Обычно это алиас системы или доменное имя сайта, записанное в обратном порядке

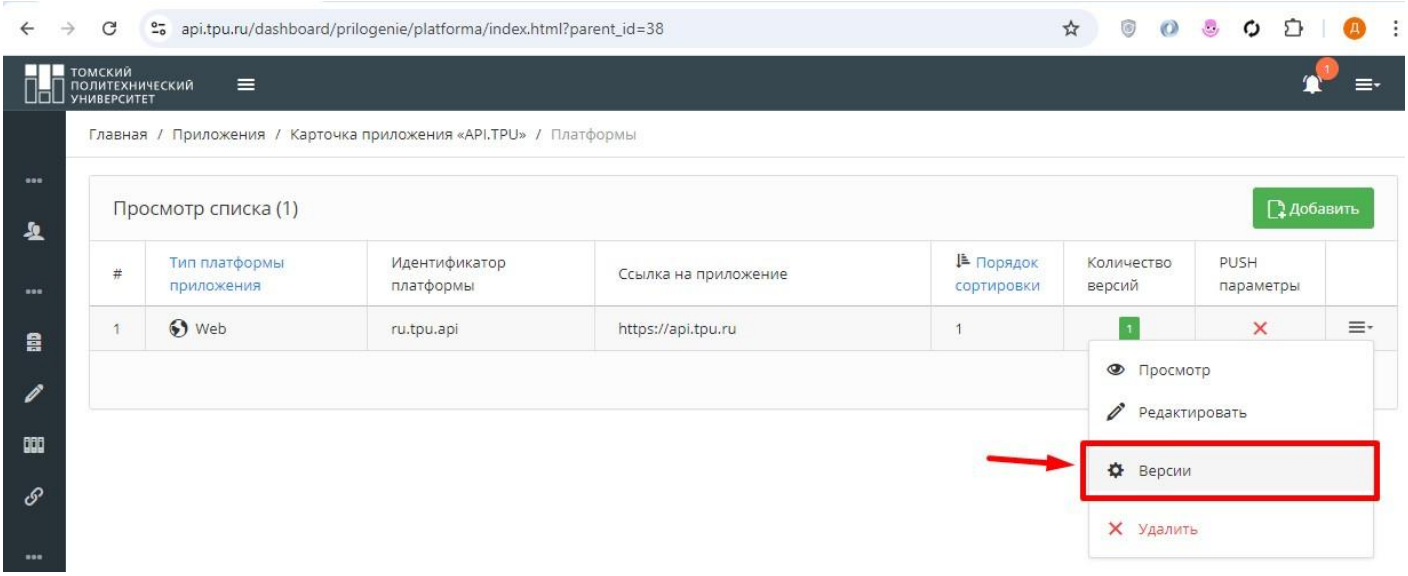

Редактирование платформы приложения
Приложение: Сервер RESTFul API TPU
Тип платформы приложения *
Web
Ссылка на приложение *
https://api.tpu.ru
Постоянная ссылка на страницу приложения на указанной платформе
Идентификатор платформы *
ru.tpu.api
Порядок сортировки
1
По умолчанию в конец списка
Заполните форму и нажмите «Сохранить»
* Поля обязательные для заполнения
Сохранить
Закрыть

Добавлении версии приложения

Следующий шаг - это добавление версии приложения, для того чтобы получить API Key и Private Key для работы с методами https://api.tpu.ru

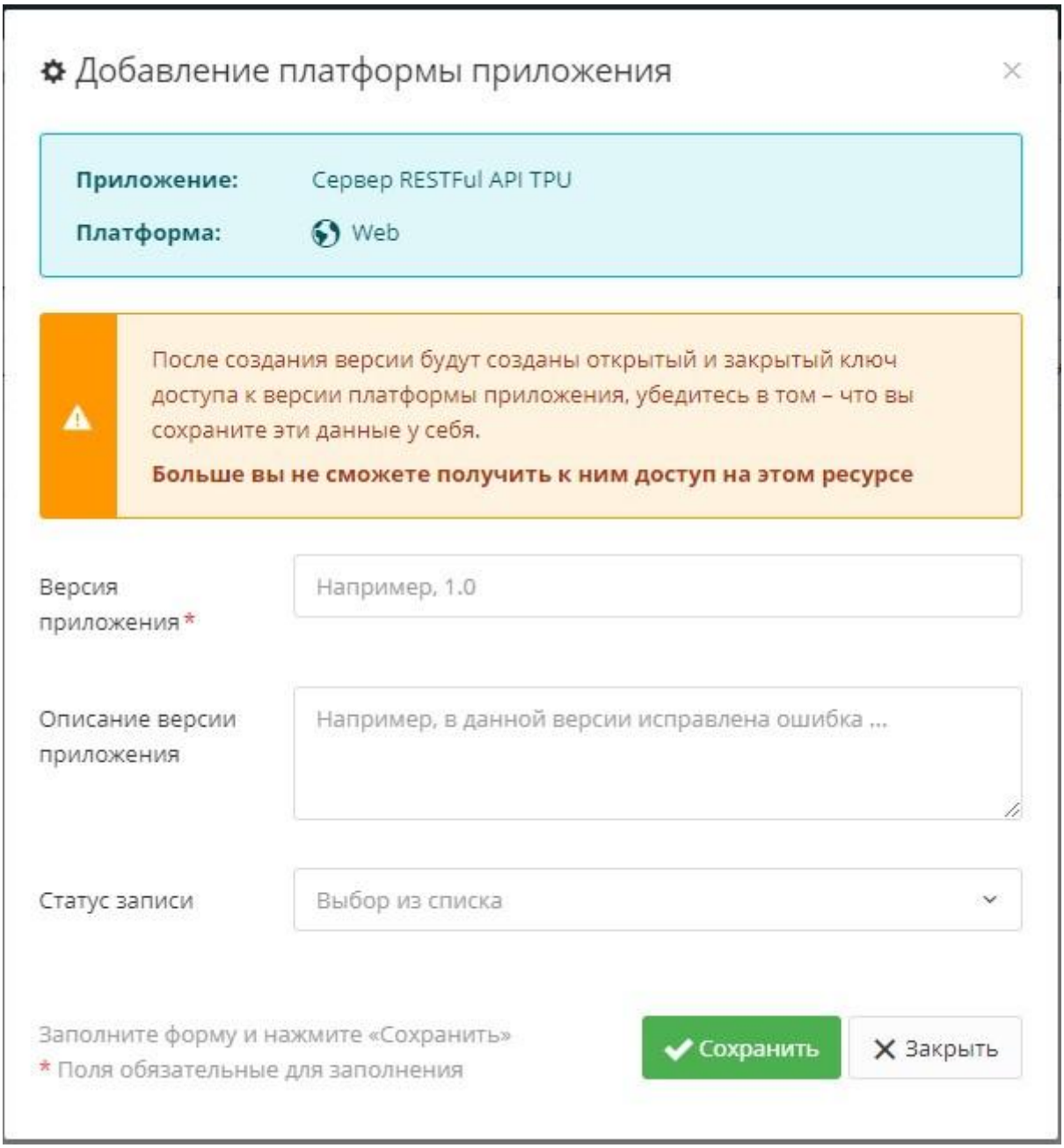
Выберите пункт "Версии" напротив интересующей вас платформы приложения.

> Каждая платформа приложения может иметь несколько разных версий

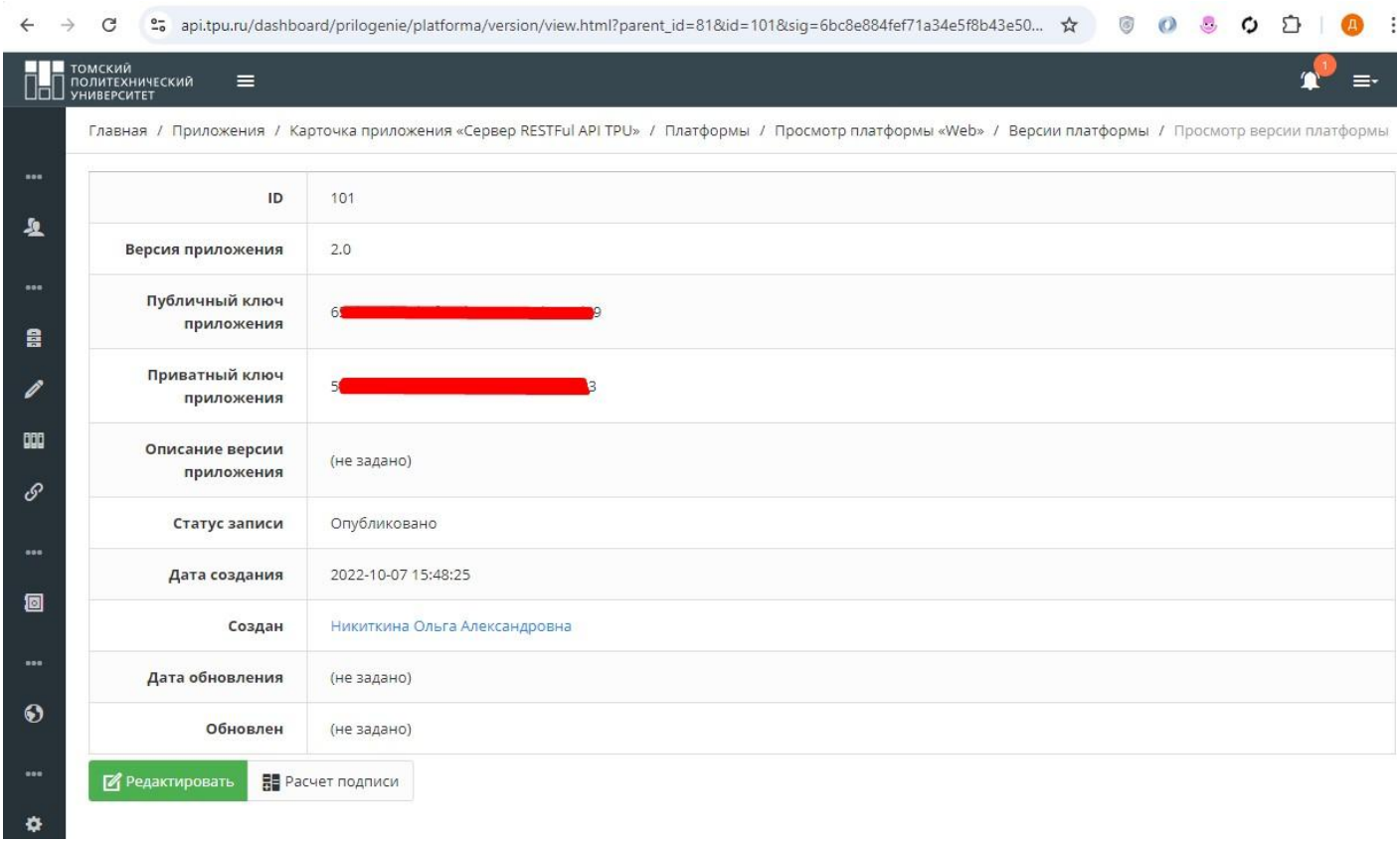


В модальном окне заполните номер версии приложения, описание и статус.

> Только *версия приложения* в статусе "Опубликовано" будет работать в API TPU

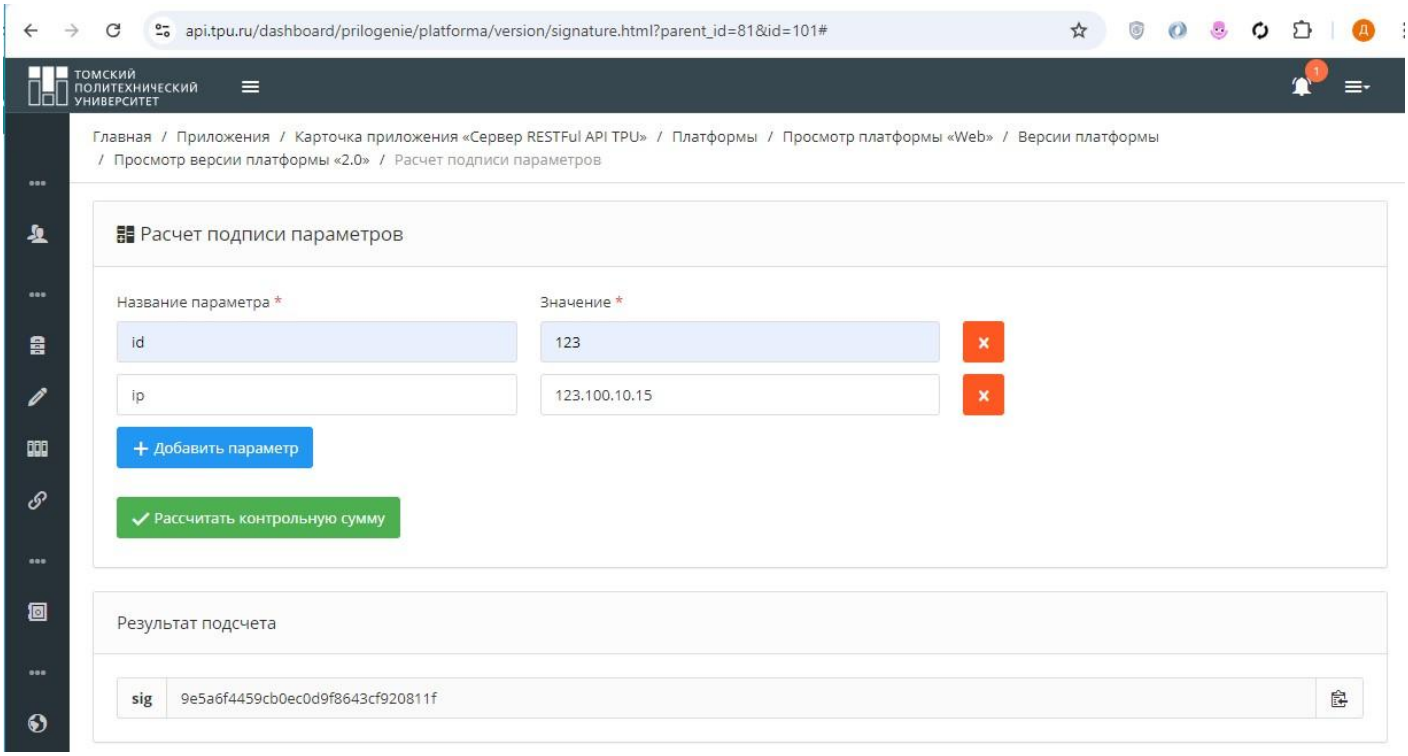


После добавления версии приложения вы получите два ключа *публичный* и *приватный.* Вы их будете использовать при работе с методами API TPU.


api.tpu.ru/dashboard/prilogenie/platforma/version/view.html?parent_id=81&id=101&sig=6bc8e884fef71a34e5f8b43e50...
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
Главная / Приложения / Карточка приложения «Сервер RESTFul API TPU» / Платформы / Просмотр платформы «Web» / Версии платформы / Просмотр версии платформы
ID
101
Версия приложения
2.0
Публичный ключ приложения
Приватный ключ приложения
Описание версии приложения
(не задано)
Статус записи
Опубликовано
Дата создания
2022-10-07 15:48:25
Создан
Никиткина Ольга Александровна
Дата обновления
(не задано)
Обновлен
(не задано)
Редактировать
Расчет подписи

Расчет контрольной суммы

Для некоторых методов требуется указать дополнительный параметр безопасности sig. В интерфейсе вы сможете проверить алгоритм расчета этого параметра на сервере. Разработчику приложения нужно реализовать свой собственный метод для получения данного параметра.



> Список параметров, которые могут быть переданы через HTTP-Get запрос, игнорируемые при расчёте контрольной суммы. т.к. эти параметры системные и не могут влиять на работу.
>
> 1. **sig**
> 2. **apikey**
> 3. **suppress_response_code**
> 4. **access_token**

Пример на языке PHP

```
public function sig(array $arguments, $secret)
{
    ksort($arguments);
```

```
    $string = '';

    foreach ($arguments as $key => $value) {
        $string .= "$key=$value";
    }

    return md5($string . $secret);
}
```

Пример на языке Java

```
import java.security.MessageDigest;
import java.util.List;
import java.util.Map;
import java.util.stream.Collectors;
import java.security.NoSuchAlgorithmException;

public String sig(Map<String, String> arguments, String secret) throws NoSuchAlgorithmException {
    StringBuilder builder = new StringBuilder();

    List<String> keys = arguments.keySet().stream().sorted().collect(Collectors.toList());

    for(String key : keys) {
        builder.append(key).append("=").append(arguments.get(key));
    }

    builder.append(secret);
    byte[] digest = MessageDigest.getInstance("MD5").digest(builder.toString().getBytes());

    StringBuilder resultBuilder = new StringBuilder();

    for(byte b : digest) {
        resultBuilder.append(String.format("%02x", b));
    }

    return resultBuilder.toString();
}
```

Пример на языке Scala

```
import java.security.MessageDigest
import scala.collection.immutable.ListMap

def sig(arguments: Map[String, String], secret: String): String = {
    MessageDigest.getInstance("MD5").digest {
        s"${ListMap(arguments.toSeq.sortBy({ case (key, value) => key }):_*).map({ case (key, value) =>
s"$key=$value" }).mkString}$secret".getBytes
    }.map(byte => String.format("%02x", byte)).mkString
}
```

Пример на языке Python

```
import collections
import hashlib

def sig(arguments, secret):
    ordered = collections.OrderedDict(sorted(arguments.items()))
    concated = "".join(list(map(lambda k,v: k + "=" + v, ordered.keys(), ordered.values()))) + secret
    return hashlib.md5(concated.encode()).hexdigest()
```