Макрос "Оглавление"

Пример KB-000059. Получить имя задачи по коду

Python

```
import requests

url =
'https://test.evateam.ru/api/'
method = 'POST'
headers = {
    'content-type':
'application/json',
    'Authorization': 'Bearer
gzz077755RRRIYxxxu50'
}
params = {
    'jsonrpc': '2.2',
    'method': 'CmfTask.get',
    'kwargs': {'filter': ['code',
'==', 'DEV-000003']}
}
result = requests.request(method,
url, json=params, headers=headers)
response = result.json()
print(response['result']['name'])
```

Curl

```
curl 'https://test.evateam.ru/api/' -X POST -H
"Content-Type: application/json" --data-raw '{
    "jsonrpc": "2.2",
    "method": "CmfTask.get",
    "kwargs": {"filter": ["code", "==", "DEV-
000003"]
    },
}' -H 'Authorization: Bearer
gzz077755RRRIYxxxu50'
```

Пример KB-000060. Получить список всех локальных пользователей

Python

```
import requests

url = 'https://test.evateam.ru/api/'
method = 'POST'
headers = {
    'content-type':
'application/json',
    'Authorization': 'Bearer
gzz077755RRRIYxxxu50'
}
params = {
    'jsonrpc': '2.2',
    'method': 'CmfPerson.list',
    'kwargs': {'filter':
[['user_local', '==', True],
['system', '==', False]]}
}
result = requests.request(method, url,
json=params, headers=headers)
response = result.json()
```

Curl

```
curl 'https://test.evateam.ru/api/' -X POST -
H "Content-Type: application/json" --data-raw
'{
    "jsonrpc": "2.2",
    "method": "CmfPerson.list",
    "kwargs": {'filter': [['user_local',
'==', True], ['system', '==', False]]
    },
}' -H 'Authorization: Bearer
gzz077755RRRIYxxxu50'
```

```
for user in response['result']:
    login = user['login']
    print(login)
```

Пример KB-000065. Всем локальным пользователям поставить опцию "Уведомления по email"

Пользователь API, с которого был запущен данный запрос, обязательно должен быть в группе **Admins.** Иначе невозможно будет включить **Режим администратора(admin_mode)**

Python пример:

```
import requests


class ApiSetEmail(object):

    def __init__(self):
        self.url = 'https://test.evateam.ru/api/'
        self.method = 'POST'
        self.headers = {
            'content-type': 'application/json',
            'Authorization': 'Bearer gzz077755RRRIYxxxu50'
        }

    def get_all_person(self):
        params = {
            'jsonrpc': '2.2',
            'method': 'CmfPerson.list',
            'kwargs': {'filter': [['user_local', '==', True], ['system', '==',
False], ['login', '!=', None]]},
            'callid': '47b51e15-24d6-43d3-b58c-55ea3f1d6a78'
        }
        result = requests.request(self.method, self.url, json=params,
headers=self.headers)
        response = result.json()
        return response

    def set_send_email(self, person_id):
        params = {
            'jsonrpc': '2.2',
            'method': 'CmfPerson.update',
            'args': [person_id],
            'kwargs': {'notify_email': True},
            'flags': {
                'admin_mode': True
            }

        }
        result = requests.request(self.method, self.url, json=params,
headers=self.headers)
        print(result.content)
        return result

    def run(self):
        all_person = self.get_all_person()
        for person in all_person['result']:
            print(f'Начинаем обрабатывать пользователя {person["name"]}')
            self.set_send_email(person['id'])


if __name__ == '__main__':
```

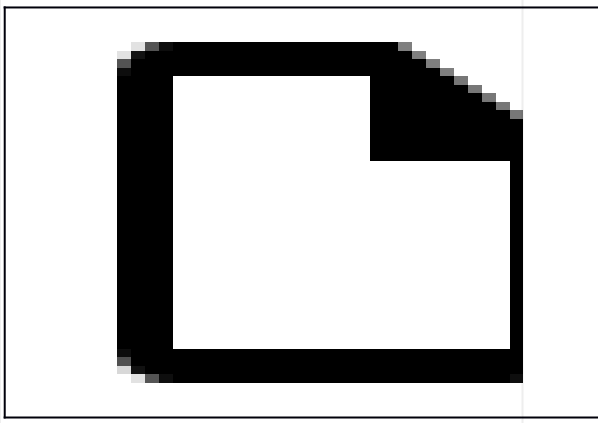
```
    worker = ApiSetEmail()
    worker.run()
```

Пример KB-000074. Создание задачи по API в спринте

Python скрипт

```
import requests

url = 'https://test.evateam.com/api/'
method = 'POST'
headers = {
    'content-type': 'application/json',
    'Authorization': 'Bearer gzz077755RRRIYxxxu50'
}
params = {
    'jsonrpc': '2.2',
    'method': 'CmfTask.create',
    'kwargs': {'name': "Новая задача по API 707",
               'lists': ['SPR-000006'],
               'text': '<p>Пример описания
7777</p>'}
}
result = requests.request(method, url, json=params,
headers=headers)
print(result.content)
```

Спринт



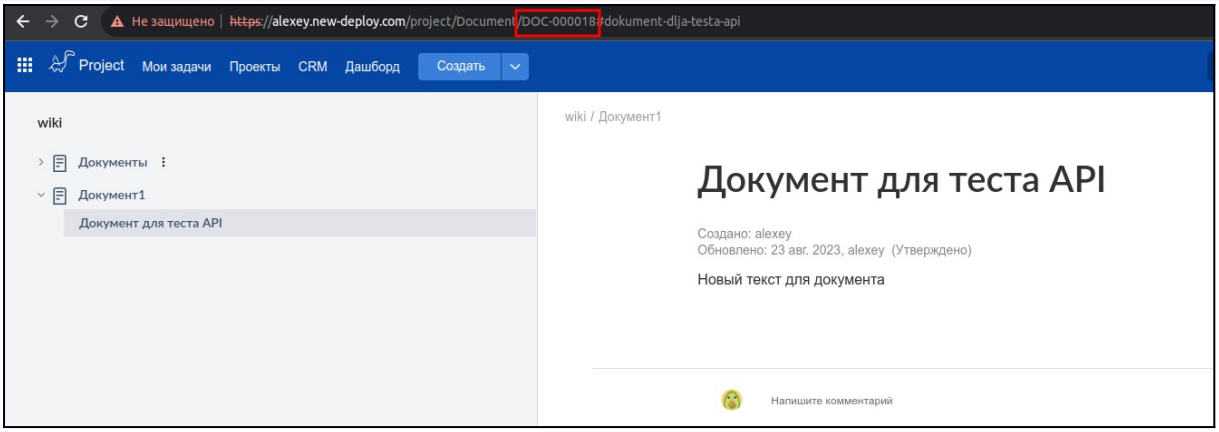
Пример KB-000079. Получение текста из документа по API

```
import requests

url = 'https://test.evateam.ru/api/'
method = 'POST'
headers = {
    'content-type': 'application/json',
    'Authorization': 'Bearer gzz077755RRRIYxxxu50'
}
params = {
    'jsonrpc': '2.2',
    'method': 'CmfDocument.get',
    'kwargs': {'filter': ['code', '==', 'DOC-000066'], 'fields':['text']}
}
result = requests.request(method, url, json=params, headers=headers)
response = result.json()
print(response['result']['text'])
```

Пример KB-000139. Изменение текста и выпуск документа по API. Документ находим по коду

Данный метод работает на версиях **02.12.32.0239** и выше

Найти код документа можно через браузерную строку



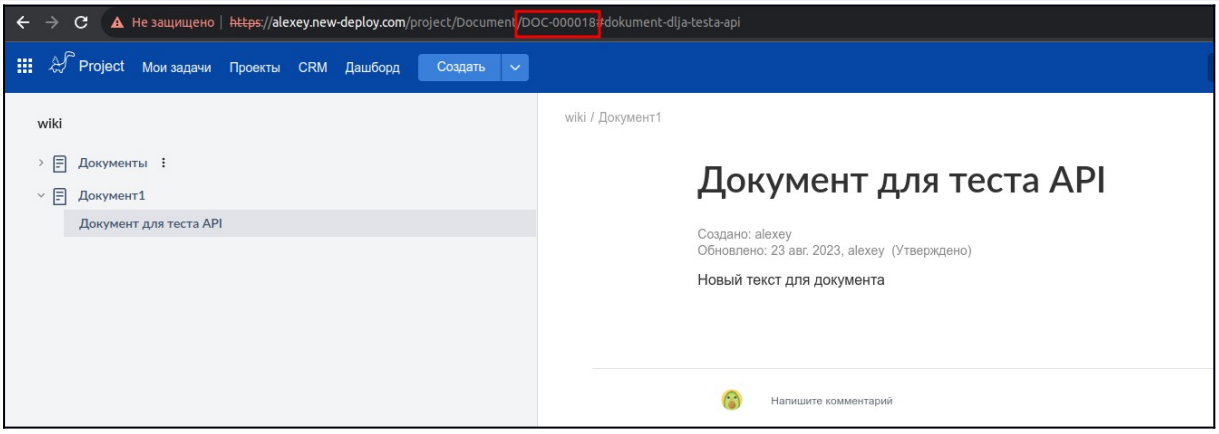

```
import requests

CODE_DOCUMENT = 'DOC-000018'
NEW_TEXT_DOCUMENT = 'Новый текст для документа'

def change_doc_text(code, text):
    url = 'https://test.evateam.com/api/'
    method = 'POST'
    headers = {
        'content-type': 'application/json',
        'Authorization': 'Bearer gzz077755RRRIYxxxu50'
    }

    params_get_document = {
    'jsonrpc': '2.2',
    'method': 'CmfDocument.get',
    'kwargs': {'filter': ['code', '==', code]}
    }

    # Находим документ для получения id документа
    result_get_document = requests.request(method, url, json=params_get_document,
headers=headers)
    if result_get_document.status_code != 200:
        print(f'Ошибка выполнения запроса - {result_get_document.content} код -
{result_get_document.status_code}')
        return
    doc_id = result_get_document.json()['result']['id']

    params_change_text = {
        'jsonrpc': '2.2',
        'method': 'CmfDocument.update',
        'args': [doc_id],
        'kwargs': {'text_draft': text}
    }

    # Меняем у найденного документа текст в черновике
    result_change_text = requests.request(method, url, json=params_change_text,
headers=headers)
    if result_change_text.status_code != 200:
        print(f'Ошибка выполнения запроса - {result_change_text.content} код -
{result_change_text.status_code}')
        return

    params_publish = {
```

```
        'jsonrpc': '2.2',
        'method': 'CmfDocument.do_publish',
        'args': [doc_id],
    }

    # Выпускаем черновик, чтобы изменения применились в документе
    result_publish = requests.request(method, url, json=params_publish,
headers=headers)
    if result_publish.status_code != 200:
        print(f'Ошибка выполнения запроса - {result_publish.content} код -
{result_publish.status_code}')
        return
    print(f'Изменили содержание документа {code}')

if __name__ == '__main__':
    change_doc_text(CODE_DOCUMENT, NEW_TEXT_DOCUMENT)
```

Пример KB-000141. Добавить в текст документа include на другой документ

Данный метод работает на версиях **02.12.32.0239** и выше

Найти код документа можно через браузерную строку




```
import requests

CODE_DOCUMENT = 'DOC-000018'
CODE_INCLUDE_DOCUMENT = 'DOC-000019'
URL = 'https://test.evateam.com/api/'
METHOD = 'POST'
HEADERS = {
    'content-type': 'application/json',
    'Authorization': 'Bearer gzz077755RRRIYxxxu50'
}


def get_document(code):
    params = {
        'jsonrpc': '2.2',
        'method': 'CmfDocument.get',
        'kwargs': {'filter': ['code', '==', code]}
    }

    result_get_document = requests.request(METHOD, URL, json=params, headers=HEADERS)
    if result_get_document.status_code != 200:
        print(f'Ошибка выполнения запроса - {result_get_document.content} код -
{result_get_document.status_code}')
        return
```

```
    return result_get_document.json()['result']


def paste_include(code, code_include):
    doc_id = get_document(code)['id']
    doc_include = get_document(code_include)
    doc_include_id = doc_include['id']
    doc_include_name = doc_include['name']

    include_template = (f'<div class=\"macros-panel macros-include-page\"
contenteditable=\"false\" ' \
                        f'data-macros=\"include-page\" data-param-
pageid=\"{doc_include_id}\" ' \
                        f'data-param-page-label=\"{doc_include_name}\" data-
id=\"ZBZ6o6s3vDcw9M\">\n<div data-layout=\"default\" ' \
                        f'class=\"extension-container\" contenteditable=\"false\"
data-id=\"ZBG0staXXaboA8\">' \
                        f'<span role=\"img\" aria-label=\"Макрос Include. \" data-
id=\"ZBmLSCs9evtSKq\"> ' \
                        f'<svg xmlns=\"http://www.w3.org/2000/svg\" id=\"Outline\"
viewbox=\"0 0 24 24\" ' \
                        f'width=\"14\" height=\"14\" data-id=\"ZBpcIwJp86j9Cy\">' \
                        f'<path
d=\"M19.949,5.536,16.465,2.05A6.958,6.958,0,0,0,11.515,0H7A5.006,5.006,0,0,0,2,5V19a5
.006,' \
                        f'5.006,0,0,0,5,5H17a5.006,5.006,0,0,0,5-
5V10.485A6.951,6.951,0,0,0,19.949,5.536ZM18.535,6.95A4.983,' \
                        f'4.983,0,0,1,19.316,8H15a1,1,0,0,1-1-
1V2.684a5.01,5.01,0,0,1,1.051.78ZM20,19a3,3,0,0,1-3,3H7a3,' \
                        f'3,0,0,1-3-
3V5A3,3,0,0,1,7,2h4.515c.164,0,.323.032.485.047V7a3,3,0,0,0,3,' \
                        f'3h4.953c.015.162.047.32.047.485Z\" data-
id=\"ZB6KtJ3w9fmvHz\"></path></svg> </span> ' \
                        f'<span class=\"extension-title\" data-
id=\"ZBQzVP7pGGEqWq\">Макрос Include.' \
                        f' \"Вставить страницу\" <span class=\"excerpt-title\" data-
id=\"ZBFPZbH2m169zR\">' \
                        f'| Имя = {doc_include_name}</span></span> <span
class=\"macros-actions\" data-id=\"ZBdApGVOHq6u5A\">' \
                        f' <span class=\"macros-edit\" style=\"margin-right: 7px;\"
data-id=\"ZB4wB7JqgUMlub\">' \
                        f' <svg _ngcontent-aiw-c17=\"\" viewbox=\"0 0 24 24\"
class=\"ng-star-inserted\" ' \
                        f'style=\"width: 14px; height: 14px; color: #444444;\" data-
id=\"ZBuw4iZdMqMjlB\">' \
                        f'<path _ngcontent-aiw-c17=\"\" fill=\"#444\"
d=\"M20.71,7.04C21.1,6.65 21.1,6 20.71,5.63L18.37,' \
                        f'3.29C18,2.9 17.35,2.9
16.96,3.29L15.12,5.12L18.87,8.87M3,17.25V21H6.75L17.81,9.93L14.06,' \
                        f'6.18L3,17.25Z\" data-id=\"ZBZ7NCCBwEkKfT\"></path></svg>
</span> <span class=\"macros-delete\" ' \
                        f'data-id=\"ZBvQ5riNHZrtWW\"> <svg viewbox=\"0 0 18 18\"
class=\"ng-star-inserted\" ' \
                        f'style=\"width: 14px; height: 14px; color: #222222;\" data-
id=\"ZB7FGUBJZQAFcE\">' \
                        f'<path fill=\"#222\" d=\"M16.5 3H12.75V1.5C12.75 1.10218
12.592 0.720644 12.3107 0.43934C12.0294' \
                        f' 0.158035 11.6478 0 11.25 0L6.75 0C6.35218 0 5.97064
0.158035 5.68934 0.43934C5.40804' \
                        f' 0.720644 5.25 1.10218 5.25 1.5V3H1.5V4.5H3V15.75C3 16.3467
3.23705 16.919 3.65901 17.341C4.08097' \
                        f' 17.7629 4.65326 18 5.25 18H12.75C13.3467 18 13.919 17.7629
14.341 17.341C14.7629 16.919 15 ' \
                        f'16.3467 15 15.75V4.5H16.5V3ZM6.75 1.5H11.25V3H6.75V1.5ZM13.5
15.75C13.5 15.9489 13.421 16.1397 ' \
                        f'13.2803 16.2803C13.1397 16.421 12.9489 16.5 12.75
16.5H5.25C5.05109 16.5 4.86032 16.421 ' \
                        f'4.71967 16.2803C4.57902 16.1397 4.5 15.9489 4.5
15.75V4.5H13.5V15.75Z\" stroke=\"none\" ' \
```

```
                        f'stroke-width=\"1\" class=\"ng-star-inserted\" data-
id=\"ZB1hzwlluN99Xh\"></path>' \
                        f'<path fill=\"#222\" d=\"M8.25 7.5H6.75V13.5H8.25V7.5Z\"
stroke=\"none\" ' \
                        f'stroke-width=\"1\" class=\"ng-star-inserted\" data-
id=\"ZBX6valjwRQmEr\">' \
                        f'</path><path fill=\"#222\" d=\"M11.25
7.5H9.75V13.5H11.25V7.5Z\" stroke=\"none\"' \
                        f' stroke-width=\"1\" class=\"ng-star-inserted\" data-
id=\"ZBkoGv5Ds4yQTk\"></path></svg> ' \
                        f'</span> </span></div>\n<div class=\"ak-renderer-wrapper\"
data-id=\"ZBGSXLvjhma8hR\">' \
                        f'</div>\n</div>\n<p data-id=\"X5RG3RtlpaKTc9\">')

    params_change_text = {
        'jsonrpc': '2.2',
        'method': 'CmfDocument.update',
        'args': [doc_id],
        'kwargs': {'text_draft': include_template}
    }

    result_change_text = requests.request(METHOD, URL, json=params_change_text,
headers=HEADERS)
    if result_change_text.status_code != 200:
        print(f'Ошибка выполнения запроса - {result_change_text.content} код -
{result_change_text.status_code}')
        return

    params_publish = {
        'jsonrpc': '2.2',
        'method': 'CmfDocument.do_publish',
        'args': [doc_id],
    }

    result_publish = requests.request(METHOD, URL, json=params_publish,
headers=HEADERS)
    if result_publish.status_code != 200:
        print(f'Ошибка выполнения запроса - {result_publish.content} код -
{result_publish.status_code}')
        return
    print(f'Изменили содержание документа {code}')


if __name__ == '__main__':
    paste_include(CODE_DOCUMENT, CODE_INCLUDE_DOCUMENT)
```

Пример KB-000145. Вывести автора и владельца у всех документов в системе

Python пример:

```
import requests

url = 'https://test.evateam.com/api/'
method = 'POST'
headers = {
    'content-type': 'application/json',
    'Authorization': 'Bearer gzz077755RRRIYxxxu50'
}
params = {
    'jsonrpc': '2.2',
    'method': 'CmfDocument.list',
    'kwargs': {'fields':['cmf_author', 'cmf_owner']}
}
result = requests.request(method, url, json=params, headers=headers)
```

```
response = result.json()
print('Ссылка;Название;Автор;Владелец')
for i in response['result']:
    doc_code = i['code']
    doc_name = i['name']
    author_login = i['cmf_author']['login']
    owner_login = i['cmf_owner']['login']
    print(f'https://test.evateam.com/project/Document/{doc_code};{doc_name};
{author_login};{owner_login}')
```

Результат:

```
Ссылка;Название;Автор;Владелец
https://test.evateam.com/project/Document/DOC-
000012;Документ1;ivanov@mail.com;ivanov@mail.com
https://test.evateam.com/project/Document/DOC-
000009;Документ1;ivanov@mail.com;ivanov@mail.com
https://test.evateam.com/project/Document/DOC-
000006;Документ2;ivanov@mail.com;ivanov@mail.com
https://test.evateam.com/project/Document/DOC-
000004;Документ1;ivanov@mail.com;ivanov@mail.com
https://test.evateam.com/project/Document/DOC-
000003;Документ1;ivanov@mail.com;ivanov@mail.com
```

Пример KB-000147. Заменить владельца в документах

Python пример:

```
import requests

# параметры выполнения запросов и авторизации
URL = 'https://test.evateam.com/api/'
METHOD = 'POST'
HEADERS = {
'content-type': 'application/json',
'Authorization': 'Bearer gzz077755RRRIYxxxu50'
}
#
# код пользователя с которого будем снимать документ
USER_LOGIN_FROM = 'ivanov@mail.com'
#код пользователя к которому документ будет привязан
USER_LOGIN_TO = 'petrov@mail.com'

# подготовка данных
def get_doc_list(user_from):
        # параметры для получения запроса списка документов
        doc_list_params = {
                'jsonrpc': '2.2',
        'method': 'CmfDocument.list',
        'kwargs': {'filter': ['cmf_owner.login', '==', user_from]}
        }
    # выполняем запросы для подготовки
    # список документов
        doc_list_result = requests.request(METHOD, URL, json=doc_list_params,
headers=HEADERS)
        if doc_list_result.status_code != 200:
                print(f'Ошибка выполнения запроса - {doc_list_result.content} код -
{doc_list_result.status_code}')
                return
        return doc_list_result.json()['result']

# обновление данных
def change_doc_owner():
        # выполяем функцию для получения документов
        doc_list = get_doc_list(USER_LOGIN_FROM)
```

```
        for elem in doc_list:
                # получаем данные из элементов в списке для вывода
                doc_id = elem['id']
                doc_name = elem['name']
                doc_code = elem['code']
                # параметры для обновления каждого элемента в списке
                doc_update_params = {
                        'jsonrpc': '2.2',
                        'method': 'CmfDocument.update',
                        'args': [doc_id],
                        'kwargs': {'cmf_owner': USER_LOGIN_TO},
                        'flags': {
                                'admin_mode': True
                        }
                }
                # выполняем запросы на обновление элементов в списке
                doc_update_result = requests.request(METHOD, URL,
json=doc_update_params, headers=HEADERS)
                if doc_update_result.status_code != 200:
                        print(f'Ошибка выполнения запроса -
{doc_update_result.content} код - {doc_update_result.status_code}')
                        return
                print(f'Изменили документ {doc_name}, код докумена {doc_code},
пользователь {USER_LOGIN_TO}')


if __name__ == '__main__':
        change_doc_owner()
```

Результат:

```
Изменили документ Документ1, код докумена DOC-000017, пользователь petrov@mail.com
Изменили документ Документ2, код докумена DOC-000020, пользователь petrov@mail.com
```

Пример KB-000149. Просмотр времени последнего изменения статуса

Python

```
import requests
import dateparser

url = 'https://test.evateam.ru/api/'
method = 'POST'
headers = {
    'content-type': 'application/json',
    'Authorization': 'Bearer gzz077755RRRIYxxxu50'
}
params = {
    'jsonrpc': '2.2',
    'method': 'CmfStatusHistory.get',
    'kwargs': {'order_by': ['-cmf_created_at'], 'filter': ['obj_code', '==', 'TSK-
4'], 'fields': ['cmf_created_at','from_status_name','to_status_name','transition']}
}
result = requests.request(method, url, json=params, headers=headers)
response = result.json()
from_status = response['result']['from_status_name']
to_status = response['result']['to_status_name']
time = response['result']['cmf_created_at']
time = dateparser.parse(time)
time = time.strftime("%Y-%m-%d, %H:%M:%S")
print('Задача была перемещена из статуса "{}" в статус "{}" в {}'.format(from_status,
to_status, time))
```

Результат:

```
Задача была перемещена из статуса "Открыто" в статус "В работе" в 2023-09-15 14:11:23
```

Пример KB-000155. Найти список документов имя которых содержит "текст"

Пользователь API, с которого был запущен данный запрос, обязательно должен быть в группе Admins. Иначе невозможно будет включить Режим администратора(admin_mode)

Поиск чувствительный к регистру символов

"Bearer gzz077755RRRIYxxxu50" - Токен, который можно сгенерировать из личной карточки в разделе безопасность.

Python пример:

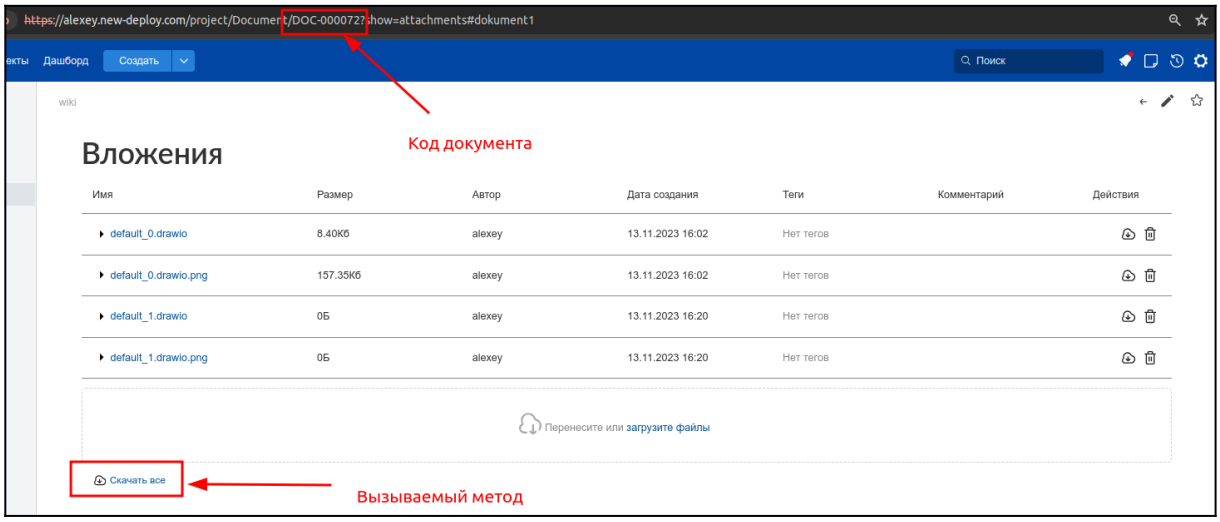
```
import requests

url = 'https://example.evateam.ru/api/'
method = 'POST'
headers = {
    'content-type': 'application/json',
    'Authorization': 'Bearer gzz077755RRRIYxxxu50'
}
params = {
    'jsonrpc': '2.2',
    'method': 'CmfDocument.list',
    'kwargs': {'filter': ['name', 'LIKE', '%текст%']},
    'flags': {'admin_mode': True}
}
result = requests.request(method, url, json=params, headers=headers)
response = result.json()
present_url = url[:-4]
for elem in response['result']:
    doc_code = elem['code']
    doc_name = elem['name']
    print(f'Название документа: {doc_name} Ссылка на документ:
{present_url}project/Document/{doc_code}')
```

Результат:

```
Название документа: Документ_текст Ссылка на документ:
https://example.evateam.ru/project/Document/DOC-000012
```

Пример KB-000173. Скачать все вложения из документа

Данный метод работает на версиях **02.17.08.0155** и выше




```
import requests, zipfile, io


domain = 'https://test.evateam.ru/'
method = 'POST'
# В эту папку на локальной машине будут сохранены все вложения
destination_directory = '/path/to/destination_directory'
# Код документа из которого нужно получить все вложения
doc_code = 'DOC-000072'
headers = {
    'content-type': 'application/json',
    'Authorization': 'Bearer gzz077755RRRIYxxxu50'
}

def get_document(code):
    params_get = {
        'jsonrpc': '2.2',
        'method': 'CmfDocument.get',
        'kwargs': {'filter': ['code', '==', code]}
    }

    result_get_document = requests.request(method, f'{domain}api/', json=params_get,
headers=headers)
    if result_get_document.status_code != 200:
        print(f'Ошибка выполнения запроса - {result_get_document.content} код -
{result_get_document.status_code}')
        return

    return result_get_document.json()['result']

def download_all_attach():
    doc_id = get_document(doc_code)['id']
    params = {
        'jsonrpc': '2.2',
        'callid': '575ff1ac-0dc6-47fc-a57d-7f77b68e6bb6',
        'method': 'CmfDocument.download_all_attachment',
        'flags': {
            'admin_mode': True
        },
        'args': [doc_id]
    }
    result = requests.request(method, f'{domain}api/', json=params, headers=headers)
    zip_url = result.json()['result']
    url = f'{domain}{zip_url}'
    r = requests.get(url, headers=headers)
    z = zipfile.ZipFile(io.BytesIO(r.content))
    z.extractall(destination_directory)


if __name__ == '__main__':
```

```
    download_all_attach()
```

Пример KB-000177. Загрузить файл в виде вложения по API в задачу или документ

Данный метод работает на версиях **02.17.08.0155** и выше

```
import requests
import os
from uuid import uuid4, UUID

token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'
# Путь до файла
full_file_path = '/path/to/local/MyFile.png'
# Имя файла
attachment_name = 'MyFile.png'
# Код объекта, в который нужно загрузить вложение (Это может быть документ или
задача)
obj_code = 'TASK-000001'

def call(url, method, params=None, args=None, fields="*", filter=None,
flags={'admin_mode': True}):
    if fields == "*":
        fields = ['**']
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "args": args,
        "kwargs": params,
        "fields": fields,
        "filter": filter,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()


session = requests.session()
session.headers['Authorization'] = f'Bearer {token}'
# Создаем пустое вложение
resp = call(domain_eva, 'CmfAttachment.create', dict(name=attachment_name,
parent=obj_code))
attach = resp['result']
# Получаем созданное вложение по id
resp = call(domain_eva, 'CmfAttachment.get', params={'filter':['id', '==', attach],
'fields':['url']})
# Получаем url, на который будем загружать файл
external_url = resp['result']['url']
full_external_url = f'{domain_eva}{external_url}'
# Открываем файл на локальной машине
files = {'file': open(full_file_path, 'rb')}
# Загружаем файл по url'y
r = session.post(full_external_url, files=files)
assert r.status_code == 200, print(f'Не удалось загрузить вложение - {r.json()}')
```

Пример KB-000185. Получение количества открытых/закрытых задач за период.

```
import requests
import os
from uuid import uuid4, UUID


token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'
# Указываем коды проектов
projects = ['main', 'pervyj']
# Указываем тип статуса(OPEN, IN_PROGRESS, IN_REVIEW, CLOSED)
type_status = 'OPEN'
# Указываем начальный период поиска
start_need_period = '2023-08-01'
# Указываем конечный период поиска
end_need_period = '2023-10-01'


def call(url, method, params=None, args=None, fields="*", filter=None,
flags={'admin_mode': True}):
    if fields == "*":
        fields = ['**']
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "args": args,
        "kwargs": params,
        "fields": fields,
        "filter": filter,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()
session = requests.session()
session.headers['Authorization'] = f'Bearer {token}'
for code_project in projects:
    if start_need_period > end_need_period:
        print('Некорректно указан период для поиска')
        break
    result_get_project = call(domain_eva, 'CmfProject.get', filter=['code', '==',
code_project], fields=['id', 'code'])
    # Выводим ID и название проекта
    project_id = result_get_project['result']['id']
    project_name = result_get_project['result']['name']
    #print(project_id)

    # В параметрах запроса необходимо указать корректные даты интервала для поиска
    result_get_tasks = call(domain_eva, 'CmfTask.list', filter=[['cache_status_type',
'!=', type_status],['parent_id', '==', project_id],
                          ['cmf_created_at', '<=', end_need_period],
['cmf_created_at', '>=', start_need_period]], fields=['name', 'code'])
```

```
    count_task = 0
    for task in result_get_tasks['result']:
        task_code = task['code']
        task_name = task['name']
        count_task += 1
        print(f'Проект - {project_name}; Код задачи - {task_code}; Название -
{task_name}')
```

Пример KB-000187. Получение суммарного значения фактических трудозатрат по проекту за период.

```
import requests
import os
from uuid import uuid4, UUID


token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'
# Указываем код проекта
projects = ['main', 'pervyj']
# Указываем тип статуса(OPEN, IN_PROGRESS, IN_REVIEW, CLOSED)
type_status = 'OPEN'
# Указываем начальный период поиска
start_need_period = '2023-08-01'
# Указываем конечный период поиска
end_need_period = '2023-09-01'

def call(url, method, params=None, args=None, fields="*", filter=None,
flags={'admin_mode': True}):
    if fields == "*":
        fields = ['**']
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "args": args,
        "kwargs": params,
        "fields": fields,
        "filter": filter,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()
session = requests.session()
session.headers['Authorization'] = f'Bearer {token}'

for code_project in projects:
    if start_need_period > end_need_period:
        print('Некорректно указан период для поиска')
        break
```

```
    result_get_project = call(domain_eva, 'CmfProject.get', filter=['code', '==',
code_project], fields=['id', 'name'])
    # Выводим ID и название проекта
    project_id = result_get_project['result']['id']
    project_name = result_get_project['result']['name']

    result_get_tasks = call(domain_eva, 'CmfTask.list', filter=[['cache_status_type',
'==', type_status], ['parent_id', '==', project_id],
                            ['cmf_created_at', '>=', start_need_period],
['cmf_created_at', '<=', end_need_period]], fields=['id', 'code', 'name'])

    all_spent = 0
    spent_all_projects = 0
    for task in result_get_tasks['result']:
        task_id = task['id']
        task_code = task['code']
        task_name = task['name']
        # Определение записей в журнале работ за этот же период
        result_timetracker = call(domain_eva, 'CmfTimeTrackerHistory.list',
filter=[['parent', '==', task_id], ['start_date', '>=', start_need_period],
                                            ['start_date', '<=', end_need_period]],
fields=['id', 'time_spent'])

        # Подсчет затраченного времени на задачу
        for time in result_timetracker['result']:
            spent = time['time_spent']
            all_spent += spent
        # Выводим данные по задачам для печати
        print(f'Проект - {project_name}; Код задачи - {task_code}; Название -
{task_name}; Зафиксировано - {all_spent} мин.')

        # Подсчет суммарного времени затраченного на все задачи в проекте
        spent_all_projects = spent_all_projects + all_spent
    print(f'Сумма потраченного времени на проект {spent_all_projects}')
```

Пример KB-000199. Выгрузка трудозатрат по задачам, отфильтрованных по дате создания задачи.

```
import requests
import os
from uuid import uuid4, UUID


token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'
#Отфильтровать задачи по периоду создания задачи
start_need_period = '2024-01-19'
end_need_period = '2024-01-20'


def call(url, method, params=None, args=None, fields="*", filter=None,
flags={'admin_mode': True}):
    if fields == "*":
        fields = ['**']
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "args": args,
        "kwargs": params,
        "fields": fields,
        "filter": filter,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    if not fields:
```

```
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()
session = requests.session()
session.headers['Authorization'] = f'Bearer {token}'

#Получим весь список задач за определенный интервал времени
result_get_task = call(domain_eva, 'CmfTask.list', filter=[['cmf_created_at', '>=',
start_need_period],
                                                           ['cmf_created_at', '<=',
end_need_period],['timetracker_history', 'EXISTS',
[]]],fields=['timetracker_history.*',

'timetracker_history.cmf_owner.name', 'parent.name'])

#Найдем ID задачи, ID проекта, код задачи, название задачи
for t in result_get_task['result']:
    task_id = t['id']
    project_id = t['parent_id']
    task_code = t['code']
    task_name = t['name']
    parent = t['parent']['name']

    # Определение записей в журнале работ у задачи
    for tt in t['timetracker_history']:
        spent = tt['time_spent']
        code_time = tt['code']
        start_date = tt['start_date']
        create_date = tt['cmf_created_at']
        modifed_date = tt['cmf_modified_at']
        tt_owner = tt['cmf_owner']['name']
        tt_owner_login = tt['cmf_owner']['login']

        print(f'Название проекта - {parent}; Название задачи - {task_name}; Код
задачи - {task_code}; '
              f'Код таймтрекера - {code_time}; автор записи - {tt_owner}; email -
{tt_owner_login}; Дата за которую отчитались - {start_date};'
              f' Зафиксированное время - {spent} мин.; Дата время изменения -
{modifed_date}')
```

Пример KB-000205. Добавление комментария в задачу

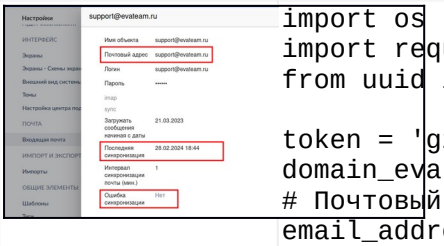
```
import requests

url = 'https://test.evateam.ru/api/'
method = 'POST'

headers = {
    'content-type': 'application/json',
    'Authorization': 'Bearer токен'
}
params = {
    'jsonrpc': '2.2',
    'method': 'CmfComment.create',
    'kwargs': {'parent': "Код_задачи",
               'text': "<p>текст</p>"}
}
result = requests.request(method, url, json=params, headers=headers)
print(result.content)
```

Пример KB-000209. Создание проекта

```
import requests
import os
from uuid import uuid4, UUID

#параметры для подключения к API
token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'


#Функция отправки запросов
def call(url, method, params=None, args=None, fields="*", filter=None,
flags={'admin_mode': True}):
    if fields == "*":
        fields = ['**']
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "args": args,
        "kwargs": params,
        "fields": fields,
        "filter": filter,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()
session = requests.session()
session.headers['Authorization'] = f'Bearer {token}'


#Параметры для запросов
logic_type_code = 'project.agile:default' #Тип проекта
activity_code = 'softdev' #Вид деятельности
workflow_code = 'task.agile:default' #Бизнес процесс по умолчанию
owner_login = 'new1@mail.com' #Владелец
person_login_list = ['new1@mail.com', 'new2@mail.com'] #Список пользователей
prj_name = 'Новый проект' #Название проекта
prj_code = 'NP' #Код проекта
prj_task_code_prefix = 'NP' #Префикс задач
prj_category_id = 'CmfProjectCategory:acc66eaa-cfd1-11ee-8de6-0242ac110002'
#Категория проекта

#Создаем проект
project = call(domain_eva, 'CmfProject.create', params={'name': prj_name, #Название
проекта
                                                        'code': prj_code, #Код
проекта
                                                        'task_code_prefix':
prj_task_code_prefix, #Префикс задач
                                                        'activity': activity_code,
#Вид деятельности
                                                        'logic_type':
logic_type_code, #Тип проекта Agile/SD/ и т.п.
                                                        #Это минимальный набор
параметров для создания проекта
                                                        #Ниже дополнительные
параметры
```

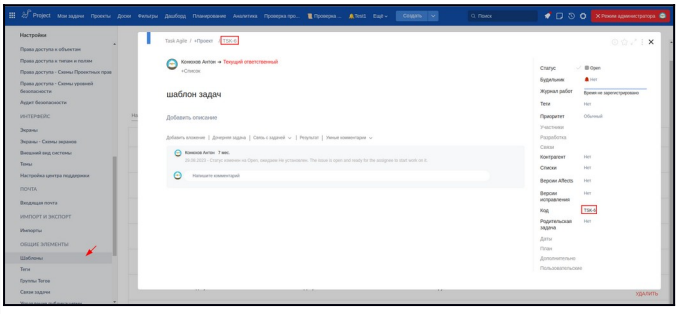
```
                                                    'workflow': workflow_code,
#Стандартный бизнес процесс
                                                    'cmf_author': owner_login,
#Автор
                                                    'cmf_owner': owner_login,
#Владелец
                                                    'cmf_owner_assistants':
person_login_list, #Заместители владельца
                                                    'perm_policy': 'private',
#Права доступа
                                                    'category': prj_category_id,
#Категория проекта
                                                    'sl_task_only_owner_close':
False, #Опция "Задачи может закрыть только постановщик"
                                                    'add_object_type': False
#Опция "Добавлять тип объекта в имя"
                                                    })['result']

print(f'Пытаемся создать проект {prj_name}, проект получает ID {project}')

#Параметры которые нельзя указать при создании (Структура проекта)
updated_proj = call(domain_eva, 'CmfProject.update', filter=['id', '==', project],
params={'show_tasks': False, #Задачи/Бэклог
                                                    'show_archive': False, #Архив
                                                    'show_chat': True, #Чат
                                                    'show_disk': False, #Диск
                                                    'show_pfeed': False, #Лента
                                                    'show_epic': False, #Эпик
                                                    'show_sprint': False, #Спринт
                                                    'show_release': False,
#Релизы
                                                    'show_roadmap': False,
#Роадмап
                                                    'show_trashcan': False,
#коризина
                                                    'show_filters': False,
#Фильтры
                                                    'show_components': False,
#Компоненты
                                                    'show_reports': False #Отчеты
                                                    })['result']

print(f'Обновили проект с ID {updated_proj}.')
```

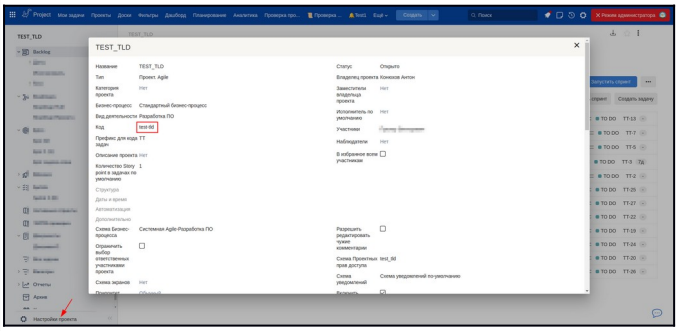
Пример KB-000215. Получить по API время последней синхронизации почтового сервера входящей почты и ошибку синхронизации

Сервер входящей почты	Код API запроса
	`import os` `import requests` `from uuid import uuid4` `token = 'gzz077755RRRIYxxxu50'` `domain_eva = 'https://test.evateam.ru/'` `# Почтовый адрес` `email_address = 'support@evateam.ru'` `def call(url, method, params=None, args=None, fields="*",`

```
filter=None, flags={'admin_mode': True}):
    if fields == "*":
        fields = ['**']
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "args": args,
        "kwargs": params,
        "fields": fields,
        "filter": filter,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()


session = requests.session()
session.headers['Authorization'] = f'Bearer {token}'
a = call(domain_eva, 'CmfMailbox2.get', filter=['email', '==',
email_address], fields=['*'])
error_log = a['result']['sync_error_log']
last_sync = a['result']['last_sync']
print(f'Время последней синхронизации {last_sync}, ошибка при
выполнении "{error_log or "Ошибки нет"}"')
```

Пример KB-000235. Создать задачу по шаблону в проекте

Коды объектов	
Код шаблона задачи 	`import os` `import requests` `import json` `from uuid import uuid4` `# Параметры подключения` `domain_eva = 'https://test.evateam.ru/'` `token = 'gzz077755RRRIYxxxu50'` `# Код проекта` `code_project = 'test-project'` `# Код шаблона задачи` `code_template ='TSK-2'`
Код проекта	`def call(url, method, params=None, args=None, f` `if fields == '*':` `fields = ['**']` `payload = {` `'callid': str(uuid4()),` `'method': method,` `'args': args,` `'kwargs': params,` `'fields': fields,`



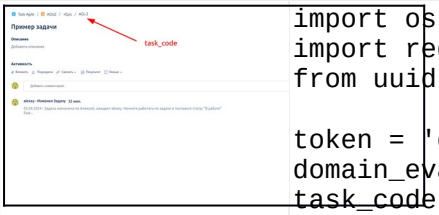
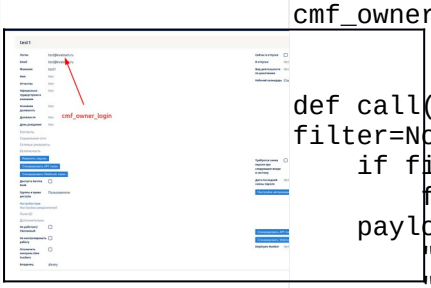
```
        'filter': filter,
        'flags': flags,
        'no_meta': no_meta,
        'jsonrpc': '2.2'
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/
    assert resp.status_code == 200, resp.conten
    return resp.json()


with requests.session() as session:
    session.headers['Authorization'] = f'Bearer
    result_get_task = call(domain_eva, 'CmfTask
    template_result = result_get_task.get('resu
    template_id = template_result.get('id')
    result = call(domain_eva, 'CmfTask.create_t

    print(result)
```

Пример KB-000237. Создание пользователя

```
import os
import requests
from uuid import uuid4


token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'
user_name = 'Иванов Иван Иванович'
user_login =  'test1@evateam.ru'
user_email = 'test1@evateam.ru'

def call(url, method, params=None, args=None, fields="*", filter=None,
flags={'admin_mode': True}):
    if fields == "*":
        fields = ['**']
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "args": args,
        "kwargs": params,
        "fields": fields,
        "filter": filter,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()
```

```
session = requests.session()
session.headers['Authorization'] = f'Bearer {token}'
params = {'name': user_name,
          'login': user_login,
          'user_local': True,
          'email': user_email}
a = call(domain_eva, 'CmfPerson.create', params=params)
print(a)
```

Пример KB-000239. Назначение пользователя в поле "Постановщик" в задаче

Коды объектов	Код API запроса
 	import os import requests from uuid import uuid4 token = 'gzz077755RRRIYxxxu50' domain_eva = 'https://test.evateam.ru/' task_code = 'AGL-2' cmf_owner_login = 'test@evateam.ru' def call(url, method, params=None, args=None, fields="*", filter=None, flags={'admin_mode': True}): if fields == "*": fields = ['**'] payload = { "callid": str(uuid4()), "method": method, "args": args, "kwargs": params, "fields": fields, "filter": filter, "flags": flags, "jsonrpc": "2.2" } if not fields: payload.pop('fields') if not params: payload.pop('kwargs') if not filter: payload.pop('filter') if not args: payload.pop('args') resp = session.post(os.path.join(url, 'api/'), json=payload) assert resp.status_code == 200, resp.content return resp.json() session = requests.session() session.headers['Authorization'] = f'Bearer {token}' # Находим id задачи по коду task = call(domain_eva, 'CmfTask.get', filter=['code', '==', task_code], fields=['*']) if not task['result']: print(f'Не найдена задача по коду {task_code}') task_id = task['result']['id']

```
# Находим id пользователя по логину
person = call(domain_eva, 'CmfPerson.get', filter=['login', '==',
cmf_owner_login], fields=['*'])
if not person['result']:
    print(f'Не найден пользователь по логину {cmf_owner_login}')

person_id = person['result']['id']

params = {
    'cmf_owner': {
        'id': person_id
    }
}
a = call(domain_eva, 'CmfTask.update', params=params,
args=[task_id])
```

Пример KB-000245. Создание контрагента по API

```
import os
import requests
from uuid import uuid4

token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'
org_name = 'ОАО Тестовая компания'
city_name = 'Москва'
org_inn = "1234"
org_kpp = "5678"
org_ogrn = '91011'

def call(url, method, params=None, args=None, fields="*", filter=None,
flags={'admin_mode': True}):
    if fields == "*":
        fields = ['**']
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "args": args,
        "kwargs": params,
        "fields": fields,
        "filter": filter,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()


session = requests.session()
session.headers['Authorization'] = f'Bearer {token}'


kwargs = {'name': org_name,
          'is_internal': True,
          'city': city_name,
```

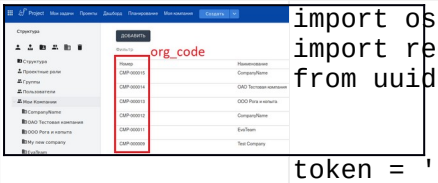
```
          'inn': org_inn,
          'kpp': org_kpp,
          'ogrn': org_ogrn
         }


a = call(domain_eva, 'CmfCompany.create', params=kwargs)
```

Пример KB-000247. Смена статуса у всех задач с типом task.base:default созданных за указанный промежуток времени

```
import os
import requests
from uuid import uuid4

token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'
start_date = '2024-4-10'
end_date = '2024-4-12'
status_code = 'closed'
lt_code = 'task.base:default'


def call(url, method, params=None, args=None, fields="*", filter=None,
flags={'admin_mode': True}):
    if fields == "*":
        fields = ['**']
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "args": args,
        "kwargs": params,
        "fields": fields,
        "filter": filter,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()


session = requests.session()
session.headers['Authorization'] = f'Bearer {token}'

lt = call(domain_eva, 'CmfLogicType.get', params={'filter': ['code', '==', lt_code]})
lt_id = lt['result']['id']
tasks = call(domain_eva, 'CmfTask.list', params={'filter': [['cmf_created_at', '>=',
start_date], ['cmf_created_at', '<=', end_date], ['logic_type', '==', lt_id]]},
             fields=['status'])

for task in tasks['result']:
    task_code = task['code']
    task_id = task['id']
    task_name = task['name']
```

```
    r = call(domain_eva, 'CmfTask.update', params={'status': status_code},
args=[task_id])
    print(f'Закрываем задачу "{task_name}" - "{task_code}"')
```

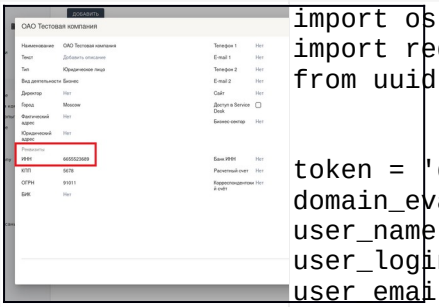
Пример KB-000249. Создание пользователя по API с указанием кода контрагента

Код организации	Код API запроса
	

```
import os
import requests
from uuid import uuid4


token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'
user_name = 'Иванов Иван Иванович'
user_login =  'test@evateam.ru'
user_email = 'test@evateam.ru'
org_code = 'CMP-000015'

def call(url, method, params=None, args=None, fields="*",
filter=None, flags={'admin_mode': True}):
    if fields == "*":
        fields = ['**']
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "args": args,
        "kwargs": params,
        "fields": fields,
        "filter": filter,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()


session = requests.session()
session.headers['Authorization'] = f'Bearer {token}'

find_org = call(domain_eva, 'CmfCompany.get', params={'filter':
['code', '==', org_code]})

if not find_org['result']:
    print(f'Не найден контрагент по коду {org_code}')
else:
    org_id = find_org['result']['id']

    params = {'name': user_name,
            'login': user_login,
            'user_local': True,
            'email': user_email,
            'company': { 'id': org_id}
            }
```

```
    a = call(domain_eva, 'CmfPerson.create', params=params)
```

Пример KB-000251. Создание пользователя по API с указанием ИНН контрагента

ИНН организации	Код API запроса
	`import os` `import requests` `from uuid import uuid4` `token = 'gzz077755RRRIYxxxu50'` `domain_eva = 'https://test.evateam.ru/'` `user_name = 'Иванов Иван Иванович'` `user_login =  'test@evateam.ru'` `user_email = 'test@evateam.ru'` `org_inn = '6655523689'` `def call(url, method, params=None, args=None, fields="*",` `filter=None, flags={'admin_mode': True}):` `    if fields == "*":` `        fields = ['**']` `    payload = {` `        "callid": str(uuid4()),` `        "method": method,` `        "args": args,` `        "kwargs": params,` `        "fields": fields,` `        "filter": filter,` `        "flags": flags,` `        "jsonrpc": "2.2"` `    }` `    if not fields:` `        payload.pop('fields')` `    if not params:` `        payload.pop('kwargs')` `    if not filter:` `        payload.pop('filter')` `    if not args:` `        payload.pop('args')` `    resp = session.post(os.path.join(url, 'api/'), json=payload)` `    assert resp.status_code == 200, resp.content` `    return resp.json()` `session = requests.session()` `session.headers['Authorization'] = f'Bearer {token}'` `find_org = call(domain_eva, 'CmfCompany.get', params={'filter':` `['inn', '==', org_inn]})` `if not find_org['result']:` `    print(f'Не найден контрагент с ИНН {org_inn}')` `else:` `    org_id = find_org['result']['id']` `    params = {'name': user_name,` `            'login': user_login,` `            'user_local': True,` `            'email': user_email,` `            'company': { 'id': org_id}`

```
                }
    a = call(domain_eva, 'CmfPerson.create', params=params)
```

Пример KB-000257. Получить все комментарии из истории изменения задачи

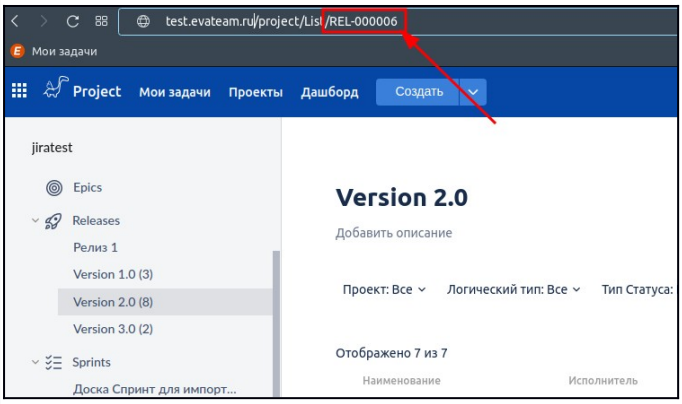
```
import requests
import os
from uuid import uuid4, UUID

token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'
#Указываем код задачи
code_task ='PERV-63'

def call(url, method, params=None, args=None, fields="*", filter=None,
flags={'admin_mode': True}):
    if fields == "*":
        fields = ['**']
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "args": args,
        "kwargs": params,
        "fields": fields,
        "filter": filter,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()
session = requests.session()
session.headers['Authorization'] = f'Bearer {token}'

task = call(domain_eva, 'CmfTask.get', params={'filter': ['code', '==', code_task],
'fields': ['comments.*']})
comments = task['result']['comments']
for i in comments:
#Выводим дату события и текст
    date = i["cmf_created_at"]
    text = i["text"]
    print(f'{date} - {text}')
```

Пример KB-000258. Назначение исполнителя и ждем ответа в задаче со сменой статуса по API

```
mport os
import requests
from uuid import uuid4
```

```
token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'
task_code = 'SD-136'
cmf_owner_login = 'test@test.ru'
status_code = 'STC-000002'


def call(url, method, params=None, args=None, fields="*", filter=None,
flags={'admin_mode': True}):
    if fields == "*":
        fields = ['**']
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "args": args,
        "kwargs": params,
        "fields": fields,
        "filter": filter,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()


session = requests.session()
session.headers['Authorization'] = f'Bearer {token}'

# Находим id задачи по коду
task = call(domain_eva, 'CmfTask.get', params={'filter':['code', '==', task_code],
'fields':['*']})
if not task['result']:
    print(f'Не найдена задача по коду {task_code}')

task_id = task['result']['id']

# Находим id пользователя по логину
person = call(domain_eva, 'CmfPerson.get', params={'filter':['login', '==',
cmf_owner_login], 'fields':['*']})
if not person['result']:
    print(f'Не найден пользователь по логину {cmf_owner_login}')

person_id = person['result']['id']

params = {
    'responsible': {
        'id': person_id
    },
    'waiting_for': {
        'id': person_id
    },
    'status': status_code
}
a = call(domain_eva, 'CmfTask.update', params=params, args=[task_id])
```

Пример KB-000269. Получить информацию по задачам, которые находятся в определенном релизе. Включая архивные задачи

Код релиза	
	``` import os import requests from uuid import uuid4 token = 'gzz077755RRRIYxxxu50' domain_eva = 'https://test.evateam.ru/' release_code = 'REL-000006' def call(url, method, params=None, args=None, f if fields == "*": fields = ['**'] payload = { "callid": str(uuid4()), "method": method, "args": args, "kwargs": params, "fields": fields, "filter": filter, "flags": flags, "jsonrpc": "2.2" } if not fields: payload.pop('fields') if not params: payload.pop('kwargs') if not filter: payload.pop('filter') if not args: payload.pop('args') resp = session.post(os.path.join(url, 'api/ assert resp.status_code == 200, resp.conter return resp.json() session = requests.session() session.headers['Authorization'] = f'Bearer {to # Находим id релиза по коду fix = call(domain_eva, 'CmfList.get', params={' if not fix['result']: print(f'Не найден релиз по коду {release_co fix_id = fix['result']['id'] tasks = call(domain_eva, 'CmfTask.list', params for task in tasks['result']: task_code = task['code'] task_name = task['name'] task_resp = task['responsible']['name'] if task_link = f'{domain_eva}project/Task/{tas print(f'{task_code} - {task_name} - {task_r ```

Пример KB-000275. Получить код, проект, автора, исполнителя, тип обращения, даты создания и закрытия по задачам в определенном проекте

```
import requests
import os
from uuid import uuid4, UUID
import json

url = 'https://test.evateam.ru/'
method = 'POST'
code_project = 'serv-desk'
token = 'gzz077755RRRIYxxxu50'

def call(url, method, params=None, args=None, fields="*", filter=None,
flags={'admin_mode': True}):
    if fields == "*":
        fields = ['**']
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "args": args,
        "kwargs": params,
        "fields": fields,
        "filter": filter,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()
session = requests.session()
session.headers['Authorization'] = f'Bearer {token}'

result_get_project = call(url, 'CmfProject.get', params={'filter':['code', '==',
code_project]}, fields=['id', 'name'])
project_id = result_get_project['result']['id']
project_name = result_get_project['result']['name']
result_get_tasks = call(url, 'CmfTask.list', params={'filter': [['parent_id', '==',
project_id],['system', '==', False]], 'fields':['cmf_author.name',
'cmf_author.login', 'request_type.name', 'responsible.name', 'cmf_created_at',
'status_closed_at']})

for task in result_get_tasks['result']:
        task_code = task['code'] #Код задачи
        task_author_name = task['cmf_author']['name'] or 'Автор не найден или удален'
        task_author_login = task['cmf_author']['login'] or ''
        task_request_type_name = task['request_type']['name'] if task['request_type']
else ''
        task_responsible_name = task['responsible']['name'] if task['responsible']
else 'Исполнитель не назначен'
        task_responsible_login = task['responsible']['login'] if task['responsible']
else ''
        task_create_date = task['cmf_created_at']
        task_closed_date = task['status_closed_at'] or ''
        result_dict = { "Код задачи":task_code, "Проект":project_name, "Имя автора":
task_author_name, "Логин автора": task_author_login,
```

```
                        "Тип обращения": task_request_type_name, "Дата создания":
task_create_date, "Дата закрытия": task_closed_date, "Имя исполнителя":
task_responsible_name, "Логин исполнителя": task_responsible_login}
        result_json = json.dumps(result_dict, ensure_ascii=False)
        print(result_json)
```

Пример KB-000281. Получить код, имя задачи, код статуса, имя статуса и дату перехода в статус по задачам из определенного проекта, которые созданы в определенный период

```
import requests
import os
from uuid import uuid4

domain = "https://test.evateam.ru/"
""" Адрес системы"""

token = "gzz077755RRRIYxxxu50"
""" Ваш сгенерированный API токен."""

code_project = "codeexample"
""" Код проекта"""

tasks_start_at = "2024-07-8"
""" Начальная дата отбора"""

tasks_end_at = "2024-07-10"
""" Конечная дата отбора"""


def call(url, method, kwargs, session, flags={"admin_mode": True}):
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "kwargs": kwargs,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    resp = session.post(os.path.join(url, "api/"), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()


def run():
    with requests.session() as session:
        session.headers["Authorization"] = f"Bearer {token}"
        get_project_call = call(domain, "CmfProject.get", kwargs={"filter": ["code",
"==", code_project]},
                                session=session)
        project_result = get_project_call.get("result")
        project_id = project_result.get("id")
        get_task_list_call = call(url=domain,
                                  method="CmfTask.list",
                                  kwargs={"filter": [["project_id", "==",
project_id],
                                                     ["cmf_created_at", ">=",
tasks_start_at],
                                                     ["cmf_created_at", "<=",
tasks_end_at]],
                                          "fields": ["id"]},
                                  session=session)
        task_list_call_result = get_task_list_call.get("result")
        task_id_list = list(map(lambda task: task.get("id"), task_list_call_result))
```

```
        status_history_list = call(url=domain,
                                   method="CmfStatusHistory.list",
                                   kwargs={"filter": ["obj_id", "IN", task_id_list],
                                           "fields": ["to_status_code",
"to_status_name", "cmf_created_at",
                                                      "obj.name", "obj.code"]},
                                   session=session)
        status_history_list_result = status_history_list.get("result")
        result_list = ["Код задачи;Имя задачи;Код статуса;Имя статуса;Дата попадания
в статус"]
        for status in status_history_list_result:
            task_name = status.get("obj").get("name")
            task_code = status.get("obj").get("code")
            to_status_code = status.get("to_status_code", None)
            to_status_name = status.get("to_status_name", None)
            cmf_created_at = status.get("cmf_created_at", None)
            result = f"{task_code};{task_name};{to_status_code};{to_status_name};
{cmf_created_at}"
            result_list.append(result)
    return result_list

call_result = run()
print(*call_result, sep='\n')
```

Пример KB-000293. Списание времени по задаче

Данный запрос списывает указанное количество времени по указанной задаче.

```
import datetime
import requests
import os
from uuid import uuid4


domain = 'https://test.evateam.ru/'
token = 'gzz077755RRRIYxxxu50'
task_code = 'TEST-134'
time_spent = 15
description = 'Выполнил задачу'

def call(url, method, session, params=None, args=None, fields='*', filter=None,
no_meta=True, flags={'admin_mode': True}):
    if fields == '*':
        fields = ['**']
    payload = {
        'callid': str(uuid4()),
        'method': method,
        'args': args,
        'kwargs': params,
        'fields': fields,
        'filter': filter,
        'flags': flags,
        'no_meta': no_meta,
        'jsonrpc': '2.2'
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()

def run():
```

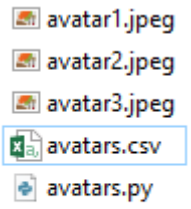
```
    if time_spent < 0:
        return
    with requests.session() as session:
        session.headers['Authorization'] = f'Bearer {token}'
        task_get_call = call(domain, 'CmfTask.get', session, params={'code':
task_code})
        task_get_call_result = task_get_call.get('result')
        args = [task_get_call_result.get('id')]
        kwargs = {
            'time_spent': time_spent,
            'remaining_estimate': 0,
            'text': description,
            'date': f'{datetime.datetime.now()}'
        }
        _ = call(domain, 'CmfTask.timetracker_change_time', session, args=args,
params=kwargs)
        print(f'В задачу "{task_code}" было списано {time_spent} минут')

run()
```

Пример KB-000305. Массовое добавление аватаров сотрудников

Данный запрос устанавливает изображения, находящиеся в одной директории со скриптом, в качестве аватаров сотрудников.

Для работы данного запроса необходимо в директории с запросом создать файл "avatars.csv" (разделитель "точка с запятой"), а также добавить нужные фотографии.

Скрипт необходимо запускать из той же директории, в которой расположены фотографии и файл avatars.csv.

Файлы в директории	Пример заполнения файла "avatars.csv"	
	eva_user1@evateam.ru;avatar3.jpeg eva_user2@evateam.ru;avatar2.jpeg eva_user3@evateam.ru;avatar1.jpeg	import os from uuid import uuid4 import base64 import requests token = 'gzz077755RRRIYxxxu50 ''' Ваш сгенерированный API т domain_eva = 'https://test.eva ''' Адрес системы.''' filename = 'avatars.csv' ''' Название файла с соответст Хранится в формате .csv с раз ''' def call(url, method, params= if fields == '*': fields = ['**'] payload = { 'callid': str(uuid4() 'method': method, 'args': args, 'kwargs': params, 'fields': fields, 'filter': filter,

```
        'flags': flags,
        'no_meta': no_meta,
        'jsonrpc': '2.2'
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.pa
    assert resp.status_code =
    return resp.json()


with requests.session() as se
    session.headers['Authoriz
    with open(filename, mode=
        for line in f:
            login, full_pictu
            user = call(domai
            user_result = use
            user_id = user_re
            with open(full_pi
                avatar = img.
            avatar_b64 = base
            result_set_code =
```

Пример KB-000321. Создание родительской связи в задаче

```
import requests
import os
from uuid import uuid4, UUID

token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'
#task_parent родитель, task_child дочерняя задач
task_parent = 'TASK-000001'
task_child = 'TASK-000002'

def call(url, method, params=None, args=None, fields="*", filter=None,
flags={'admin_mode': True}):
    if fields == "*":
        fields = ['**']
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "args": args,
        "kwargs": params,
        "fields": fields,
        "filter": filter,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
```

```
    return resp.json()


session = requests.session()
session.headers['Authorization'] = f'Bearer {token}'
#Получаем id дочерней задачи
resp = call(domain_eva, 'CmfTask.get', params={'filter':['code', '==', task_child]})
child_id = resp['result']['id']
#Обновляем дочернюю задачу
resp = call(domain_eva, 'CmfTask.update', params={'parent_task':task_parent},
args=[child_id])
print(resp['result'])
```

Пример KB-000323. Создание взаимной связи (Относится к) в задаче

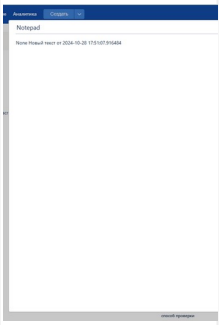
```
import requests
import os
from uuid import uuid4, UUID

token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'
#В задачу task_out добавляется связь на задачу task_in
task_out = 'TASK-000001'
task_in = 'TASK-000002'
#Код взаимной связи
link_code = 'system.link'

def call(url, method, params=None, args=None, fields="*", filter=None,
flags={'admin_mode': True}):
    if fields == "*":
        fields = ['**']
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "args": args,
        "kwargs": params,
        "fields": fields,
        "filter": filter,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()


session = requests.session()
session.headers['Authorization'] = f'Bearer {token}'
#Создаем взаимную связь с задачами
resp = call(domain_eva, 'CmfRelationOption.create', params={'out_link':task_out,
'in_link':task_in, 'relation_type':link_code})
print(resp['result'])
```

Пример KB-000325. Получить все связи задачи

```
import requests
import os
from uuid import uuid4, UUID

token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'
obj_code = 'TASK-107'

def call(url, method, params=None, args=None, fields="*", filter=None,
flags={'admin_mode': True}):
    if fields == "*":
        fields = ['**']
    payload = {
        "callid": str(uuid4()),
        "method": method,
        "args": args,
        "kwargs": params,
        "fields": fields,
        "filter": filter,
        "flags": flags,
        "jsonrpc": "2.2"
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()

session = requests.session()
session.headers['Authorization'] = f'Bearer {token}'
task = call(domain_eva, 'CmfTask.get', params={'code': obj_code, 'fields':
['out_tasks.**', 'in_tasks.**'], })
task_result = task['result']
in_task_relation = task_result.get('in_tasks')
out_task_relation = task_result.get('out_tasks')
print('Входящие связи: ', in_task_relation, '\n')
print('Исходящие связи: ', out_task_relation)
for relation_in in in_task_relation:
    code_task_in = relation_in['in_link']['code']
    print('Код задачи вход.связи: ', code_task_in, '\n')

for relation_out in out_task_relation:
    code_task_out = relation_out['out_link']['code']
    print('Код задачи исх.связи:',code_task_out)
```

Пример KB-000326. Добавить текст в блокнот пользователя по номеру телефона

Данная автоматизация позволяет добавить в блокнот пользователя произвольный текст. Пользователь определяется по номеру телефона

Код	Номер телефона в профиле пользователя	Блокнот пользователя
import datetime import os from uuid import uuid4 import requests token = 'gzz077755RRRIYxxxu50' domain_eva = 'https://test.evateam.ru/' user_phone = '79999999999;' def call(url, method, params=None, args=None, fields='*', filter=None, no_meta=True, flags={'admin_mode': True}): if fields == '*': fields = ['**'] payload = { 'callid': str(uuid4()), 'method': method, 'args': args, 'kwargs': params, 'fields': fields, 'filter': filter, 'flags': flags, 'no_meta': no_meta, 'jsonrpc': '2.2' } if not fields: payload.pop('fields') if not params: payload.pop('kwargs') if not filter: payload.pop('filter') if not args: payload.pop('args') resp = session.post(os.path.join(url, 'api/'), json=payload) assert resp.status_code == 200, resp.content return resp.json() with requests.session() as session: session.headers['Authorization'] = f'Bearer {token}' user = call(domain_eva, 'CmfPerson.get', params={'phone': user_phone}) user_result = user.get('result') notepad = call(domain_eva, 'CmfNotepad.get', params={'cmf_owner.id': user_result.get('id')}) # cmf_owner notepad_result = notepad.get('result') current_date_time = datetime.datetime.now() text_for_update = f'{notepad_result.get("text")}\n\nНовый текст от {current_date_time}' # Текст надо скорректировать notepad_update = call(domain_eva, 'CmfNotepad.update', args=[notepad_result.get('id')], params={'text': text_for_update})		

Пример KB-000327. Создание задачи с указанием плановых дат

Данный запрос позволяет создать задачу, с указанием:

1. Проекта, в который будет добавлена задача
2. Epic, к которому будет привязана задача
3. Плановых дат начала и окончания задачи
4. Названия, описания задачи, а также исполнителя по задаче

```
import os
from uuid import uuid4
import requests

token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru'

# Список параметров для заполнения
project = 'project_code'
'''Код проекта'''
epic = 'epic_code'
'''Код эпика'''
planing_start_date = '2024-10-02 12-00-00'
'''Плановая дата начала в формате гггг-мм-дд ЧЧ-ММ-СС'''
planing_finish_date = '2024-10-09 16-00-00'
'''Плановая дата окончания в формате гггг-мм-дд ЧЧ-ММ-СС'''

planing_timezone = '+3'
''' Часовой пояс в формате +-Ч'''

task_name = 'My task name'
'''Название задачи'''
description_task = 'Описание создаваемой задачи'
'''Описание задачи'''
responsible = 'eva_test@evateam.ru'
'''Логин исполнителя'''


def call(url, method, params=None, args=None, fields='*', filter=None, no_meta=True,
flags={'admin_mode': True}):
    if fields == '*':
        fields = ['**']
    payload = {
        'callid': str(uuid4()),
        'method': method,
        'args': args,
        'kwargs': params,
        'fields': fields,
        'filter': filter,
        'flags': flags,
        'no_meta': no_meta,
        'jsonrpc': '2.2'
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()
```

```
with requests.session() as session:
    session.headers['Authorization'] = f'Bearer {token}'
    task_kwargs = {
        'parent': project,
        'epic': epic,
        'name': task_name,
        'text': description_task,
        'responsible': responsible
    }
    create_task = call(domain_eva, 'CmfTask.create', params=task_kwargs)
    create_task_result = create_task.get('result')

    created_task = call(domain_eva, 'CmfTask.get', params={'filter': ['id', '==',
create_task_result],
                                                           'fields':
['op_gantt_task']})
    created_task_result = created_task.get('result')
    gant_task = created_task_result.get('op_gantt_task')
    gant_task_id = gant_task.get('id')
    gantt_task_kwargs = {
        'sched_start_date': f'{planing_start_date}{planing_timezone}',
        'sched_finish_date': f'{planing_finish_date}{planing_timezone}'
    }
    update_gantt_task = call(domain_eva, 'CmfGanttTask.update', args=[gant_task_id],
params=gantt_task_kwargs)
```

Пример KB-000332. Сбор статистики входа в систему

Данный запрос позволяет выгрузить из системы следующую информацию:

1. ФИО пользователя
2. Логин пользователя
3. Дата последнего входа в систему
4. Количество авторизаций за месяц

```
import os
from uuid import uuid4
from datetime import timedelta, datetime
import requests

token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'


def call(url, method, params=None, args=None, fields='*', filter=None, no_meta=True,
flags={'admin_mode': True}):
    if fields == '*':
        fields = ['**']
    payload = {
        'callid': str(uuid4()),
        'method': method,
        'args': args,
        'kwargs': params,
        'fields': fields,
        'filter': filter,
        'flags': flags,
        'no_meta': no_meta,
        'jsonrpc': '2.2'
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
```

```
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()


with requests.session() as session:
    session.headers['Authorization'] = f'Bearer {token}'
    last_month_as_date = datetime.now().date() - timedelta(days=30)
    last_month_as_string = last_month_as_date.strftime('%Y-%m-%d')
    all_users = call(domain_eva, 'CmfPerson.list', params={
        'filter': ['login', '!=', None],
        'fields': ['name', 'login']
    })
    all_users_result = all_users.get('result')
    statistics_table = ['ФИО;Логин;Дата последнего входа в систему;Количество
авторизаций за месяц\n']
    for user in all_users_result:
        user_login = user.get('login')
        audit_kwargs = {
            'filter': [
                ['cmf_author.login', '==', user_login],
                ['operate', '==', 'login_successed']
            ],
            'fields': ['cmf_created_at'],
            'order_by': ['-cmf_created_at']
        }
        audit = call(domain_eva, 'CmfAudit.get', params=audit_kwargs)
        audit_result = audit.get('result')
        if not audit_result:
            continue
        last_login = audit_result.get('cmf_created_at')
        last_month_kwargs = {
            'filter': [
                ['cmf_author.login', '==', user_login],
                ['operate', '==', 'login_successed'],
                ['cmf_created_at', '>=', last_month_as_string]
            ],
            'fields': ['id'],
        }
        audit_user_login_last_month = call(domain_eva, 'CmfAudit.count',
params=last_month_kwargs)
        login_time_by_last_month = audit_user_login_last_month.get('result')
        statistics_table.append(f'{user.get("name")};{user_login};{last_login};
{login_time_by_last_month}\n')
with open('statistics.csv', mode='a+') as f:
    f.writelines(statistics_table)
```

Пример KB-000334. Получение данных о пользователе из аудита

Данный запрос позволяет получить информацию о действиях пользователя из аудита системы

```
import os
from uuid import uuid4
import requests

token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'

user_login = 'test_user@evateam.ru'
''' Логин пользователя, чей аудит будем получать'''
```

```
lower_limit = 0
''' Нижний предел записей'''

upper_limit = 100
''' Верхний предел количества записей'''

slice_step = 50
''' Шаг получения записей.
    Выбирать так, чтобы верхний предел был кратен шагу получения записей.
    Чем меньше шаг, тем больше запросов будет сделано, но они будут более легкими.
'''

first_day = '2024-11-01'
''' Нижняя граница в формате %Y-%m-%d'''

last_day = '2024-11-20'
'''Верхняя граница в формате %Y-%m-%d'''


def call(url, method, params=None, args=None, fields='*', filter=None, no_meta=True,
flags={'admin_mode': True}):
    if fields == '*':
        fields = ['**']
    payload = {
        'callid': str(uuid4()),
        'method': method,
        'args': args,
        'kwargs': params,
        'fields': fields,
        'filter': filter,
        'flags': flags,
        'no_meta': no_meta,
        'jsonrpc': '2.2'
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()


with requests.session() as session:
    session.headers['Authorization'] = f'Bearer {token}'
    full_audit_result = []
    for i in range(lower_limit, upper_limit-slice_step+1, slice_step):
        slice_range = [i, i + slice_step]
        audit = call(url=domain_eva,
                     method='CmfAudit.list',
                     params={
                         'filter': [['cmf_author.login', '==', user_login],
                                    ['cmf_created_at', '>=', first_day],
                                    ['cmf_created_at', '<=', last_day]],
                         'fields': ['id', 'cmf_created_at', 'cmf_author',
'audit_data', 'operate', 'cmf_model_name',
                                    'result_status'],
                         'slice': slice_range,
                         'order_by': ['-cmf_created_at']}
                     )
        audit_result = audit.get('result')
        full_audit_result.extend(audit_result)
    print(*full_audit_result, sep='\n')
```

Пример KB-000337. Добавить пользователю документ или задачу в избранное

Данный запрос позволяет принудительно добавить указанному пользователю документ или задачу в избранное

Код запроса

```
import os
from uuid import uuid4
import requests

token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'

user_login = 'login@mail.ru'
''' Логин пользователя, которому добавляем в избранное'''

favorites_object_code = 'OBJ-81'
''' Код добавляемого в избранное объекта'''


def call(url, method, params=None, args=None, fields='*', filter=None, no_meta=True, flags
    if fields == '*':
        fields = ['**']
    payload = {
        'callid': str(uuid4()),
        'method': method,
        'args': args,
        'kwargs': params,
        'fields': fields,
        'filter': filter,
        'flags': flags,
        'no_meta': no_meta,
        'jsonrpc': '2.2'
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()


with requests.session() as session:
    session.headers['Authorization'] = f'Bearer {token}'
    user_kwargs = {
        'filter': ['login', '==', user_login],
        'fields': ['person_var']
    }
    user = call(domain_eva, 'CmfPerson.get', params=user_kwargs)
    user_res = user.get('result')
    user_pv = user_res.get('person_var')
    user_pv_id_with_favorites = user_pv[0].get('id')
    object_kwargs = {
        'filter': ['code', '==', favorites_object_code]
    }
    task_call = call(domain_eva, 'CmfTask.get', params=object_kwargs)
    task_call_result = task_call.get('result')
```

```
    document_call = call(domain_eva, 'CmfDocument.get', params=object_kwargs)
    document_call_result = document_call.get('result')
    favorites_object_id = task_call_result.get('id') if task_call_result else document_cal
    person_var = call(domain_eva, 'CmfPersonVar.favorites.append',
                      args=[user_pv_id_with_favorites, favorites_object_id])
    person_var_result = person_var.get('result')
```

Пример KB-000338. Выгрузка из аудита данных о пользователях

Данный запрос позволяет выгрузить в формате ".csv" следующую информацию о пользователях из аудита системы:

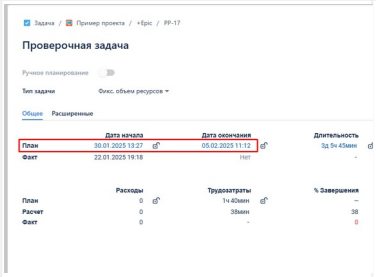
1. Логин пользователя
2. Дата последнего входа в систему
3. Входит ли пользователь в указанную группу
4. Наличие у пользователя параметра "Не работает\уволен"

```
import os
import csv
from uuid import uuid4
from datetime import timedelta, datetime
import requests

token = 'gzz077755RRRIYxxxu50'
domain_eva = 'https://test.evateam.ru/'

user_group_code = 'Admins'
''' Код группы в которой должен находиться пользователь'''

def call(url, method, params=None, args=None, fields='*', filter=None, no_meta=True,
flags={'admin_mode': True}):
    if fields == '*':
        fields = ['**']
    payload = {
        'callid': str(uuid4()),
        'method': method,
        'args': args,
        'kwargs': params,
        'fields': fields,
        'filter': filter,
        'flags': flags,
        'no_meta': no_meta,
        'jsonrpc': '2.2'
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'), json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()


with requests.session() as session:
    session.headers['Authorization'] = f'Bearer {token}'
    statistics_table = [
```

```
        ['Логин', 'Дата последнего входа в систему', 'Наличие у пользователя группы',
         'Наличие у пользователя галочки "Не работает/уволен"']
    ]
    all_users = call(domain_eva, 'CmfPerson.list', params={'filter': ['login', '!=',
None],
                                                           'fields': ['rg_member_of',
'does_not_work']})
    all_users_result = all_users.get('result')
    for user in all_users_result:
        user_login = user.get('login')
        audit_kwargs = {
            'filter': [
                ['cmf_author.login', '==', user_login],
                ['operate', '==', 'login_successed']
            ],
            'fields': ['cmf_created_at'],
            'order_by': ['-cmf_created_at']
        }
        audit = call(domain_eva, 'CmfAudit.get', params=audit_kwargs)
        audit_result = audit.get('result')
        last_login = audit_result.get('cmf_created_at') if audit_result else '-'
        user_groups = user.get('rg_member_of')
        user_group_filter = list(filter(lambda x: x.get('code') == user_group_code,
user_groups))
        user_in_group = True if user_group_filter else False
        user_doesnt_work_checkbox = user.get('does_not_work')
        statistics_string = [user_login, last_login, user_in_group,
user_doesnt_work_checkbox]
        statistics_table.append(statistics_string)

    with open('statistics.csv', mode='a+', newline='') as f:
        csv_writer = csv.writer(f, delimiter=';')
        csv_writer.writerows(statistics_table)
```

Пример KB-000346. Получение плановых дат начала и окончания задачи

Данный запрос позволяет получить плановые даты начала и окончания задачи

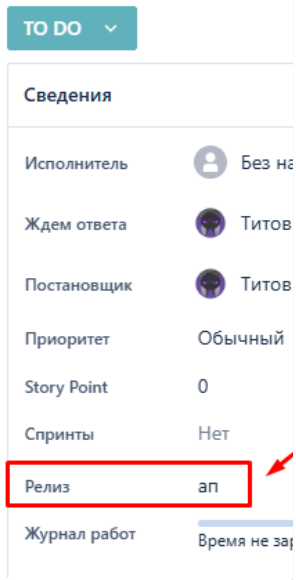
Плановые даты в задаче	Код запроса
	`import os` `from uuid import uuid4` `import requests` `import urllib3` `urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)` `token = 'gzz077755RRRIYxxxu50'` `""" Ваш сгенерированный API токен."""` `domain_eva = 'https://test.evateam.ru/'` `""" Адрес системы."""` `task_code = 'TEST-120'` `''' Код задачи'''` `def call(url, method, params=None, args=None, fields='*', filter=None, no_meta=True, flags={'admin_mode': True}):` `    if fields == '*':` `        fields = ['**']` `    payload = {` `        'callid': str(uuid4()),` `        'method': method,` `        'args': args,` `        'kwargs': params,`

	`` 'fields': fields, 'filter': filter, 'flags': flags, 'no_meta': no_meta, 'jsonrpc': '2.2' } if not fields: payload.pop('fields') if not params: payload.pop('kwargs') if not filter: payload.pop('filter') if not args: payload.pop('args') resp = session.post(os.path.join(url, 'api/'), json=payload) assert resp.status_code == 200, resp.content return resp.json() with requests.session() as session: session.headers['Authorization'] = f'Bearer {token}' task_call_kwargs = { 'code': task_code, 'fields': ['op_gantt_task.sched_start_date', 'op_gantt_task.sched_finish_date'] } task_call = call(domain_eva, 'CmfTask.get', params=task_call_kwargs) task_call_result = task_call.get('result') gantt_task = task_call_result.get('op_gantt_task') sched_start_date = gantt_task.get('sched_start_date') sched_finish_date = gantt_task.get('sched_finish_date') print(f'Плановая дата начала: {sched_start_date}\ nПлановая дата окончания: {sched_finish_date}') ```

Пример KB-000353. Добавление релиза в задачу через API

Данный API запрос позволяет добавить релиз в задачу. Если в задаче уже был назначен релиз, то новый релиз не заменит его, а добавится вторым.

Релиз в задаче	Код запроса




```
import os
from uuid import uuid4
import requests

token = 'gzz077755RRRIYxxxu50'
''' Токен для API запросов'''
domain_eva = 'https://test.evateam.ru/'
''' Домен системы'''
task_code = 'HEHE-86'
''' Код задачи'''
release_code = 'REL-000092'
''' Код релиза'''


def call(url, method, params=None, args=None, fields='*',
filter=None, no_meta=True, flags={'admin_mode': True}):
    if fields == '*':
        fields = ['**']
    payload = {
        'callid': str(uuid4()),
        'method': method,
        'args': args,
        'kwargs': params,
        'fields': fields,
        'filter': filter,
        'flags': flags,
        'no_meta': no_meta,
        'jsonrpc': '2.2'
    }
    if not fields:
        payload.pop('fields')
    if not params:
        payload.pop('kwargs')
    if not filter:
        payload.pop('filter')
    if not args:
        payload.pop('args')
    resp = session.post(os.path.join(url, 'api/'),
json=payload)
    assert resp.status_code == 200, resp.content
    return resp.json()


with requests.session() as session:
    session.headers['Authorization'] = f'Bearer {token}'
    call_task = call(domain_eva, 'CmfTask.get',
params={'code': task_code})
    call_task_res = call_task.get('result')
    task_id = call_task_res.get('id')
    call_rel = call(domain_eva, 'CmfList.get',
params={'code': release_code})
    call_rel_res = call_rel.get('result')
    rel_id = call_rel_res.get('id')
    update_task_release = call(domain_eva,
'CmfTask.fix_versions.append', args=[task_id, rel_id])
```