ООО “ТРИДИВИ”

OMNI Platform 1.11.0

Справочник по API

Миасс - 2023

Содержание

1 Объектная модель

Sample (Семпл) - это объект в формате JSON, в котором реализуется и хранится результат обработки изображения (изображение лица, атрибуты лица и биометрический шаблон), предназначенный для распознавания.

Profile (Профиль) - это объект, который содержит метаданные о человеке, его изображение (аватар), активности и главный семпл. Главный семпл представляет собой биометрический шаблон, используемый агентом для определения наличия профиля в базе и сервером для определения того, в какой профиль добавлять новые активности.

Profile Group (Группа) - это объект, который содержит набор идентификаторов профилей и метаданные о группе. Используется для группировки профилей и получения оповещений, т.к. именно к этому объекту прикрепляется триггер с условием создать и отправить оповещение, в случае если созданная активность принадлежит профилю из этой группы.

Activity (Активность) - это объект, в котором хранится информация о любом обнаруженном присутствии человека в поле зрения камеры. Динамические данные такого типа передаются агентами и хранятся в форме процессов. При появлении человека в кадре, агент делает снимок, проверяет его на качество, определяет человека на снимке и собирает данные активности. Далее происходит сравнение семпла активности с главными семплами профилей на предмет схожести биометрического шаблона.

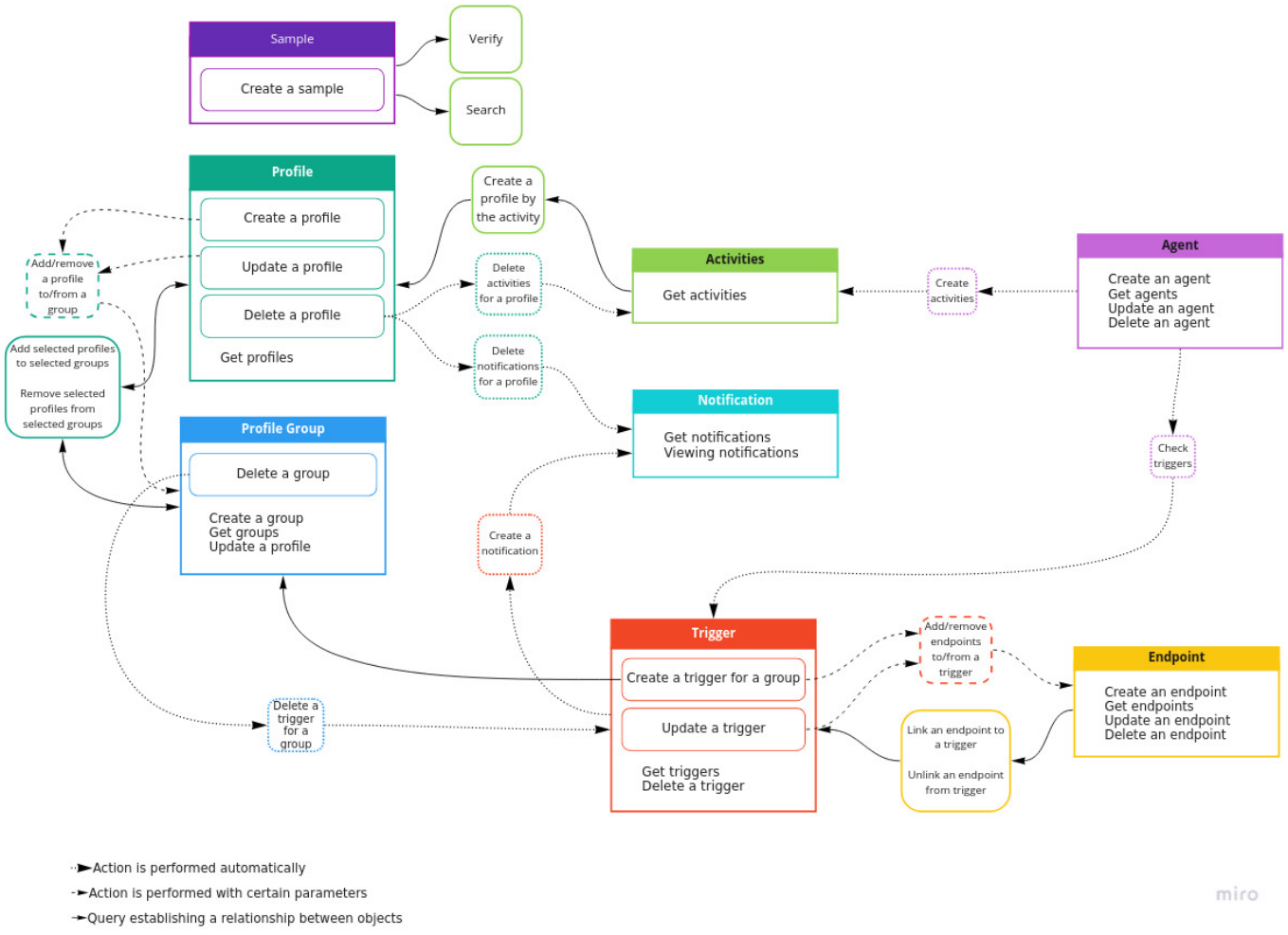
Notification (Оповещение) - это объект, который используется для оповещения пользователя в режиме реального времени об активности профиля, который был добавлен в список наблюдения.

Endpoint (Эндпоинт) - это объект, который содержит данные о месте доставки оповещений.

Trigger (Триггер) - это объект, который содержит условие срабатывания, набор эндпоинтов и идентификатор списка наблюдения, к которому прикреплен. Триггер

необходим для создания и отправки оповещения на эндпоинты, если информация с активности удовлетворяет условию триггера.

Agent (Агент) - это объект, в котором хранятся результаты обработки данных с периферийного устройства (камеры).



2 Интеграционный API

Все доступные операции с объектами OMNI Platform можно реализовать с помощью интеграционного API. Доступ к API обеспечивается через GraphQL - язык запросов и манипулирования данными API с открытым исходным кодом, а также среды выполнения существующих запросов данных. GraphQL позволяет отправлять запросы на OMNI Platform и получать ответные данные, которые можно интегрировать в ваше собственное приложение.

Более подробно о GraphQL смотрите по ссылкам:

- Введение в GraphQL
- Как пользоваться GraphQL
- Руководства и практические рекомендации

Для начала работы с API войдите в свой аккаунт на OMNI Platform. На главной странице веб-интерфейса нажмите на кнопку Platform API в виджете *Ресурсы* для перехода в интерактивную консоль GraphQL.

Для использования интеграционного API в вашем приложении добавьте токен в заголовок HTTP-запроса и отправьте запрос на https://cloud.3divi.ai/api/v2/ .

Для получения токена выполните следующие шаги:

1. Откройте интерактивную консоль GraphQL, кликнув на кнопку *Platform API* в виджете *Ресурсы* на главной странице веб-интерфейса.
2. Скопируйте указанный ниже запрос в консоль и нажмите *Выполнить запрос*.

```
{
  me {
    workspaces {
      accesses {
        id
      }
    }
  }
}
```

3. В результате GraphQL вернет ответ, который содержит токен (id):

```
{
   "data":{
      "me":{
         "workspaces":[
            {
               "accesses ":[
                  {
                     "id":"b3a4f990-d2d0-4989-8684-41cb88f3d0f9"
                  }
               ]
            }
         ]
      }
   }
}
```

Пользователь, зарегистрированный под PLATFORM_DEFAULT_EMAIL может получить токен доступа к API через следующую команду:

```
$ ./setup/get-token.sh
```

С более подробной информацией по получению и использованию токена можно ознакомиться в Руководстве администратора.

Помимо GraphQL запросы можно отправлять с помощью cURL. cURL-шаблон для отправки API запросов указан ниже:

```
curl --location --request POST "<url>" --header "token: <your access token>" --header
"Content-Type: application/json" --data-raw "{\"query\":\"<GraphQL query or
mutation>\",\"variables\":{<GraphQL variables>}}"
```

где `<url>` - URL вашего сервера, а `<your access token>` - токен доступа. Без указания токена доступа ваши cURL-запросы не смогут быть отправлены.

Пример cURL-запроса: С помощью данного запроса можно получить список 5 первых профилей из базы.

```
curl --location --request POST "<url>" --header "token:
b3a4f990-d2d0-4989-8684-41cb88f3d0f9" --header "Content-Type: application/json"
--data-raw "{\"query\":\"query {profiles(limit: 5, offset: 0) {totalCount,
collectionItems {id, info}}}\",\"variables\":{}}"
```

API возвращает следующий результат:


```
{
  "data": {
    "profiles": {
      "totalCount": 5,
      "collectionItems": [
        {
          "id": "195ed5fe-580e-476f-8496-befdb0003d85",
          "info": {
            "age": 46,
            "gender": "MALE",
            "avatar_id": "9945328f-ebd8-4683-bde5-25284686f439",
            "main_sample_id": "83800c22-5ed3-4c9f-91bf-ce79c0de747a"
          }
        },
        {
          "id": "8d46bf22-5d24-4e4f-bfd3-e215cbde53f0",
          "info": {
            "age": 24,
            "gender": "FEMALE",
            "avatar_id": "1e7c8293-75e8-485a-8861-415eec854011",
            "main_sample_id": "6cbd4c9a-7cd0-4c43-9561-387be9dff6f3"
          }
        },
        {
          "id": "944ab599-bd6d-47bf-b8e3-ed201a4e6530",
          "info": {
            "age": 31,
            "gender": "MALE",
            "avatar_id": "9f36bee0-efdf-42fc-a059-c60d5c7e6da9",
            "main_sample_id": "db9998a4-4331-41b5-9abd-30499b881be8"
          }
        },
        {
          "id": "aa40fb31-96d3-421b-96ae-05566fc57bee",
          "info": {
            "age": 35,
            "gender": "MALE",
            "avatar_id": "ae61ec9f-2953-4f24-86d8-7718df6f9ff5",
            "main_sample_id": "33a72136-175b-4361-8444-e1e0c4ce04d4"
          }
        },
        {
          "id": "c5e1b7d9-b446-4739-a3d0-a8cf8d717c70",
          "info": {
            "age": 21,
            "gender": "MALE",
            "avatar_id": "9938407f-d101-4005-8d30-4b1ce8319bc7",
            "main_sample_id": "f2b3d23a-6f73-4d9b-97aa-13edaf5e82f7"
          }
        }
      ]
    }
  }
}
```


Запросы и ответы для тестирования API в GraphQL указаны ниже.

2.1 Распознавание лиц

2.1.1 Общие ошибки входных данных

Ошибки при загрузке изображения через API:

Неверная строка base64:

```
{
  "message": "image file is truncated (21 bytes not processed)"
}
```

Изображение представлено не в формате base64:

```
{
  "message": "Expected value of type 'CustomBinaryType', found \"wrong_data\";
}
```

Изображение не передано:

```
{
  "message": "Sample Data is not valid",
  "code": "0xc69c44d4"
}
```

Неверный формат изображения:

```
{
  "message": "Image decode failed"
}
```

Изображение без лиц:

```
{
  "message": "No faces found",
  "code": "0x95bg42fd"
}
```

Слишком большой размер изображения:

```
Bad Request (400)
```

Ошибки при фильтрации, пагинации и сортировке сущностей:

Переданные фильтры не поддерживают формат словаря:

```
{
  "message": "'str' object has no attribute 'keys'"
}
```

Неверное поле фильтрации:

```
{
  "message": "Cannot resolve keyword '' into field. Choices are: creation_date, id,
info, last_modified, link_to_label, person, person_id, profile_groups, samples,
workspace, workspace_id"
}
```

Неверное поле для фильтрации подобъектов или фильтрации поля по функции:

```
{
  "message": "Unsupported lookup '1' for UUIDField or join on the field not
permitted."
}
```

Отрицательное значение в поле пагинации:

```
{
  "message": "Negative indexing is not supported."
}
```

Неверное поле сортировки:

```
{
  "message": "Cannot resolve keyword '' into field. Choices are: creation_date, id,
info, last_modified, link_to_label, person, person_id, profile_groups, samples,
workspace, workspace_id"
}
```

Другие общие ошибки:

ID объекта не задан в формате UUID:

```
{
  "message": "“%(value)s” is not a valid UUID."
}
```

Переданный JSON не прошел валидацию по JSON-схеме:

```
{
  "message": "Invalid JSON request"
}
```

2.1.2 Детекция лиц

Этот запрос позволяет обнаружить несколько лиц на изображении и получить информацию об этих лицах (например, пол, возраст, эмоции, liveness, наличие маски и т. д.). В качестве ответа вы получаете результат детекции без возможности его сохранения в локальной базе данных.

Внимание! Поддержка данного API-запроса действует до 2024 года.

```
detect(
image: CustomBinaryType!
pupils: [EyesInput!] = null): JSON!
```

image: CustomBinaryType! : Изображение в формате base64

pupils: [EyesInput!] : Для повышения точности детекции можно указать X и Y координаты зрачков.

EyesInput!

- **leftPupil: PointInputType!**
  - x: Float!
  - y: Float!
- **rightPupil: PointInputType!**
  - x: Float!
  - y: Float!

JSON! : API возвращает семпл в формате JSON со следующими параметрами:

- **image :** Изображение в формате base64
- **id :** Порядковый номер лица на изображении
- **class :** Имя класса объекта, например, лицо
- **template :** Уникальный набор биометрических характеристик, извлеченных из изображения лица. Шаблоны используются для сравнения двух изображений лиц и определения степени сходства.
- **bbox :** Прямоугольная рамка вокруг лица, которая показывает положение и размер лица на исходном изображении.
- **crop image :** Система обрезает загруженное изображение с целью извлечения изображений лиц. Такое извлеченное изображение лица и называется crop image.
- **keypoints :** Набор из 21 точки лица, указывающих на положение частей лица.
- **rotation angles :** Углы вращения: Yaw (вращение по оси Z), pitch (вращение по оси Y) и roll (вращение по оси X)Алгоритм позволяет обнаруживать лица в следующем диапазоне углов: yaw [-60; 60], pitch [-60; 60], roll [-30; 30]

- **age :** Возраст в годах
- **gender:** Пол персоны на изображении
- **emotions :** Эмоции по степени достоверности
- **liveness :** Встроенный liveness-алгоритм, который позволяет определить, что на фотографии изображен реальный человек.
- **mask presence :** Параметр наличия маски на лице.

Пример запроса:

```
{
  detect(image: "your image in Base64 format")
}
```

Пример ответа:


```
{
  "data": {
    "detect": {
      "$image": "/9j/4AAQSkZJRgABAQAAAQABAAD/2wCEAAUDBAQEAwUE***",
      "objects@common_capturer_uld_fda": [
        {
          "id": 1,
          "class": "face",
          "templates": {
            "$template10v100": "T5JnWgcfMPHQANMdP8NA8FwPQJLc5TDfcJFVFS/tYgEw9yN***"
          },
          "bbox": [
            0,
            0.21308482869466144,
            1,
            0.9869169769287109
          ],
          "$cropImage": "/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAgGBgcGBQgH***",
          "keypoints": {
            "left_eye_brow_left": {
              "x": 0.1554061550564236,
              "y": 0.3466766866048177
            },
            "left_eye_brow_up": {
              "x": 0.25802788628472223,
              "y": 0.3236322784423828
            },
            "left_eye_brow_right": {
              "x": 0.3854702419704861,
              "y": 0.3430420430501302
            },
            "right_eye_brow_left": {
              "x": 0.6136109754774306,
              "y": 0.345932362874349
            },
            "right_eye_brow_up": {
              "x": 0.7370150417751736,
              "y": 0.3286321258544922
```

```
        },
        "right_eye_brow_right": {
          "x": 0.8359405517578125,
          "y": 0.35814239501953127
        },
        "left_eye_left": {
          "x": 0.21608330620659721,
          "y": 0.4157813008626302
        },
        "left_pupil": {
          "x": 0.30088453504774304,
          "y": 0.4109149678548177
        },
        "left_eye_right": {
          "x": 0.3838650173611111,
          "y": 0.4230572001139323
        },
        "right_eye_left": {
          "x": 0.6079323323567708,
          "y": 0.4287389628092448
        },
        "right_pupil": {
          "x": 0.6891607666015624,
          "y": 0.4170384724934896
        },
        "right_eye_right": {
          "x": 0.7724110243055555,
          "y": 0.4233682759602865
        },
        "left_ear_bottom": {
          "x": 0.1043272230360243,
          "y": 0.5988115437825521
        },
        "nose_left": {
          "x": 0.3953646511501736,
          "y": 0.5805702209472656
        },
        "nose": {
          "x": 0.4884576416015625,
          "y": 0.5780504862467448
        },
        "nose_right": {
          "x": 0.5848392740885416,
          "y": 0.5861359659830729
        },
        "right_ear_bottom": {
          "x": 0.8565327962239583,
          "y": 0.607865956624349
        },
        "mouth_left": {
          "x": 0.33830030653211807,
          "y": 0.6955980936686198
        },
        "mouth": {
          "x": 0.4904075113932292,
          "y": 0.7090469868977864
        },
        "mouth_right": {
          "x": 0.6333636474609375,
          "y": 0.6958428955078125
        },
```

```
            "chin": {
              "x": 0.4895579020182292,
              "y": 0.8763695271809896
            }
          },
          "age": 23,
          "emotions": {
            "neutral": 0.9834117889404297,
            "angry": 0.013813868165016174,
            "happy": 0.002322095213457942,
            "surprised": 0.00045228638919070363
          },
          "gender": "FEMALE",
          "liveness": {
            "value": "FAKE",
            "confidence": 0.801980197429657
          },
          "angles": {
            "yaw": -0.8611235618591309,
            "pitch": -15.951218605041504,
            "roll": 1.7437981367111206
          },
          "mask": {
            "value": false,
            "confidence": 1
          }
        }
      ]
    }
  }
}
```


2.1.3 Создание семпла

Мутация позволяет создавать семплы с атрибутами лиц (пола, возраста, эмоций, ключевых точек, liveness, наличия маски и т.д.).

```
createSample(
anonymousMode: Boolean = false
image: CustomBinaryType = null
pupils: [EyesInput!] = null
sampleData: JSON = null): [SampleOutput!]!
```

anonymousMode: Boolean = false : При работе с анонимными данными можно установить для anonymousMode значение true (по умолчанию установлено значение false). В этом случае изображение не будет храниться на OMNI-сервере.

image: CustomBinaryType : Изображение в формате base64

pupils: [EyesInput!] : Для повышения точности детекции лиц можно установить X и Y координаты зрачков.

EyesInput!

- **leftPupil: PointInputType!**

  - x: Float!

- y: Float!

- **rightPupil: PointInputType!**

- x: Float!

- y: Float!

sampleData: JSON : Результат детекции лица, не сохраненный в базе.

SampleOutput! : API возвращает JSON-файл со следующими параметрами:

- **id :** ID семпла.
- **creationDate :** Дата создания семпла в формате ISO 8601 с часовыми поясами.
- **lastModified :** Дата последнего изменения семпла в формате ISO 8601 с часовыми поясами.
- **data :** Изображение и/или шаблон и/или результат детекции в формате семпла.

Возвращенный семпл автоматически сохраняется на OMNI-сервере (если для параметра anonymousMode установлено значение false) и может использоваться для верификации лиц или поиска людей по базе.

Ошибки входных данных:

Отсутствуют входные данные для создания семпла:

```
{
  "message": "One of the parameters sampleData or sourceImage is required",
  "code": "0xnf5825dh"
}
```

Переданы неверные данные семпла:

```
{
  "message": "argument should be a bytes-like object or ASCII string, not 'NoneType'"
}
```

Переданы неверные координаты зрачков:

```
{
  "message": "0x8905a659: Assertion '( transform_m(0, 0) * transform_m(0,
0) + transform_m(0, 1) * transform_m(0, 1) + transform_m(1, 0) * transform_m(1,
0) + transform_m(1, 1) * transform_m(1, 1) ) > 1e-5' failed, error code:
0x8905a659. wrap code: 0x7df96daf."
}
```

Пример запроса:

```
mutation {
  createSample(image/sampleData: "your image in Base64 format or sample data") {
    id
    creationDate
    lastModified
    data
```

```
  }}
```

Пример ответа:


```
{
  "data": {
    "createSample": [
      {
        "id": "3f7352c9-94be-4449-aaa0-b93a3812e1c6",
        "creationDate": "2022-04-28T06:23:45.902315+00:00",
        "lastModified": "2022-04-28T06:23:46.450993+00:00",
        "data": {
          "$image": {
            "id": "3403c21e-44e1-4ea4-9e57-2d2c0933861e"
          },
          "objects@common_capturer_uld_fda": [
            {
              "id": 1,
              "age": 23,
              "bbox": [
                0,
                0.21308482869466144,
                1,
                0.9869169769287109
              ],
              "mask": {
                "value": false,
                "confidence": 1
              },
              "class": "face",
              "angles": {
                "yaw": -0.8611235618591309,
                "roll": 1.7437981367111206,
                "pitch": -15.951218605041504
              },
              "gender": "FEMALE",
              "emotions": {
                "angry": 0.013813868165016174,
                "happy": 0.002322095213457942,
                "neutral": 0.9834117889404297,
                "surprised": 0.00045228638919070363
              },
              "liveness": {
                "value": "FAKE",
                "confidence": 0.801980197429657
              },
              "keypoints": {
                "chin": {
                  "x": 0.4895579020182292,
                  "y": 0.8763695271809896
                },
                "nose": {
                  "x": 0.4884576416015625,
                  "y": 0.5780504862467448
                },
                "mouth": {
                  "x": 0.4904075113932292,
                  "y": 0.7090469868977864
                },
                "nose_left": {
                  "x": 0.3953646511501736,
```

```
      "y": 0.5805702209472656
    },
    "left_pupil": {
      "x": 0.30088453504774304,
      "y": 0.4109149678548177
    },
    "mouth_left": {
      "x": 0.33830030653211807,
      "y": 0.6955980936686198
    },
    "nose_right": {
      "x": 0.5848392740885416,
      "y": 0.5861359659830729
    },
    "mouth_right": {
      "x": 0.6333636474609375,
      "y": 0.6958428955078125
    },
    "right_pupil": {
      "x": 0.6891607666015624,
      "y": 0.4170384724934896
    },
    "left_eye_left": {
      "x": 0.21608330620659721,
      "y": 0.4157813008626302
    },
    "left_eye_right": {
      "x": 0.3838650173611111,
      "y": 0.4230572001139323
    },
    "right_eye_left": {
      "x": 0.6079323323567708,
      "y": 0.4287389628092448
    },
    "left_ear_bottom": {
      "x": 0.1043272230360243,
      "y": 0.5988115437825521
    },
    "right_eye_right": {
      "x": 0.7724110243055555,
      "y": 0.4233682759602865
    },
    "left_eye_brow_up": {
      "x": 0.25802788628472223,
      "y": 0.3236322784423828
    },
    "right_ear_bottom": {
      "x": 0.8565327962239583,
      "y": 0.607865956624349
    },
    "right_eye_brow_up": {
      "x": 0.7370150417751736,
      "y": 0.3286321258544922
    },
    "left_eye_brow_left": {
      "x": 0.1554061550564236,
      "y": 0.3466766866048177
    },
    "left_eye_brow_right": {
      "x": 0.3854702419704861,
      "y": 0.3430420430501302
```

```
                },
                "right_eye_brow_left": {
                  "x": 0.6136109754774306,
                  "y": 0.345932362874349
                },
                "right_eye_brow_right": {
                  "x": 0.8359405517578125,
                  "y": 0.35814239501953127
                }
              },
              "templates": {
                "$template10v100": {
                  "id": "708a5da8-104e-4b83-9f26-ec0329e08b89"
                }
              },
              "$cropImage": {
                "id": "3cf0792f-1dde-4e8b-841a-c18330080d21"
              },
              "quality": -540.9627075195312
            }
          ]
        }
      }
    ]
  }
}
```


2.1.4 Верификация лиц

Запрос verify() позволяет сравнить два семпла и определить принадлежность двух изображений лиц одному и тому же человеку.

```
verify(
sourceImage: CustomBinaryType = null
sourceSampleData: JSON = null
sourceSampleId: ID = null
targetSampleId: ID!): MatchResult!
```

sourceImage: CustomBinaryType : Изображение в формате base64

sourceSampleData: JSON : Результат детекции лица, не сохраняемый в базе.

sourceSampleId: ID : ID сэмпла для сравнения с ID целевого сэмпла.

targetSampleId: ID : ID сэмпла для сравнения с исходным изображением, данными исходного сэмпла или ID исходного сэмпла

MatchResult : Результат верификации со следующими параметрами:

- **distance :** Параметр показывает дистанцию между сравниваемыми векторами шаблонов. Чем короче дистанция, тем выше степень верификации.
- **faR :** False acceptance rate (FAR) Коэффициент ложной идентификации показывает уровень сопротивления системы ошибкам ложной идентификации. Такая ошибка возникает, когда биометрическая система определяет новое лицо как ранее

распознанное. Коэффициент измеряется количеством ложных распознаваний, деленным на общее количество попыток распознавания.

- **frR :** False rejection rate (FRR). В случае если система не способна распознать ранее обнаруженное лицо, происходит ложное отклонение. Коэффициент ложного отклонения показывает процент попыток распознавания с ложным отклонением.
- **score :** Параметр показывает уровень верификации от 0 (0%) до 1 (100%)

Ошибки входных данных:

Не передан объект сравнения или передана неоднозначно интерпретируемая комбинация:

```
{
  "message": "One of the parameters sourceSampleData or sourceSampleId or
sourceImage is required",
  "code": "0x963fb254"
}
```

Отсутствует сэмпл по переданному ID:

```
{
  "message": "Sample matching query does not exist."
}
```

Некорректные данные переданного исходного семпла:

```
{
  "message": "'objects@common_capturer_uld_fda'"
}
```

Пример запроса:

```
{
  verify(
    sourceSampleId: "fa76e8a4-3c90-4007-a72f-94d5fc655c36"
    targetSampleId: "a2d852e8-aa00-4403-bc5d-f8b94cc183ca"
  ) {
    distance
    faR
    frR
    score
  }
}
```

Пример ответа:

```
{
  "data": {
    "verify": {
      "distance": 0,
      "faR": 0,
      "frR": 1,
      "score": 1
```

```
    }
  }
}
```

2.1.5 Идентификация лиц

Этот запрос позволяет выполнить поиск человека по базе. Функция search() используется для сравнения одного семпла со всеми остальными семплами в базе.

```
search(
confidenceThreshold: Float = 0
maxNumOfCandidatesReturned: Int = 5
scope: ID = null
sourceImage: CustomBinaryType = null
sourceSampleData: JSON = null
sourceSampleIds: [ID!] = null): [SearchType!]!
```

confidenceThreshold: Float = 0 : Чтобы исключить совпадения с низкой достоверностью из возвращаемого результата, используйте параметр confidenceThreshold (мин. значение: 0, макс. значение: 1; значение по умолчанию: 0)

maxNumOfCandidatesReturned: Int = 5 : Чтобы установить максимальное числом возвращенных кандидатов, укажите значение для параметра maxNumOfCandidatesReturned (мин. значение: 1, макс. значение: 100). По умолчанию возвращаются 5 ближайших кандидатов.

scope: ID : По умолчанию поиск человека выполняется по всей базе лиц. Для поиска совпадений по конкретному списку укажите ID списка в поле scope .

sourceImage: CustomBinaryType : Изображение в формате base64

sourceSampleData: JSON : Результат детекции лица, не сохраняемый в базе данных.

sourceSampleIds: [ID!] : ID сэмплов

SearchType! : API возвращает список кандидатов по каждому запрошенному сэмплу в порядке убывания достоверности. Результат поиска включает следующие параметры:

- **template**
- **searchResult**
- **PersonSearchResult**
    - **sample:** SampleOutput! (id: ID! , creationDate: DateTime , lastModified: DateTime , data: JSON!)
    - **profile:** ProfileOutputData! (id: ID! , info: JSON! , lastModified: DateTime! , creationDate: DateTime! , personId: ID! , mainSample: SampleOutput)

- **matchResult:** MatchResult! (distance: Float! , faR: Float! , frR: Float! , score: Float!)

Примечание: исходные данные сравниваются с профилями, созданными на сервере, а не с сэмплами. Таким образом, перед запуском поиска убедитесь, что у вас создан хотя бы 1 профиль.

Ошибки входных данных:

Не передан объект сравнения или передана неоднозначно интерпретируемая комбинация:

```
{
  "message": "One of the parameters sourceSampleData or sourceSampleId or sourceImage
is required",
  "code": "0x963fb254"
}
```

Значение параметра confidenceThreshold передается за пределами допустимого диапазона:

```
{
  "message": "Confidence threshold must be between 0 and 1",
  "code": "0xf47f116a"
}
```

Значение параметра maxNumOfCandidatesReturned передается за пределами допустимого диапазона:

```
{
  "message": "Max num of candidates must be between 1 and 100",
  "code": "0xf8be6762"
}
```

Некорректные данные переданного исходного семпла:

```
{
  "message": "'objects@common_capturer_uld_fda'"
}
```

Пример запроса:

```
{
  search(sourceSampleIds: "fa76e8a4-3c90-4007-a72f-94d5fc655c36") {
    template
    searchResult {
      matchResult {
        distance
        faR
        frR
        score
      }
```

```
      sample {
        id
      }
      profile {
        id
        info
      }
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "search": [
      {
        "template":
"T5JnWuAN4j5MHWRP7tDj7/AOcg9S8AWw4AAQZwJAHeMRw1MUGfkxFTTVUALbWmAHBHJTBKM9LfMgwCUvXlQg
AizEAzwP0BvkQDHQfykA4ZDvHrkDFhCvD3Eh73TSD+UQTMDSQBPj4DAtYBEi/wDQMH9A8PMObwQQAFHDAgMOA
QHjMB+vEHc2Q/ACH/EHAw+9MPUDA+ImodYPbeKv7Q9/Xw8wD39CPQpQd/wrfwPcZRDSIgkj0PEA8L0NZNoAId
ACER8Q/hAQPQAa8UDdovBN6eXBAHUwQOC7DjECPjBQ5FJBE+MNPeAB1OUPEW787jBCAsHQkBCSRXD0MHEP4E0
O49IgQ7MBGyBEDyMldtHx4RT0MH9MMReb0g==",
        "searchResult": [
          {
            "matchResult": {
              "distance": 0,
              "faR": 0,
              "frR": 1,
              "score": 1
            },
            "sample": {
              "id": "805be807-dd89-4265-9ee7-bbdc19473136"
            },
            "profile": {
              "id": "fd606132-9757-419b-97d6-d8cdd67ed476",
              "info": {
                "age": 25,
                "gender": "MALE",
                "main_sample_id": "805be807-dd89-4265-9ee7-bbdc19473136"
              }
            }
          },
          {
            "matchResult": {
              "distance": 9356,
              "faR": 0.31920063495635986,
              "frR": 0,
              "score": 0.000001847743988037109 4
            },
            "sample": {
              "id": "c18b3a5a-ccfb-4785-9248-b7ce90052754"
            },
            "profile": {
              "id": "218db30f-b6ff-4a46-a83d-a0701005a12b",
              "info": {
                "age": 23,
                "gender": "FEMALE",
                "main_sample_id": "c18b3a5a-ccfb-4785-9248-b7ce90052754"
              }
```

```
                }
            }
        ]
    }
  ]
 }
}
```

Обратите внимание, что если в базе отсутствуют профили, созданные до начала поиска, поле профиля в возвращаемом результате останется пустым.

2.2 Профили

2.2.1 Получить список профилей

Запрос profiles() позволяет получить список созданных профилей.

```
profiles(
filter: JSONString = null
ids: [ID] = null
limit: Int = null
offset: Int = null
order: [String] = null): ProfilesCollection!
```

filter: JSONString : Фильтрация профилей по следующим параметрам:

- **creation_date:** Точная дата создания профиля в формате ISO 8601
- **id:** ID профиля из базы данных
- **info:** Объект со следующими параметрами:
  - **age:** Возраст профиля. Например, запрос с параметром **info_age:28** вернет список профилей с возрастом 28 лет.
  - **gender:** Пол профиля. Например, запрос с параметром **info_gender: “MALE”** вернет список профилей мужского пола.
  - **main_sample_id:** ID лучшего сэмпла профиля
  - **avatar_id:** ID аватара
- **last_modified:** Точная дата последнего изменения профиля в формате ISO 8601
- **link_to_label:**
- **person:**
- **person_id:** ID профиля из базы
- **profile_groups:** Список ID групп
- **samples:** ID сэмпла
- **workspace:** ID воркспейса
- **workspace_id:** ID воркспейса

ids: [ID] : Получение списка профилей по ID;
limit: Int : Получение первых n профилей из списка;
offset: Int : Исключение первые n профилей из списка;
order: [String] : Сортировка профилей по следующим параметрам: **creation_date, id, info, last_modified, link_to_label, person, person_id, profile_groups, samples, workspace, workspace_id.**
ProfilesCollection! : Типы возвращаемых данных:

- **totalCount :** Число возвращенных профилей
- **collectionItems:** [ProfileOutput!]: Список профилей с указанием следующих параметров по каждому профилю:
  - **id :** ID профиля
  - **lastModified :** Дата последнего изменения профиля в формате ISO 8601
  - **creationDate :** Дата создания профиля в формате ISO 8601
  - **info :** Информация профиля (возраст, пол, ID аватара, ID лучшего сэмпла)
  - **samples :** Список сэмплов профиля
  - **profileGroups :** Список групп профиля
  - **mainSample :** Лучший сэмпл профиля
  - **avatar :** Аватар профиля
  - **activities :** Все активности профиля
  - **activitiesCount :** Число активностей профиля
  - **firstActivityDate :** Дата первой активности профиля, зафиксированной системой в формате ISO 8601
  - **lastActivityDate :** Дата последней активности профиля, зафиксированной системой в формате ISO 8601

Пример запроса:

```
{
  profiles(ids: ["1cf13933-00be-4a6d-8dc3-3ff17f94aef8"]) {
    totalCount
    collectionItems {
      id
      avatar
      creationDate
      firstActivityDate
      info
      lastActivityDate
      lastModified
      mainSample {
        id
      }
      profileGroups {
        id
      }
      samples {
        id
      }
      activitiesCount
      activities {
        id
      }
    }
  }
}
```

Пример ответа:


```
{
  "data": {
    "profiles": {
      "totalCount": 1,
      "collectionItems": [
        {
          "id": "1cf13933-00be-4a6d-8dc3-3ff17f94aef8",
          "avatar": "dc3e1949-b4e8-4feb-a51c-67740e323e8b",
          "creationDate": "2022-07-07T08:59:56.227343+00:00",
          "firstActivityDate": "2022-07-07T12:36:51.644000+05:00",
          "info": {
            "age": 36,
            "gender": "MALE",
            "avatar_id": "dc3e1949-b4e8-4feb-a51c-67740e323e8b",
            "main_sample_id": "3071e862-7116-4ed3-94f1-309b0b5b912f"
          },
          "lastActivityDate": "2022-07-07T12:37:07.425000+05:00",
          "lastModified": "2022-07-07T09:00:15.028365+00:00",
          "mainSample": {
            "id": "3071e862-7116-4ed3-94f1-309b0b5b912f"
          },
          "profileGroups": [
            {
              "id": "97c1a9fa-bfde-460a-9bda-f610060423ca"
            }
          ],
          "samples": [
            {
              "id": "3071e862-7116-4ed3-94f1-309b0b5b912f"
            }
          ],
          "activitiesCount": 5,
          "activities": [
            {
              "id": "78f7c795-0f28-42ce-a9d2-818095dd5f84"
            },
            {
              "id": "a4ddc400-8fb8-41ef-aacb-9fc195868368"
            },
            {
              "id": "ea4ba238-4c64-4e07-bed6-d8f29f426e7e"
            },
            {
              "id": "29c8b4f5-11d0-4523-a17f-748ba25d4b97"
            },
            {
              "id": "4a1cad4a-ed81-409e-8a16-38abe9145a77"
            }
          ]
        }
      ]
    }
  }
}
```

2.2.2 Создать профиль

Мутация createProfile() используется для создания нового профиля. Созданный профиль автоматически сохраняется на OMNI-сервере.

```
profiles(
filter: JSON = null
ids: [ID] = null
limit: Int = null
offset: Int = null
order: [String] = null): ProfilesCollection!
```

image: CustomBinaryType : Для добавления аватара в новый профиль передайте изображение в формате base64 размером до 8Мб.

profileData: ProfileInput : Для сохранения дополнительной информации в новом профиле передайте следующие параметры:

- **profileGroupIds: [ID] :** Список ID групп, к которым привязан новый профиль
- **info: JSON :** Дополнительная информация о профиле (age: Int , gender: "MALE" | "FEMALE")

ProfileCreateOutput! : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean :** Флаг об успешном завершении мутации
- **profile: ProfileOutput! :** Объект нового профиля
- **isCreated: Boolean! :** Параметр определяет, были ли создан новый профиль, или фотография прикрепилась к существующему профилю.

Ошибки входных данных:

Низкое качество передаваемого изображения:

```
{
"message": "Low quality photo",
"code": "0x86bd49dh"
}
```

Группы профилей не найдены по переданным ID:

```
{
"message": "One or several profiles_groups does not exist",
"code": "0x573bkd35"
}
```

Пример запроса:

```
mutation {
  createProfile {
    isCreated
    ok
    profile {
      id
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "createProfile": {
      "isCreated": true,
      "ok": true,
      "profile": {
        "id": "d5ca09fb-199a-471f-99ca-b9b4954fb45f"
      }
    }
  }
}
```

2.2.3 Обновить профиль

Мутация updateProfile() используется для обновления информации профиля.

```
updateProfile(profileData: ProfileInput!
profileId: ID): ProfileUpdateOutput!
```

profileData: ProfileInput! : Информация профиля для обновления

(profileGroupIds: [ID], info: JSON)

profileId: ID : ID профиля

ProfileUpdateOutput! : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean :** Флаг об успешном завершении мутации
- **profile: ProfileOutput! :** Обновленный объект профиля

Ошибки входных данных:

Профиль не найден по переданному ID:

```
{
  "message": "Profile matching query does not exist."
}
```

Группы профилей не найдены по переданным ID:

```
{
  "message": "One or several profiles_groups does not exist",
  "code": "0x573bkd35"
}
```

Пример запроса:

```
mutation {
  updateProfile(
    profileData: {info: {age: 20}}
    profileId: "d5ca09fb-199a-471f-99ca-b9b4954fb45f"
  ) {
    ok
    profile {
      id
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "updateProfile": {
      "ok": true,
      "profile": {
        "id": "d5ca09fb-199a-471f-99ca-b9b4954fb45f"
      }
    }
  }
}
```

2.2.4 Удалить профиль

Мутация deleteProfiles() используется для удаления профилей.

```
deleteProfiles(profileIds: [ID!]!): MutationResult!
```

profileIds: [ID!] : Список ID профилей, которые необходимо удалить

MutationResult! : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean :** Флаг об успешном завершении мутации

Ошибки входных данных:

Пустой список ID профилей, переданных для удаления:

```
{
  "message": "Empty profiles ids list",
  "code": "0xd2ae0ef8"
}
```

Пример запроса:

```
mutation {
  deleteProfiles(profileIds: "d5ca09fb-199a-471f-99ca-b9b4954fb45f") {
    ok
  }
}
```

Пример ответа:

```
{
  "data": {
    "deleteProfiles": {
      "ok": true
    }
  }
}
```

2.3 Группы

2.3.1 Получить список групп

Запрос profileGroups() позволяет получить список всех групп, хранящихся в базе данных.

```
profileGroups(
filter: JSON = null
ids: [ID] = null
limit: Int = null
offset: Int = null
order: [String] = null): ProfileGroupsCollection!
```

filter: JSON : Вы можете отфильтровать список групп по одному или нескольким параметрам:

- **area_type :**
- **attention_areas :**
- **camera_location :**
- **cameras :**
- **creation_date :** Точная дата создания группы в формате ISO 8601
- **id :** ID группы из базы данных
- **info :** В формате JSON, полностью аналогичен параметру info объекта группы
- **last_modified :** Точная дата последнего изменения группы в формате ISO 8601
- **link_to_profile :**
- **profiles :** Список ID профилей
- **title :** Название группы
- **type :**
- **workspace :** ID воркспейса
- **workspace_id :** ID воркспейса

ids: [ID] : Для получения списка конкретных групп укажите их ID в списке.

limit: Int : Параметр позволяет получить первые n профилей из списка.

offset: Int : Параметр позволяет убрать первые n профилей из списка.

order: [String] : Вы можете отсортировать список по следующим параметрам: **area_type, attention_areas, camera_location, cameras, creation_date, id, info, is_active, last_modified, link_to_profile, profiles, title, type, workspace, workspace_id.**

ProfileGroupsCollection! : Результат мутации - список групп со следующими параметрами:

- **totalCount :** Число возвращенных групп

- **collectionItems:** [ProfileGroupOutput!]!
  - **id :** ID группы
  - **title :** Название группы
  - **info :** Дополнительная информация о группе в JSON-формате
  - **lastModified :** Дата последнего изменения группы в формате ISO 8601
  - **creationDate :** Дата создания группы в формате ISO 8601
  - **profileIds :** Список ID профилей, добавленных в группу

Пример запроса:

```
{
  profileGroups(ids: ["97c1a9fa-bfde-460a-9bda-f610060423ca"]) {
    totalCount
    collectionItems {
      id
      creationDate
      info
      lastModified
      profileIds
      title
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "profileGroups": {
      "totalCount": 1,
      "collectionItems": [
        {
          "id": "97c1a9fa-bfde-460a-9bda-f610060423ca",
          "creationDate": "2022-07-07T07:21:06.428216+00:00",
          "info": {
            "color": "red.600"
          },
          "lastModified": "2022-07-07T07:21:06.428205+00:00",
          "profileIds": [
            "e4975119-cac7-4ced-ae3f-779bb1093c89",
            "1cf13933-00be-4a6d-8dc3-3ff17f94aef8",
            "b4a16647-1336-4ed8-a440-e4e8896e1de3"
          ],
          "title": "My persons"
        }
      ]
    }
  }
}
```

2.3.2 Создать группу

Мутация createProfileGroup() позволяет создать группу. Созданная группа автоматически сохраняется на OMNI-сервере.

```
createProfileGroup(profileGroupData: ProfileGroupInput!): ProfileGroupModifyOutput!
```

profileGroupData: ProfileGroupInput! : JSON с данными для создания группы:

- **title: String! :** Название новой группы
- **info: JSON = null :** Дополнительная информация о группе

ProfileGroupInput! : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean :** Флаг об успешном завершении мутации
- **profileGroup: ProfileGroupOutput :** Объект новой группы

Пример запроса:

```
mutation {
  createProfileGroup(profileGroupData: {title: "My new group"}) {
    ok
    profileGroup {
      creationDate
      id
      info
      lastModified
      profileIds
      title
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "createProfileGroup": {
      "ok": true,
      "profileGroup": {
        "creationDate": "2022-07-08T07:44:10.766783+00:00",
        "id": "d77c80eb-fffb-4812-a63c-cd8007270ef3",
        "info": {},
        "lastModified": "2022-07-08T07:44:10.766767+00:00",
        "profileIds": [],
        "title": "My new group"
      }
    }
  }
}
```

2.3.3 Удалить группу

Мутация deleteProfileGroup() позволяет удалить список групп.

```
deleteProfileGroup(groupIds: [ID!]!): MutationResult!
```

groupIds: [ID!]! : Список ID групп, которые необходимо удалить

MutationResult! : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean :** Флаг об успешном завершении мутации

Пример запроса:

```
mutation {
  deleteProfileGroup(groupIds: ["d77c80eb-fffb-4812-a63c-cd8007270ef3"]) {
    ok
  }
}
```

Пример ответа:

```
{
  "data": {
    "deleteProfileGroup": {
      "ok": true
    }
  }
}
```

2.3.4 Обновить группу

Мутация updateProfileGroupInfo() используется для обновления информации группы.

```
updateProfileGroupInfo(
profileGroupData: ProfileGroupModifyInput!
profileGroupId: ID!): ProfileGroupModifyOutput!
```

profileGroupData: ProfileGroupModifyInput! : JSON с информацией для обновления:

- **title :** Название группы
- **info :** Дополнительная информация о группе

profileGroupId: ID! : ID группы

ProfileGroupModifyOutput! : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean :** Флаг об успешном завершении мутации
- **profileGroup: ProfileGroupOutput :** Обновленный объект группы

Ошибки входных данных:

Группа не найдена по переданному ID:

```
{
  "message": "Label matching query does not exist."
}
```

Пример запроса:

```
mutation {
  updateProfileGroupInfo(
    profileGroupData: {title: "New group's name"}
    profileGroupId: "800f0b65-7dbe-42b2-8f66-89af95a734e7"
  ) {
    ok
    profileGroup {
      id
      title
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "updateProfileGroupInfo": {
      "ok": true,
      "profileGroup": {
        "id": "800f0b65-7dbe-42b2-8f66-89af95a734e7",
        "title": "New group's name"
      }
    }
  }
}
```

2.3.5 Добавить профили в группы

Мутация addProfilesToGroups() позволяет добавить профили в одну или несколько групп.

```
addProfilesToGroups(
groupIds: [ID!]!
profilesIds: [ID!]!): ProfilesUpdateOutput!
```

groupIds: [ID!]! : Список ID групп

profilesIds: [ID!]! : Список ID профилей

ProfilesUpdateOutput! : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean :** Флаг об успешном завершении мутации
- **profiles: [ProfileOutput!]! :** Список обновленных профилей

Ошибки входных данных:

Профиль не найден по переданному ID:

```
{
  "message": "Profile matching query does not exist."
}
```

Группы не найдены по переданным ID:

```
{
  "message": "One or several profiles_groups does not exist",
  "code": "0x573bkd35"
}
```

Пример запроса:

```
mutation {
  addProfilesToGroups(
    groupIds: ["5b53aa69-d515-4fd3-81bf-c3d8696c3ab8",
"800f0b65-7dbe-42b2-8f66-89af95a734e7"]
    profilesIds: ["1cf13933-00be-4a6d-8dc3-3ff17f94aef8",
"292a2f47-8bfd-4782-8fdf-c8b8189e300e", "3f4c1579-4e5e-4ef4-b9e1-a19808c4d931"]
  ) {
    ok
    profiles {
      id
      profileGroups {
        id
      }
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "addProfilesToGroups": {
      "ok": true,
      "profiles": [
        {
          "id": "1cf13933-00be-4a6d-8dc3-3ff17f94aef8",
          "profileGroups": [
            {
              "id": "97c1a9fa-bfde-460a-9bda-f610060423ca"
            },
            {
              "id": "5b53aa69-d515-4fd3-81bf-c3d8696c3ab8"
            },
            {
              "id": "800f0b65-7dbe-42b2-8f66-89af95a734e7"
            }
          ]
        },
        {
          "id": "292a2f47-8bfd-4782-8fdf-c8b8189e300e",
          "profileGroups": [
            {
```

```
              "id": "5b53aa69-d515-4fd3-81bf-c3d8696c3ab8"
            },
            {
              "id": "800f0b65-7dbe-42b2-8f66-89af95a734e7"
            }
          ]
        },
        {
          "id": "3f4c1579-4e5e-4ef4-b9e1-a19808c4d931",
          "profileGroups": [
            {
              "id": "edbb2c49-bacc-4b43-aac8-e06fd3da35eb"
            },
            {
              "id": "5b53aa69-d515-4fd3-81bf-c3d8696c3ab8"
            },
            {
              "id": "800f0b65-7dbe-42b2-8f66-89af95a734e7"
            }
          ]
        }
      ]
    }
  }
}
```


2.3.6 Удалить профили из групп

Мутация removeProfilesFromGroups() используется для удаления выбранных профилей из одной или нескольких групп.

```
removeProfilesFromGroups(
groupIds: [ID!]!
profilesIds: [ID!]!): ProfilesUpdateOutput!
```

groupIds: [ID!]! : Список ID групп

profilesIds: [ID!]! : Список ID профилей

ProfilesUpdateOutput! : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean :** Флаг об успешном завершении мутации
- **profiles: [ProfileOutput!]! :** Список обновленных профилей

Ошибки входных данных:

Профиль не найден по переданному ID:


```
{
  "message": "Profile matching query does not exist."
}
```

Группы не найдены по переданным ID:

```
{
  "message": "One or several profiles_groups does not exist",
  "code": "0x573bkd35"
}
```

Пример запроса:

```
mutation {
  removeProfilesFromGroups(
    groupIds: ["5b53aa69-d515-4fd3-81bf-c3d8696c3ab8",
"800f0b65-7dbe-42b2-8f66-89af95a734e7"]
    profilesIds: ["1cf13933-00be-4a6d-8dc3-3ff17f94aef8",
"292a2f47-8bfd-4782-8fdf-c8b8189e300e", "3f4c1579-4e5e-4ef4-b9e1-a19808c4d931"]
  ) {
    ok
    profiles {
      id
      profileGroups {
        id
      }
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "removeProfilesFromGroups": {
      "ok": true,
      "profiles": [
        {
          "id": "1cf13933-00be-4a6d-8dc3-3ff17f94aef8",
          "profileGroups": [
            {
              "id": "97c1a9fa-bfde-460a-9bda-f610060423ca"
            }
          ]
        },
        {
          "id": "292a2f47-8bfd-4782-8fdf-c8b8189e300e",
          "profileGroups": []
        },
        {
          "id": "3f4c1579-4e5e-4ef4-b9e1-a19808c4d931",
          "profileGroups": [
            {
              "id": "edbb2c49-bacc-4b43-aac8-e06fd3da35eb"
            }
          ]
        }
      ]
    }
  }
}
```

2.4 Эндпоинты

2.4.1 Получить эндпоинты

Запрос endpoints() позволяет получить список эндпоинтов, на которые направляются оповещения.

```
endpoints(
filter: JSON = null
ids: [ID] = null
limit: Int = null
offset: Int = null
order: [String] = null
withArchived: WithArchived = null): EndpointCollection!
```

filter: JSON : Вы можете отфильтровать список эндпоинтов по одному или нескольким параметрам:

- **creation_date :** Точная дата создания эндпоинта в формате ISO 8601
- **id :** ID эндпоинта из базы данных
- **is_active :**
- **last_modified :** Точная дата последнего изменения эндпоинта в формате ISO 8601
- **meta :** JSON-файл с данными по нахождению эндпоинта
- **triggers :** ID триггера, который запускает оповещения
- **type :** Параметр включает следующие типы эндпоинтов (WI = веб-интерфейс , EM = Email , WH = веб-хук)
- **workspace :** ID воркспейса
- **workspace_id :** ID воркспейса

ids: [ID] : Для получения списка конкретных эндпоинтов укажите их ID в списке.

limit: Int : Параметр позволяет получить первые n профилей из списка.

offset: Int : Параметр позволяет удалить первые n профилей из списка.

order: [String] : Вы можете отсортировать список по следующим параметрам: **creation_date, id, is_active, last_modified, meta, triggers, type, workspace, workspace_id.**

withArchived: WithArchived : Для получения списка всех эндпоинтов, включая архивированные, укажите значение all . Для получения только архивированных эндпоинтов укажите значение archived .

EndpointCollection! : Результат запроса - список эндпоинтов со следующими параметрами:

- **totalCount :** Число возвращенных эндпойнтов

- **profiles: [EndpointOutput!]! :**

  - **id: ID! :** ID эндпойнта

  - **type: EndpointType! :** Тип эндпойнта может иметь следующие значения: Email , Webhook или WebInterface

  - **meta: JSONString! :** Мета-информация для нахождения эндпойнта

  - **lastModified: DateTime! :** Дата последнего изменения эндпойнта в формате ISO 8601

  - **creationDate: DateTime! :** Дата создания эндпойнта в формате ISO 8601

  - **defaultAlias: DefaultAlias :**

    - OWNER_EMAIL

    - WEB_INTERFACE

  - **archived: Boolean! :** Атрибут указывает на то, что эндпоинт заархивирован.

Пример запроса:

```
{
  endpoints(ids: ["6fd67f81-a195-442b-a991-1a0eca6bbb69"]) {
    totalCount
    collectionItems {
      archived
      creationDate
      id
      defaultAlias
      lastModified
      meta
      type
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "endpoints": {
      "totalCount": 1,
      "collectionItems": [
        {
          "archived": false,
          "creationDate": "2022-07-07T07:21:06.421413+00:00",
          "id": "6fd67f81-a195-442b-a991-1a0eca6bbb69",
          "defaultAlias": "OWNER_EMAIL",
          "lastModified": "2022-07-07T07:21:06.421425+00:00",
          "meta": "{\"target_email\": \"aa@aa.ru\", \"default_alias\":
\"owner_email\"}",
          "type": "Email"
        }
      ]
    }
  }
}
```

2.4.2 Создать эндпоинт типа email

Мутация createEmailEndpoint() позволяет создать эндпоинт типа Email .

```
createEmailEndpoint(endpointData: EmailEndpointInput!): EndpointManageOutput!
```

endpointData: EmailEndpointInput! : JSON с информацией по созданию эндпоинта:

- **targetEmail: String! :** Email, куда будут отправляться оповещения.

EndpointManageOutput! : Результат мутации - JSON со следующими параметрами:

- **ok: Boolean! :** Атрибут успешного завершения мутации
- **endpoint: EndpointOutput! :** Объект нового эндпоинта

Пример запроса:

```
mutation {
  createEmailEndpoint(endpointData: {targetEmail: "test@test.com"}) {
    ok
    endpoint {
      id
      archived
      creationDate
      defaultAlias
      lastModified
      meta
      type
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "createEmailEndpoint": {
      "ok": true,
      "endpoint": {
        "id": "2eff291d-8c8c-4325-8953-5a70b31bc63e",
        "archived": false,
        "creationDate": "2022-07-10T12:27:22.568988+00:00",
        "defaultAlias": null,
        "lastModified": "2022-07-10T12:27:22.569003+00:00",
        "meta": "{\"target_email\": \"test@test.com\"}",
        "type": "EMAIL"
      }
    }
  }
}
```

2.4.3 Создать эндпоинт типа вебхук

Мутация createWebhookEndpoint() позволяет создать эндпоинт типа вебхук.

```
createWebhookEndpoint(endpointData: WebhookEndpointInput!): EndpointManageOutput!
```

endpointData: WebhookEndpointInput! : JSON с информацией для создания эндпоинта:

- **url: String! :** URL, куда будут отправляться запросы
- **requestMethod: String! :** Метод запроса

EndpointManageOutput! : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean! :** Атрибут успешного завершения мутации
- **endpoint: EndpointOutput! :** Объект нового эндпоинта

Пример запроса:

```
mutation {
  createWebhookEndpoint(
    endpointData: {url: "https://endpoint_test.requestcatcher.com/test /",
requestMethod: "POST"}
  ) {
    ok
    endpoint {
      archived
      creationDate
      defaultAlias
      id
      lastModified
      meta
      type
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "createWebhookEndpoint": {
      "ok": true,
      "endpoint": {
        "archived": false,
        "creationDate": "2022-07-10T12:42:34.957171+00:00",
        "defaultAlias": null,
        "id": "afd7c121-3140-4a8c-b0ee-3717581eb76d",
        "lastModified": "2022-07-10T12:42:34.957188+00:00",
        "meta": "{\"url\": \"https://endpoint_test.requestcatcher.com/test /\",
\"method\": \"POST\"}",
        "type": "WEBHOOK"
      }
    }
  }
```

```
}
```

2.4.4 Обновить эндпоинт

Мутация updateEndpoint() позволяет обновить данные эндпоинта.

```
updateEndpoint(endpointId: ID!
endpointInfo: JSON): EndpointManageOutput!
```

endpointId: ID! : ID эндпоинта

endpointInfo: JSON : JSON-объект с параметрами для обновления. Эти параметры аналогичны параметрам из meta: JSON

EndpointManageOutput! : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean! :** Атрибут успешного завершения мутации
- **endpoint: EndpointOutput! :** Объект обновленного эндпоинта

Ошибки входных данных:

Эндпоинт не найден по переданному ID:

```
{
  "message": "Endpoint matching query does not exist."
}
```

Пример запроса:

```
mutation {
  updateEndpoint(
    endpointId: "6fd67f81-a195-442b-a991-1a0eca6bbb69"
    endpointInfo: {target_email: "new-email@test.com"}
  ) {
    ok
    endpoint {
      id
      meta
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "updateEndpoint": {
      "ok": true,
      "endpoint": {
        "id": "6fd67f81-a195-442b-a991-1a0eca6bbb69",
        "meta": "{\"target_email\": \"new-email@test.com\", \"default_alias\":
\"owner_email\"}"
      }
```

```
    }
  }
}
```

2.4.5 Удалить эндпоинт

Мутация deleteEndpoint() позволяет удалить список эндпоинтов.

```
deleteEndpoint(endpointIds: [String!]!): MutationResult!
```

endpointIds: [String!]! : Список ID эндпоинтов

MutationResult! : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean! :** Атрибут успешного завершения мутации

Пример запроса:

```
mutation {
  deleteEndpoint(
    endpointIds: ["2eff291d-8c8c-4325-8953-5a70b31bc63e",
"29bcfb3a-3a26-4544-9def-d5d44bbd3be4"]
  ) {
    ok
  }
}
```

Пример ответа:

```
{
  "data": {
    "deleteEndpoint": {
      "ok": true
    }
  }
}
```

2.5 Триггеры

2.5.1 Получить список триггеров

Запрос triggers позволяет получить список триггеров для создания оповещений. Триггеры связаны с эндпоинтами, на которые отправляются оповещения.

```
triggers(
filter: JSON = null
ids: [ID] = null
limit: Int = null
offset: Int = null
order: [String] = null
targetId: ID = null
withArchived: WithArchived = null): TriggerCollection!
```

filter: JSON : Вы можете отфильтровать список триггеров, указав один или несколько параметров:

- **creation_date :** Точная дата создания триггера в формате ISO 8601
- **endpoints :** ID связанного эндпоинта
- **id :** ID триггера из базы данных
- **is_active :**
- **last_modified :** Точная дата последнего изменения триггера в формате ISO 8601
- **meta :**
- **workspace :** ID воркспейса
- **workspace_id :** ID воркспейса

ids: [ID] : Для получения списка конкретных триггеров укажите их ID в списке.

limit: Int : Параметр позволяет получить первые n триггеров из списка.

offset: Int : Параметр позволяет удалить первые n триггеров из списка.

order: [String] : Вы можете отсортировать список по следующим параметрам: **creation_date, endpoints, id, is_active, last_modified, meta, workspace, workspace_id.**

targetId: ID : ID группы, к которой привязан триггер

withArchived: WithArchived : Для получения списка всех триггеров, включая заархивированные триггеры, укажите значение all . Для получения списка только заархивированных триггеров укажите значение archived .

TriggerCollection! : Результат запроса - список триггеров со следующими параметрами:

totalCount: Int : Число возвращенных триггеров

collectionItems: [TriggerType!]! :

- **id: ID! :** ID триггера

- **creationDate: DateTime! :** Дата создания триггера в формате ISO 8601
- **lastModified: DateTime! :** Дата последнего изменения триггера в формате ISO 8601
- **meta: Meta! :** Мета-информация триггера
- **endpoints: [EndpointOutput!]! :** Список эндпоинтов, связанных с триггером
- **archived: Boolean! :** Атрибут об архивации триггера

Пример запроса:

```
{
  triggers(ids: ["3e57a95e-028c-4d27-9419-a9c4bfd6f3fb"]) {
    totalCount
    collectionItems {
      id
      creationDate
      lastModified
      meta {
        notificationParams
        conditionLanguage {
          condition
          variables {
            name
            target {
              type
              uuid
            }
            type
          }
        }
      }
      endpoints {
        id
      }
      archived
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "triggers": {
      "totalCount": 1,
      "collectionItems": [
        {
          "id": "3e57a95e-028c-4d27-9419-a9c4bfd6f3fb",
          "creationDate": "2022-07-07T08:38:22.982190+00:00",
          "lastModified": "2022-07-08T05:03:48.961423+00:00",
          "meta": {
            "notificationParams": "{\"lifetime\": 5}",
            "conditionLanguage": {
              "condition": null,
              "variables": [
                {
                  "name": "0_v",
                  "target": [
```

```
                    {
                      "type": "Label",
                      "uuid": "edbb2c49-bacc-4b43-aac8-e06fd3da35eb"
                    }
                  ],
                  "type": "presence"
                }
              ]
            }
          },
          "endpoints": [
            {
              "id": "db38ec53-c09d-45f2-bfed-5877c9bd9016"
            }
          ],
          "archived": false
        }
      ]
    }
  }
}
```


2.5.2 Создать триггер для группы

Мутация createProfileGroupTrigger позволяет создать триггер для группы. При каждом обнаружении камерой профиля из этой группы система будет создавать оповещение.

```
createProfileGroupTrigger(
endpointAliases: [DefaultAlias!] = null
endpointIds: [ID!] = null
endpointUrl: String = null
profileGroupId: ID!): TriggerManageOutput!
```

endpointAliases: [DefaultAlias!] : Вы можете передать значения OWNER_EMAIL или WEB_INTERFACE. В этом случае эндпоинт с аналогичным значением для параметра defaultAlias будет привязан к триггеру.

endpointIds: [ID!] : Вы можете передать список ID эндпоинтов, привязанных к триггеру.

endpointUrl: String : Вы можете передать URL в ваш вебхук. Это создаст новый эндпоинт для отправления оповещений на ваш URL адрес.

profileGroupId: ID! : ID группы для создания триггера.

TriggerManageOutput! : Результат мутации - JSON-файл со следующими параметрами:

ok: Boolean! : Атрибут успешного завершения мутации

trigger: TriggerType! : Объект нового триггера

Ошибки входных данных:

Группа не найдена по переданному ID:

```
{
  "message": "Label matching query does not exist."
}
```

Эндпоинт не найден по переданному ID:

```
{
  "message": "Endpoint does not exist."
}
```

Пример запроса:

```
mutation {
  createProfileGroupTrigger(
    profileGroupId: "800f0b65-7dbe-42b2-8f66-89af95a734e7"
    endpointIds: ["afd7c121-3140-4a8c-b0ee-3717581eb76d"]
  ) {
    trigger {
      endpoints {
        id
      }
      id
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "createProfileGroupTrigger": {
      "trigger": {
        "endpoints": [
          {
            "id": "afd7c121-3140-4a8c-b0ee-3717581eb76d"
          }
        ],
        "id": "9675aede-3ee8-495b-ac8b-e284a07db99e"
      }
    }
  }
}
```

2.5.3 Обновить триггер

Мутация updateTrigger позволяет обновить эндпоинты, привязанные к триггеру. Список привязанных эндпоинтов заменяется новым списком.

```
updateTrigger(
endpointAliases: [DefaultAlias!] = null
endpointIds: [ID!] = null
triggerId: ID!): TriggerManageOutput!
```

endpointAliases: [DefaultAlias!] : Вы можете передать значения OWNER_EMAIL или WEB_INTERFACE . В этом случае один из эндпоинтов с аналогичным значением в параметре defaultAlias будет привязан к триггеру.

endpointIds: [ID!] : Вы можете передать список ID эндпоинтов, привязанных к триггеру.

triggerId: ID! : ID триггера

TriggerManageOutput! : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean! :** Атрибут успешного завершения мутации
- **trigger: TriggerType! :** Объект обновленного триггера

Ошибки входных данных:

Триггер не найден по переданному ID:

```
{
  "message": "Trigger matching query does not exist."
}
```

Эндпоинт не найден по переданному ID:

```
{
  "message": "Endpoint does not exist."
}
```

Пример запроса:

```
mutation {
  updateTrigger(
    triggerId: "64e0a7ec-0df4-4734-a460-601fa1b65a1f"
    endpointIds: ["6e5a6d3b-8247-488e-95d7-57a306b294ed"]
    endpointAliases: OWNER_EMAIL
  ) {
    ok
    trigger {
      id
      endpoints {
        id
        defaultAlias
      }
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "updateTrigger": {
      "ok": true,
      "trigger": {
        "id": "64e0a7ec-0df4-4734-a460-601fa1b65a1f",
        "endpoints": [
```

```
          {
            "id": "6e5a6d3b-8247-488e-95d7-57a306b294ed",
            "defaultAlias": null
          },
          {
            "id": "6fd67f81-a195-442b-a991-1a0eca6bbb69",
            "defaultAlias": "OWNER_EMAIL"
          }
        ]
      }
    }
  }
}
```


2.5.4 Привязать эндпоинт к триггеру

Мутация linkEndpoint позволяет привязать эндпоинт к триггеру. Необходимо добавить эндпоинт в список эндпоинтов, привязанных к триггеру.

```
linkEndpoint(
endpointAliases: [DefaultAlias!] = null
endpointIds: ID! = null
triggerId: ID!): MutationResult!
```

endpointAliases: [DefaultAlias!] : Можно передать значения OWNER_EMAIL или WEB_INTERFACE . В этом случае один из эндпоинтов с аналогичным значением для параметра defaultAlias будет привязан к триггеру.

endpointIds: ID! : ID эндпоинта, который будет привязан к триггеру.

triggerId: ID! : ID триггера

MutationResult! : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean! :** Атрибут успешного завершения мутации

Ошибки входных данных:

Не передан ID эндпоинта для привязки к триггеру:


```
{
  "message": "Unknown exception code for 'bad_input_data' type",
  "code": "No id or alias provided"
}
```


Не найден эндпоинт по переданному ID:


```
{
  "message": "Endpoint matching query does not exist."
}
```

Не найден триггер по переданному ID:

```
{
  "message": "Trigger matching query does not exist."
}
```

Пример запроса:

```
mutation {
  linkEndpoint(
    triggerId: "64e0a7ec-0df4-4734-a460-601fa1b65a1f"
    endpointAlias: WEB_INTERFACE
  ) {
    ok
  }
}
```

Пример ответа:

```
{
  "data": {
    "linkEndpoint": {
      "ok": true
    }
  }
}
```

2.5.5 Отвязать эндпоинт от триггера

Мутация unlinkEndpoint позволяет отвязать эндпоинт от триггера. Эндпоинт будет удален из списка эндпоинтов, привязанных к триггеру.

```
unlinkEndpoint(
endpointAliases: [DefaultAlias!] = null
endpointIds: ID! = null
triggerId: ID!): MutationResult!
```

endpointAliases: [DefaultAlias!] : Вы можете передать значения OWNER_EMAIL или WEB_INTERFACE . В этом случае один из эндпоинтов с аналогичным значением параметра defaultAlias будет удален из списка эндпоинтов, привязанных к этому триггеру.
endpointIds: ID! : Вы можете передать ID эндпоинта, который необходимо удалить из списка эндпоинтов, привязанных к этому триггеру.
triggerId: ID! : ID триггера
MutationResult! : Результат мутации - JSON-файл со следующими параметрами::

- **ok: Boolean! :** Атрибут успешного завершения мутации

Ошибки входных данных:

ID эндпоинта, который необходимо отвязать от триггера, не передан:

```
{
  "message": "Unknown exception code for 'bad_input_data' type",
  "code": "No id or alias provided"
}
```

Эндпоинт не найден по переданному ID:

```
{
  "message": "Endpoint matching query does not exist."
}
```

Триггер не найден по переданному ID:

```
{
  "message": "Trigger matching query does not exist."
}
```

Пример запроса:

```
mutation {
  unlinkEndpoint(
    triggerId: "64e0a7ec-0df4-4734-a460-601fa1b65a1f"
    endpointAlias: WEB_INTERFACE
  ) {
    ok
  }
}
```

Пример ответа:

```
{
  "data": {
    "unlinkEndpoint": {
      "ok": true
    }
  }
}
```

2.5.6 Удалить триггер

Мутация deleteTrigger позволяет удалить триггер.

```
deleteTrigger(triggerId: ID!): MutationResult!
```

triggerId: ID! : ID триггера

MutationResult! : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean! :** Атрибут успешного завершения мутации.

Ошибки входных данных:

Триггер не найден по переданному ID:

```
{
  "message": "Trigger matching query does not exist."
}
```

Пример запроса:

```
mutation {
  deleteTrigger(triggerId: "64e0a7ec-0df4-4734-a460-601fa1b65a1f") {
    ok
  }
}
```

Пример ответа:

```
{
  "data": {
    "deleteTrigger": {
      "ok": true
    }
  }
}
```

2.6 Оповещения

2.6.1 Получить список оповещений

Запрос notifications позволяет получить список оповещений.

```
notifications(
filters: NotificationFilter
order: NotificationOrdering
pagination: OffsetPaginationInput): NotificationOutputCountList!
```

filters: NotificationFilter : Можно настроить получение списка конкретных оповещений

id : Фильтрация по ID оповещений:

- **exact: ID :** Получить точный объект оповещения по ID оповещения.
- **iExact: ID :** Получить точный объект оповещения по ID оповещения. Без учета регистра.
- **contains: String :** Получить объекты оповещений, в которых значение является частью ID оповещения.
- **iContains: String :** Получить объекты оповещений, в которых значение является частью ID оповещения. Без учета регистра.
- **inList: [ID!] :** Получить точные объекты оповещений по списку ID оповещений.
- **gt: ID :** Получить объекты оповещений с ID больше(>) переданных ID. Используется сравнение строк.
- **gte: ID :** Получить объекты оповещений с ID больше или равным (>=) переданным ID. Используется сравнение строк.
- **lt: ID :** Получить объекты оповещений с ID меньше (<) переданных ID. Используется сравнение строк.
- **lte: ID :** Получить объекты оповещений с ID меньше или равным(<=) переданным ID. Используется сравнение строк.
- **startsWith: String :** Получить объекты оповещений, чьи ID начинаются с переданного значения.
- **iStartsWith: String :** Получить объекты оповещений, чьи ID начинаются с переданного значения. Без учета регистра.
- **endsWith: String :** Получить объекты оповещений, чьи ID заканчиваются переданным значением.

- **iEndsWith: String :** Получить объекты оповещений, чьи ID заканчиваются переданным значением. Без учета регистра.
- **range: [ID!] :** Получить объекты оповещений, чьи ID находятся в диапазоне переданных ID. Используется сравнение строк.
- **isNull: Boolean :** Получить объекты оповещений, чьи ID равны(true) или не равны (false) нулю null .
- **regex: String :** Получить объекты оповещений, чьи ID соответствуют переданному регулярному выражению.
- **iRegex: String :** Получить объекты оповещений, чьи ID соответствуют переданному регулярному выражению. Без учета регистра.

creationDate: DatetimeFilterLookupCustom : Фильтрация по дате создания оповещения. Параметры аналогичны параметрам для фильтрации по ID.

lastModified: DatetimeFilterLookupCustom : Фильтрация по дате последнего изменения оповещения. Параметры аналогичны параметрам для фильтрации по ID.

isViewed: Boolean : Фильтрация просмотренных (true) и непросмотренных (false) оповещений.

isActive: Boolean : Фильтрация активных (true) и неактивных (false) оповещений.

endpointId: UUID : Фильтрация оповещений по ID эндпойнта, на который отправляются оповещения.

isSent: Boolean : Фильтрация оповещений по оповещениям внутри системы (true) и оповещениям, направленным на внешние эндпойнты (false).

triggerId: ID : Фильтрация оповещений по ID триггера.

profileId: ID : Фильтрация оповещений по ID профиля.

type: NotificationType : Фильтрация оповещений по типу.

profileGroupTitle: String : Фильтрация оповещений по названию группы.

order: NotificationOrdering : Вы можете настроить сортировку списка оповещений:

- **id: Ordering :** Сортировка по ID. Используется сравнение строк. (ASC : Сортировка по возрастанию, DESC : Сортировка по убыванию)
- **creationDate: Ordering :** Сортировка по дате создания оповещения.
- **lastModified: Ordering :** Сортировка по дате последнего изменения оповещения.

pagination: OffsetPaginationInput :

- **limit: Int! :** Этот параметр позволяет получить первые n оповещений из

списка.

- **offset: Int! :** Этот параметр позволяет удалить первые n оповещений из списка.

NotificationOutputCountList! : Результат запроса - список оповещений со следующими параметрами:

- **totalCount: Int! :** Общее число оповещений.
- **collectionItems: [NotificationOutput!]! :**
  - **id: ID! :** ID оповещения.
  - **isActive: Boolean! :** Атрибут о том, что оповещение активно.
  - **isViewed: Boolean! :** Атрибут о том, что оповещение просмотрено.
  - **lastModified: DateTime :** Дата последнего изменения оповещения в формате ISO 8601.
  - **activityId: ID :** ID активности, вызвавшей оповещение.
  - **avatarId: ID :** ID аватара профиля.
  - **cameraId: ID :** ID камеры, обнаружившей активность.
  - **cameraTitle: String :** Название камеры
  - **creationDate: DateTime! :** Дата создания оповещения в формате ISO 8601.
  - **currentCount: Int :** Число людей в поле зрения камеры
  - **description: String :** Описание профиля
  - **endpointStatuses: [EndpointStatusOutput!] :** Список эндпоинтов со статусами:
    - **endpoint: EndpointOutput! :** Объект эндпоинта.
    - **status: String! :** Статус передачи.
  - **limit: Int :** Максимальное число персон.
  - **name: String :** Имя профиля.
  - **profileGroupColor: String :** Цвет группы, привязанной к триггеру.
  - **profileGroupId: ID :** ID группы, привязанной к триггеру.
  - **profileGroupTitle: String :** Название группы, привязанной к триггеру.
  - **profileId: ID :** ID профиля.
- **realtimeBodyPhotoId: String :** ID изображения человека, обнаруженного камерой (Отсутствует в анонимном режиме).

Пример запроса:

```
{
  notifications(filters: {id: {exact: "ed1a55af-6daf-4368-ac68-0defdff5159c"}}) {
    totalCount
    collectionItems {
      profileGroupColor
      activityId
      avatarId
      cameraId
      cameraTitle
      creationDate
      currentCount
      description
      id
      endpointStatuses {
        status
        endpoint {
          id
        }
      }
      isActive
      isViewed
      lastModified
      limit
      name
      profileGroupId
      profileGroupTitle
      profileId
      realtimeBodyPhotoId
      realtimeFacePhotoId
      triggerId
      type
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "notifications": {
      "totalCount": 1,
      "collectionItems": [
        {
          "profileGroupColor": "red.600",
          "activityId": "f3bf0ac7-5849-45d7-85e0-12de925633da",
          "avatarId": "4312809b-99b2-47a7-beb9-b60b87c07594",
          "cameraId": "3c818dc4-352c-47a7-b32b-465fbd4a9665",
          "cameraTitle": "My <ShortProductName/> Agent",
          "creationDate": "2022-07-07T13:10:53.619145+00:00",
          "currentCount": null,
          "description": "gap\n",
          "id": "ed1a55af-6daf-4368-ac68-0defdff5159c",
          "endpointStatuses": [
            {
              "status": "success",
              "endpoint": {
                "id": "db38ec53-c09d-45f2-bfed-5877c9bd9016"
```

```
              }
            }
          ],
          "isActive": false,
          "isViewed": true,
          "lastModified": "2022-07-07T13:11:12.264337+00:00",
          "limit": null,
          "name": null,
          "profileGroupId": "97c1a9fa-bfde-460a-9bda-f610060423ca",
          "profileGroupTitle": "My persons",
          "profileId": "b4a16647-1336-4ed8-a440-e4e8896e1de3",
          "realtimeBodyPhotoId": "339a4020-a9af-49e8-83d4-3fc34ef794e5",
          "realtimeFacePhotoId": null,
          "triggerId": "66b3eb1a-de88-469e-83c9-697785a0a4c2",
          "type": "presence"
        }
      ]
    }
  }
}
```


2.6.2 Просмотреть оповещение

Мутация viewingNotifications отмечает одно или несколько оповещений как просмотренные.

```
viewingNotifications(notificationIds: [String!]!): MutationResult!
```

notificationIds: [String!]! : Список ID оповещений, отмеченных как просмотренные.

MutationResult : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean! :** Атрибут успешного завершения мутации.

Пример запроса:

```
mutation {
  viewingNotifications(notificationIds: ["5a1a1bed-5bc0-4e8e-8154-e56c5ef5ea87"]) {
    ok
  }
}
```

Пример ответа:

```
{
  "data": {
    "viewingNotifications": {
      "ok": true
    }
  }
}
```

2.6.3 Просмотреть все оповещения

Мутация markAllNotificationsAsViewed отмечает все оповещения как просмотренные.

```
markAllNotificationsAsViewed: MutationResult!
```

MutationResult : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean! :** Атрибут успешного завершения мутации.

Пример запроса:

```
mutation {
  markAllNotificationsAsViewed {
    ok
  }
}
```

Пример ответа:

```
{
  "data": {
    "markAllNotificationsAsViewed": {
      "ok": true
    }
  }
}
```

2.7 Активности

2.7.1 Получить список активностей

Запрос activities позволяет получить список активностей, обнаруженных камерой.

```
notifications(
filters: ActivityFilter order: ActivityOrdering
pagination: OffsetPaginationInput): ActivityOutputCountList!
```

filters: ActivityFilter : Для получения только конкретных активностей можно настроить фильтрацию

- **id :** Фильтрация по ID активности:

  - **exact: ID :** Получить точный объект активности по значению ID активности.

  - **iExact: ID :** Получить точный объект активности по значению ID активности. Без учета регистра.

  - **contains: String :** Получить объекты активностей, в которых значение является частью ID активности.

  - **iContains: String :** Получить объекты активностей, в которых значение является частью ID активности. Без учета регистра.

  - **inList: [ID!] :** Получить точные объекты активностей по списку ID активностей.

  - **gt: ID :** Получить объекты активностей с ID больше(>) переданных ID. Используется сравнение строк.

  - **gte: ID :** Получить объекты активностей с ID больше или равным (>=) переданным ID. Используется сравнение строк.

  - **lt: ID :** Получить объекты активностей с ID меньше (<) переданных ID. Используется сравнение строк.

  - **lte: ID :** Получить объекты активностей с ID меньше или равным(<=) переданным ID. Используется сравнение строк.

  - **startsWith: String :** Получить объекты активностей, чьи ID начинаются с переданного значения.

  - **iStartsWith: String :** Получить объекты активностей, чьи ID начинаются с переданного значения. Без учета регистра.

  - **endsWith: String :** Получить объекты активностей, чьи ID заканчиваются переданным значением.

  - **iEndsWith: String :** Получить объекты активностей, чьи ID заканчиваются переданным значением. Без учета регистра.

- **range: [ID!] :** Получить объекты активностей, чьи ID находятся в диапазоне переданных ID. Используется сравнение строк.
- **isNull: Boolean :** Получить объекты активностей, чьи ID равны(true) или не равны (false) нулю null .
- **regex: String :** Получить объекты активностей, чьи ID соответствуют переданному регулярному выражению.
- **iRegex: String :** Получить объекты активностей, чьи ID соответствуют переданному регулярному выражению. Без учета регистра.

- **creationDate: DatetimeFilterLookupCustom :** Фильтрация по дате создания активности. Параметры аналогичны параметрам для фильтрации по ID.
- **lastModified: DatetimeFilterLookupCustom :** Фильтрация по дате последнего изменения активности. Параметры аналогичны параметрам для фильтрации по ID.
- **profileId: ID :** Фильтрация активностей по ID профиля.

order: ActivityOrdering : Вы можете настроить сортировку списка активностей:

- **id: Ordering :** Сортировка по ID. Используется сравнение строк.(ASC : Сортировка по возрастанию, DESC : Сортировка по убыванию)
- **creationDate: Ordering :** Сортировка по дате создания активности.
- **lastModified: Ordering :** Сортировка по дате последнего изменения активности.
- **pagination:** OffsetPaginationInput :
- **limit: Int! :** Параметр позволяет получить первые n активностей из списка.
- **offset: Int! :** Параметр позволяет удалить первые n активностей из списка.
- **ActivityOutputCountList! :** Результат запроса - список активностей со следующими параметрами:
- **totalCount: Int! :** Число возвращенных активностей.
- **collectionItems: [ActivityOutput!]! :**
  - **id: ID! :** ID активности.
  - **data: JSON :** Дополнительная информация об активности.
  - **lastModified: DateTime :** Дата последнего изменения активности в формате ISO 8601.
  - **creationDate: DateTime! :** Дата создания активности в формате ISO 8601.
  - **bestShotId: ID :** ID лучшего изображения лица, полученного с камеры, обнаружившей активность (Отсутствует, если включен анонимный режим или камера не смогла обнаружить изображение лица).
  - **cameraId: ID! :** ID камеры, заметившей активность.

- **locationId: String! :** ID локации, к которой привязана камера.
- **profileId: ID :** ID профиля.
- **status: ActivityType!:** статус активности:
  - Progress (продолжается, человек находится в кадре)
  - Finalized (завершена, человек вышел из кадра)
  - Failed (не выполнена, завершающая активность не передана)
- **timeStart: String! :** Время начала активности
- **timeEnd: String! :** Время завершения активности

Пример запроса:

```
{
  activities(filters: {id: {exact: "78f7c795-0f28-42ce-a9d2-818095dd5f84"}}) {
    totalCount
    collectionItems {
      id
      data
      cameraId
      bestShotId
      locationId
      profileId
      timeStart
      creationDate
      lastModified
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "activities": {
      "totalCount": 1,
      "collectionItems": [
        {
          "id": "78f7c795-0f28-42ce-a9d2-818095dd5f84",
          "data": {
            "processes": [
              {
                "id": "78f7c795-0f28-42ce-a9d2-818095dd5f84",
                "type": "track",
                "object": {
                  "id": "c4398bcc-6fee-495a-be86-a59918a875e5",
                  "class": "human"
                },
                "time_interval": [
                  "2022-07-07T12:36:51.644000+05:00",
                  "2022-07-07T12:36:51.725000+05:00"
                ]
              },
              {
                "id": "ecb65ed1-e3c5-484f-b1e3-7f4bdf869757",
                "type": "track",
```

```
                "object": {
                  "id": "c4398bcc-6fee-495a-be86-a59918a875e5",
                  "age": 36,
                  "class": "face",
                  "gender": "MALE",
                  "quality": -1114.097412109375,
                  "embeddings": {
                    "$binary_image": {
                      "id": "7fd8abfb-258b-4d6a-bdd7-eec1e30fd0b4"
                    },
                    "$template10v100": {
                      "id": "35b60b5f-d6da-4998-ac4c-5725d3e64520"
                    }
                  }
                },
                "parent": "78f7c795-0f28-42ce-a9d2-818095dd5f84",
                "sample_id": "3071e862-7116-4ed3-94f1-309b0b5b912f",
                "$best_shot": {
                  "id": "ab8d76c8-0a8c-4575-8d81-e9e469f8ef64"
                },
                "time_interval": [
                  "2022-07-07T12:36:51.644000+05:00",
                  "2022-07-07T12:36:51.725000+05:00"
                ]
              },
              {
                "id": "3e33004f-6531-40ca-90d5-8345f66002ac",
                "type": "attention",
                "parent": "ecb65ed1-e3c5-484f-b1e3-7f4bdf869757",
                "time_interval": [
                  "2022-07-07T12:36:51.806000+05:00",
                  "2022-07-07T12:36:52.413000+05:00"
                ]
              }
            ]
          },
          "cameraId": "3c818dc4-352c-47a7-b32b-465fbd4a9665",
          "bestShotId": "ab8d76c8-0a8c-4575-8d81-e9e469f8ef64",
          "locationId": "",
          "profileId": "1cf13933-00be-4a6d-8dc3-3ff17f94aef8",
          "timeStart": "2022-07-07T12:36:51.644000+05:00",
          "creationDate": "2022-07-07T07:36:52.848815+00:00",
          "lastModified": "2022-07-07T08:59:56.385040+00:00"
        }
      ]
    }
  }
}
```


2.7.2 Создать профиль из активности

Мутация createProfileByActivity используется для создания профиля из активности.

```
createProfileByActivity(
activityId: ID!
profileData: ProfileInput = null): ProfileCreateOutput!
```

activityId: ID! : ID активности, из которой должен быть создан профиль.

profileData: ProfileInput : Вы можете указать дополнительную информацию для сохранения в профиле:

- **profileGroupIds: [ID] :** Список ID групп, к которым привязан новый профиль.
- **info: JSON :** Дополнительная информация о профиле (age: Int , gender: "MALE" | "FEMALE")

ProfileCreateOutput : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean! :** Атрибут успешного завершения мутации
- **profile: ProfileOutput! :** Объект нового профиля.
- **isCreated: Boolean! :** Определяет, был ли создан новый профиль, или фотография привязана к уже существующему профилю.

Ошибки входных данных:

Ошибка при попытке создать профиль из анонимной активности:

```
{
  "message": "Activity is anonymous",
  "code": "0x358vri3s"
}
```

Активность не найдена по переданному ID:

```
{
  "message": "Activity matching query does not exist."
}
```

Группы не найдены по переданным ID:

```
{
  "message": "One or several profiles_groups does not exist",
  "code": "0x573bkd35"
}
```

Пример запроса:

```
mutation {
  createProfileByActivity(activityId: "78f7c795-0f28-42ce-a9d2-818095dd5f84") {
    ok
    isCreated
    profile {
      id
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "createProfileByActivity": {
```

```
      "ok": true,
      "isCreated": false,
      "profile": {
        "id": "1cf13933-00be-4a6d-8dc3-3ff17f94aef8"
      }
    }
  }
}
```

2.8 Агенты

2.8.1 Получить список агентов

Запрос agents позволяет получить список агентов.

```
agents(
filter: JSON = null
ids: [ID] = null
limit: Int = null
offset: Int = null
order: [String] = null
withArchived: WithArchived = null): AgentsCollection!
```

filter: JSON : Вы можете отфильтровать список агентов, указав значения для одного или нескольких параметров:

- **activations :**
- **cameras :** ID камеры, сохраненный в базе данных.
- **creation_date :** Точная дата создания агента в формате ISO 8601
- **id :** ID агента, сохраненный в базе данных.
- **info__title :** Название агента.
- **is_active :** Атрибут активности агента.
- **last_modified :** Точная дата последнего изменения агента в формате ISO 8601
- **workspace :** ID воркспейса
- **workspace_id :** ID воркспейса

ids: [ID] : Для получения списка конкретных агентов укажите их ID в списке.

limit: Int : Параметр позволяет получить первые n агентов из списка.

offset: Int : Параметр позволяет удалить первые n агентов из списка.

order: [String] : Вы можете отсортировать список по следующим параметрам: **activations, cameras, creation_date, id, info, is_active, last_modified, workspace, workspace_id.**

withArchived: WithArchived : Для получения списка всех агентов, в том числе заархивированных,укажите значение all . Для получения списка только заархивированных агентов укажите значение archived .

AgentsCollection! : Результат запроса - список агентов со следующими параметрами:

- **totalCount: Int :** Число возвращенных агентов
- **collectionItems: [AgentsCollection!]! :**
    - **id: ID! :** ID агента
    - **creationDate: DateTime! :** Дата создания агента в формате ISO 8601.

- **lastModified: DateTime! :** Дата последнего изменения агента в формате ISO 8601.
- **token: String! :** Токен агента
- **camerasIds: [ID!] :** Список ID камер, подключенных к агенту.
- **title: String :** Название агента
- **agentStatus: String! :** Статус агента
- **agentLastActiveTime: String :** Дата последней активности, зафиксированной камерой агента.
- **archived: Boolean! :** Атрибут архивации агента.

Пример запроса:

```
{
  agents {
    totalCount
    collectionItems {
      token
      title
      lastModified
      id
      creationDate
      camerasIds
      archived
      agentStatus
      agentLastActiveTime
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "agents": {
      "totalCount": 1,
      "collectionItems": [
        {
          "token": "19df2b38-7d23-44a5-85e7-c5a29b304581",
          "title": "My Agent",
          "lastModified": "2022-07-12T08:50:45.296225+00:00",
          "id": "19df2b38-7d23-44a5-85e7-c5a29b304581",
          "creationDate": "2022-07-07T07:23:08.965949+00:00",
          "camerasIds": [
            "3c818dc4-352c-47a7-b32b-465fbd4a9665"
          ],
          "archived": false,
          "agentStatus": "active",
          "agentLastActiveTime": "2022-07-12T08:50:45.296115"
        }
      ]
    }
  }
}
```

2.8.2 Создать агент

Мутация createAgent позволяет создать агент.

```
createAgent(agentData: AgentInput!): AgentCreateOutput!
```

agentData: AgentInput! : Информация, которая потребуется для создания агента:

- **title: String :** Название агента.
- **extra: JSON :** Дополнительная информация об агенте.

AgentCreateOutput : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean! :** Атрибут успешного завершения мутации.
- **agent: AgentOutput! :** Объект нового агента.
- **channelCost: String :** Ежемесячная плата за обслуживание.
- **writeoffDate: String :** Дата следующего обслуживания в формате ISO 8601.

Ошибки входных данных:

Ошибка при создании агента из-за превышения лимита созданных агентов:

```
{
  "message": "Agent limit exceeded",
  "code": "0x6245cd00"
}
```

Переданы неверные дополнительные данные:

```
{
  "message": "'int' object has no attribute 'get'"
}
```

Пример запроса:

```
mutation {
  createAgent(agentData: {title: "My new agent"}) {
    ok
    agent {
      id
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "createAgent": {
      "ok": true,
      "agent": {
        "id": "2b769e5a-a49a-46ac-ac38-ec924bd4ec83"
      }
    }
```

```
  }
}
```

2.8.3 Обновить агент

Мутация updateAgent используется для обновления информации об агенте.

```
updateAgent(
agentData: AgentUpdateInput!
agentId: ID!): AgentManageOutput!
```

agentData: AgentUpdateInput! : Информация, которая потребуется для обновления агента:

- **title: String :** Название агента.

agentId: ID! : ID агента, который необходимо обновить.

AgentManageOutput : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean! :** Атрибут успешного завершения мутации.
- **agent: AgentOutput! :** Объект обновленного агента.

Ошибки входных данных:

Агент не найден по переданному ID:

```
{
  "message": "Camera matching query does not exist."
}
```

Пример запроса:

```
mutation {
  updateAgent(
    agentId: "19df2b38-7d23-44a5-85e7-c5a29b304581"
    agentData: {title: "My old agent"}
  ) {
    ok
    agent {
      id
      title
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "updateAgent": {
      "ok": true,
      "agent": {
        "id": "19df2b38-7d23-44a5-85e7-c5a29b304581",
        "title": "My old agent"
```

```
      }
    }
  }
}
```

2.8.4 Удалить агент

Мутация deleteAgent позволяет удалить один или несколько агентов.

```
deleteAgent(agentId: ID = "" agentIds: [ID!] = null): MutationResult!
```

agentId: ID : ID агента, который требуется удалить.

agentIds: [ID!] : Список ID агентов, которые требуется удалить.

MutationResult! : Результат мутации - JSON-файл со следующими параметрами:

- **ok: Boolean! :** Атрибут успешного завершения мутации.

Ошибки входных данных:

Не передан ID для удаления агента:

```
{
  "message": "One of the parameters agentId or agentIds is required",
  "code": "0xe509f74d"
}
```

Пример запроса:

```
mutation {
  deleteAgent(agentId: "19df2b38-7d23-44a5-85e7-c5a29b304581") {
    ok
  }
}
```

Пример ответа:

```
{
  "data": {
    "deleteAgent": {
      "ok": true
    }
  }
}
```

2.9 Другое

2.9.1 Получить информацию о пользователе

Запрос me позволяет получить информацию об авторизованном пользователе.

```
me: UserType!
```

UserType! : Результат запроса - список ссылок:

- **username: String! :** Логин.
- **email: String! :** Почта пользователя.
- **firstName: String! :** Имя пользователя.
- **lastName: String! :** Фамилия пользователя.
- **workspaces: [WorkspaceType!]! :** Информация о воркспейсах пользователя.

Пример запроса:

```
{
  me {
    email
    firstName
    lastName
    username
    workspaces {
      id
    }
  }
}
```

Пример ответа:

```
{
  "data": {
    "me": {
      "email": "aa@aa.ru",
      "firstName": "",
      "lastName": "",
      "username": "aa@aa.ru",
      "workspaces": [
        {
          "id": "8d100a02-1b5b-4da1-b645-9fd01ef09c77"
        }
      ]
    }
  }
}
```