FindFace

Выпуск 1.6

NtechLab

сент. 07, 2023

Может быть полезно:

Документация состоит из блоков информации. Используйте их, чтобы познакомиться с *FindFace Lite* или найти нужную часть быстрее.

Ниже кратко описано содержание каждого блока документации:

- Блок **Может быть полезно** содержит дополнительные статьи, которые могут помочь в работе с FindFace Lite.
- Блок **Подробнее о FindFace Lite** содержит информацию о сценариях использования FindFace Lite и функциональности, которая может быть полезна при использовании сервиса.
- Блок **С чего начать** проведет вас по шагам, которые помогут начать работу с FindFace Lite.
- Блок **Настройки** содержит информационные статьи, описывающие, какие настройки вы можете использовать для персонализации FindFace Lite.
- Блок **Документация по интеграции** включает статьи, описывающие методы интеграции Find Face Lite.

Глава 1

Словарь терминов

FindFace Lite
FindFace Lite — упрощенный вариант FindFace Multi.

Файл инсталлятора
Файл инсталлятора — файл, отвечающий за установку FindFace Lite.

Терминал доступа
Терминал доступа — устройство контроля доступа с возможностью распознавать лица.

VideoWorker
Интерфейсный объект для трекинга, обработки и распознавания лиц на нескольких видеопотоках.

Лайвнес
(от англ. liveness). Технология проверки объекта на "живость": система определяет, взаимодействует ли она с человеком, имеющим доступ или мошенником, использующим поддельный идентификатор: фотографию лица, видео.

Event
Объект в системе, соответствующий результату фиксации объекта (лица или автомобиля) в кадре камеры. При активированном объекте Camera создается автоматически из данных обработчика видеопотоков VideoWorker.

Card
Объект в системе, соответствующий профилю человека или автомобиля в FindFace Lite. Он может быть двух типов: *face* — для лица или *car* — для автомобиля.

Camera
Образ видеопотока или файла, передающего видео в сервис.

Object
Объект в системе, соответствующий реальному лицу или автомобиля. Чтобы создать объект необходимо добавить соответствующее изображение и ссылку на Card, к которому необходимо привязать Object.

Вебхук
: Механизм отправки уведомлений при наступлении в системе события, на которое подписано клиентское приложение.

СКУД
: СКУД — это особый тип системы контроля доступа, используемый в качестве электронной меры безопасности. СКУД можно использовать для контроля доступа сотрудников и посетителей на объект и в контролируемые внутренние зоны.

Дедупликация
: Дедупликация — это функциональность, которая используется для предотвращения распознавания одного человека или автомобиля как нескольких разных Event'ов в течение определенного промежутка времени

Антиспам Event'ов
: Antispam is a feature that is used for distinguishing of a real Event to process and so called "spam" Event fixed in the system accidentally.

Периферийные устройства
: Периферийное устройство — это физическое устройство (например, терминалы доступа), которое может подключаться и отправлять изображения в FindFace Lite для распознавания объектов.

Глава 2

Сценарии использования

FindFace Lite может легко интегрироваться в существующие бизнес системы и отправлять туда обработанные данные видеоаналитики с подключенных камер и терминалов доступа.

При этом бизнес логика системы не подвергается изменениям, но в тоже время обогащается новыми данными видеоаналитики.

Сценарии использования FindFace Lite делятся на две категории:

- Сценарии для автомобиля
- Сценарии по лицу

Совет: Все сценарии могут быть дополнены интегрированными в FindFace Lite опциями, которые описаны в статье **Функциональность**.

2.1 Сценарии для автомобиля

FindFace Lite обрабатывает видеопоток, определяет параметры автомобиля и отправляет события в Систему Контроля Управления Доступом (СКУД) через webhook в JSON формате. Основывая на информаций от FindFace Lite СКУД осуществляет дальнейший сценарии по предоставлению доступа.

FindFace Lite предполагает два варианта взаимодействия со СКУД:

2.1.1 Распознавание только номера государственного регистрационного знака

Если СКУД принимает решение о предоставлении доступа, основываясь только на номере государственного регистрационного знака (ГРЗ) и не использует другие параметры автомобиля.

FindFace Lite отправляет событие в СКУД с номером ГРЗ, фотографией автомобиля, фотографией ГРЗ, датой, временем и ID камеры (для идентификации в СКУД)

2.1.2 Распознавание множества параметров автомобиля

Сложный сценарий, который использует множество дополнительных параметров, помимо ГРЗ:

Параметр	Сценарий
Передняя и **задняя** части автомобиля	Чтобы избежать ложных срабатываний, когда камера видит ГРЗ на задней части автомобиля после проезда, через шлагбаум.
	Чтобы избежать ложных срабатываний при движении задним ходом.
Тип автомобиля	Для того, чтобы удовлетворять муниципальные требования для спецтранспорта (например машина скорой помощи или пожарная машина).
Видимость **ГРЗ**	Чтобы избежать ложных срабатываний, когда автомобиль еще слишком далеко, либо припаркован в стороне от проезда, но в зоне видимости камеры.

2.2 Сценарии по лицу

FindFace Lite исключает любые случаи мошенничества или ложного срабатывания, используя продвинутый фунционал системы, такой как **Liveness** и **Headpose**.

Более того FindFace Lite может контролировать наличие медицинской маски у сотрудников, а также легко интегрируется с терминалами доступа по биометрии.

2.2.1 Сценарии со СКУД

FindFace Lite обрабатывает видеопоток, определяет параметры человека и отправляет событие в СКУД, через вебхук в JSON формате.

Основываясь на информации от FindFace Lite СКУД осуществляет дальнейший сценарий по предоставлению доступа.

2.2.2 Отслеживание времени

FindFace Lite позволяет собирать информацию по времени работы сотрудника, через приложение для Chrome на разных устройствах и при необходимости осуществляет экспорт в систему учета и планирования рабочего времени (ERP, WFM).

Сотрудник может легко поставит отметки о начале и конце рабочего дня нажав на кнопку и посмотрев в камеру. Продвинутая система защиты от фальсификации Liveness предотвращает любые ложные срабатывания.

2.2.3 Сценарии белого и черного списка

FindFace Lite может быть интегрирован с CRM системой и системой безопасности, чтобы осуществлять сценарии управления доступом:

- Когда клиент или любой гостя из белого списка приходит, FindFace Lite может взаимодействовать с программой лояльности, отправлять в CRM (или другую систему) событие с ID клиента и другими данными из его карточки.
- Когда приходит человек из черного списка, FindFace Lite отправляет событие во внешнюю систему для осуществления необходимого сценария.
- FindFace Lite может контролировать наличие у сотрудника наличие медицинской маски и регистрировать все фиксировать всю необходимую информацию о нем (изображение без маски, ID сотрудника итд.)

Глава 3

Функциональность

3.1 Дедупликация Event'ов `car face`

Дедупликация Event'ов — функциональность, которая используется для предотвращения распознавания одного человека или автомобиля как нескольких разных событий в течение определенного периода времени и упрощает интеграцию с СКУД, не поддерживающими дедупликацию.

Примечание: Дедупликация работает для событий со всех подключенных к системе камер.

Это означает, что в FindFace Lite вы можете установить период времени, в течение которого Event'ы с одним и тем же человеком или автомобилем будут считаться дубликатами. Система будет реагировать только на первое событие, а последующие повторяющиеся будут игнорироваться.

3.1.1 Возможные сценарии дупликации Event'ов

- События с **автомобилем** могут дублироваться, если транспортное средство останавливается на некоторое время перед шлагбаумом, или если камера выезда распознает задний номер автомобиля, уже проехавшего через шлагбаум.
- События с **лицом** могут дублироваться, если камера зафиксировала лицо человека, уже прошедшего через СКУД, который по какой-либо причине снова посмотрел в камеру.

Больше о сценариях использования сервиса читайте в *статье*.

3.1.2 Как настроить дедупликацию

1. Откройте конфигурационный файл, расположенный в **FFlite -> api_config.yml**, используя один из редакторов (например, команду `nano`).
2. Включите эту функциональность, установив для параметра **dedup_enabled** значение *true*.
3. Настройте параметры сохранения для дубликатов в **save_dedup_event**: выберите *true*, чтобы сохранить дубликаты, и *false*, чтобы не сохранять.
4. Задайте параметры для распознавания **car** – автомобилей и **faces** – лиц отдельно.

Параметры	Значение по умолчанию	Описание
face_dedup_confidence	0.9	Порог совпадения между двумя дублирующими Event'ами. Если результат совпадения Event'ов равен или превышает установленное значение, событие помечается как дублирующее.
car_dedup_confidence	0.9	
face_dedup_interval	5	Временной интервал в секундах, в течение которого несколько Event'ов с одним автомобилей или лицом отмечаются как дубликаты.
car_dedup_interval:	5	

5. Сохраните изменения в файле и закройте редактор.
6. Примените новые параметры, перезагрузив `сервис api`. Для этого используйте команду:

```
docker compose restart api
```

Совет: Читайте подробное описание работы с конфигурационным файлом в *статье*.

3.2 Антиспам Event'ов `car face`

Антиспам Event'ов — это функциональность, которая используется для того, чтобы отличить реальный Event, подлежащий обработке, от «спамового» Event'а, зафиксированного в системе по ошибке.

Это означает, что в момент создания Event'а камеры или отправки Event'а с *периферийного устройства через POST-запрос* вы можете установить область обнаружения изображения и опустить ее спам-часть.

3.2.1 Возможные сценарии фиксации спам Event'ов

- Event'ы с **автомобилем** могут считаться спам-событиями, если транспортное средство припарковано возле шлагбаума и номерной знак попадает в поле зрения видеокамеры, в таком случае без использования этой функциональности будет сгенерировано несколько Event'ов.

  Если номерной знак найден в базе данных СКУД, шлагбаум будет открыт.

- События с **лицом** могут считаться спам-событиями, если человек находится рядом с камерой, но не собирается проходить через СКУД. В таком случае, если человек будет найден в базе данных СКУД, доступ на закрытую территорию будет открыт.

Больше о сценариях использования сервиса читайте в *статье*.

3.2.2 Как настроить функциональность

Настроить область захвата кадра можно в параметре **roi**, где необходимо указать расстояние от каждой стороны кадра из видеопотока или изображения, полученного от периферийных устройств.

Совет: В настоящее время настройки доступны только в запросах через API

- Чтобы настроить параметр **roi** для **видеопотоков**, используйте запросы ниже и формат **WxH+X+Y** для параметра **roi**:
  - `POST /v1/cameras/` для создания объекта Camera.
  - `PATCH /v1/cameras/{camera_id}` для обновления объекта Camera.

```
{
  "name": "test cam",
  "url": "rtmp://example.com/test_cam",
  "active": true,
  "single_pass": false,
  "stream_settings": {
    "rot": "",
    "play_speed": -1,
    "disable_drops": false,
    "ffmpeg_format": "",
    "ffmpeg_params": [],
    "video_transform": "",
    "use_stream_timestamp": false,
    "start_stream_timestamp": 0,
    "detectors": {
      "face": {
        "roi": "1740x915+76+88", <<-- roi
        "jpeg_quality": 95,
        "overall_only": false,
        "filter_max_size": 8192,
        "filter_min_size": 1,
        "fullframe_use_png": false,
        "filter_min_quality": 0.45,
        "fullframe_crop_rot": false,
        "track_send_history": false,
        "track_miss_interval": 1,
        "post_best_track_frame": true,
        "post_last_track_frame": false,
        "post_first_track_frame": false,
        "realtime_post_interval": 1,
        "track_overlap_threshold": 0.25,
        "track_interpolate_bboxes": true,
        "post_best_track_normalize": true,
        "track_max_duration_frames": 0,
        "realtime_post_every_interval": false,
        "realtime_post_first_immediately": false
      }
    }
  }
}
```

- Чтобы настроить параметр **roi** для **изображений**, полученных с периферийных устройств, используйте запрос ниже и формат **[left, top, right, bottom]** для параметра **roi**:
  - `POST /v1/events/{object_type}/add` для создания Event'а через POST-запрос.

```
{
  "object_type": "face",
  "token": "change_me",
  "camera": 2,
  "fullframe": "somehash.jpg",
  "rotate": true,
  "timestamp": "2000-10-31T01:30:00.000-05:00",
  "mf_selector": "biggest",
  "roi": 15,20,12,14 <<-- roi
}
```

3.3 Лайвнес `face`

Лайвнес — технология, используемая в камерах видеонаблюдения для определения того, принадлежат ли биометрические данные, представленные системе, живому человеку или это попытка подделки, например, фотография.

FindFace Lite использует пассивный метод применения Лайвнес. Метод использует алгоритмы для анализа различных особенностей лица человека, таких как движение зрачков, выражение лица или наличие микродвижений, чтобы определить живого человека.

Этот метод требует меньше подготовки и использования сторонних средств, чем другие, а также:

- не требует дополнительных действий от пользователя.
- работает в режиме реального времени: может быстро и точно аутентифицировать пользователя, не вызывая задержек или сбоев в процессе аутентификации.
- не требует дополнительного оборудования или датчиков;
- его сложнее подделать, чем другие методы, например, те, которые требуют от пользователя выполнения определенного действия.

3.3.1 Возможные сценарии для применения Лайвнес

Лайвнес можно использовать в следующих сценариях:

- Банковская сфера: для подтверждения личности пользователя.

  Например, попросить пользователей показать свое лицо на камеру во время видеозвонка с банковским сотрудником, в течение которого система с использованием Лайвнес функциональности поможет удостовериться, что пользователь реален и не использует поддельное изображение.

- Контроль доступа сотрудников: для осуществления контроля доступа к рабочему месту.

  Например, если сотрудник пытается войти на защищенную территорию с помощью фотографии, камера видеонаблюдения может передать соответствующий сигнал и СКУД не пропустит таких пользователей.

3.3.2 Как настроить Лайвнес

1. Откройте конфигурационный файл, расположенный в **FFlite -> api_config.yml**, используя один из редакторов (например, команду `nano`).
2. Включите эту функциональность, добавив значение **liveness** к параметру **face_features**.
3. Настройте источник для Лайвнес в параметрах **liveness_source**.

- Если источником для функциональности будет картинка (Event'ы будут создаваться через POST */{object_type}/add*), установите значение **eapi**.
- Если источником для обнаружения будет видеопоток (Event'ы будут создаваться в FindFace Lite автоматически после распознания из видеопотока), установите значение **vw**.

4. Сохраните изменения в файле и закройте редактор.
5. Примените новые параметры, перезагрузив `сервис api`. Для этого используйте команду:

```
docker compose restart api
```

Совет: Читайте подробное описание работы с конфигурационным файлом в *статье*.

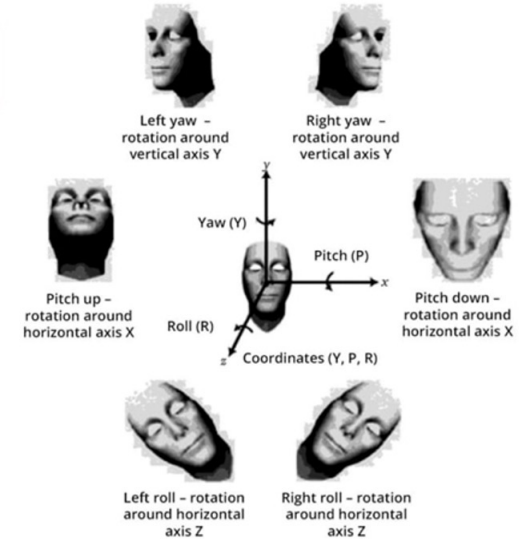
3.4 Положение головы `face`

Функциональность распознавания положения головы — это возможность камеры обнаруживать и отслеживать положение и движение головы человека относительно камеры видеонаблюдения в реальном времени.

Предупреждение: Функциональность распознавания положения головы не работает, если человек носит медицинскую маску.

Для распознавания положения головы в двумерном пространстве FindFace Lite использует параметры **pitch** and **yaw**.

- **Pitch** — угол наклона головы вверх/вниз (т.е.относительно горизонтальной оси). Положительный **pitch** показывает, что голова наклонена вперед, а отрицательный указывает на наклон головы назад.
- **Yaw** — угол поворота головы вправо/влево (т.е. относительно вертикальной оси). Положительный **yaw** указывает на то, что голова повернута направо, а отрицательный — налево.



3.4.1 Возможные сценарии использования распознавания положения головы

Функциональность распознавания положения головы можно использовать в сценариях для улучшения точности и повышения безопасности, например:

- Улучшение системы контроля доступа сотрудников с помощью фиксации того, что лицо сотрудника совпадает с ожидаемым положением.

  Это значит, что если человек стоит рядом с камерой, поворачивает голову к камере, но не собирается проходить через СКУД, доступ не будет предоставлен благодаря установленному заранее значению положения головы.

- Повышение комфорта сотрудников во время использования СКУД системы. Если сотрудник подходит к контрольно-пропускному пункту под неудобным углом, камера видеонаблюдения может подать сигнал системе контроля доступа, чтобы переместить камеру.

3.4.2 Как настроить функциональность распознавания положения головы

1. Откройте конфигурационный файл, расположенный в **FFlite -> api_config.yml**, используя один из редакторов (например, команду `nano`).
2. Включите эту функциональность, установив для параметра *face_features** значение *headpose*.
3. Сохраните изменения в файле и закройте редактор.
4. Примените новые параметры, перезагрузив `сервис api`. Для этого используйте команду:

```
docker compose restart api
```

Совет: Читайте подробное описание работы с конфигурационным файлом в *статье*.

Глава 4

Введение к блоку

Блок **С чего начать** содержит 5 статей, описывающих шаги установки FindFace Lite. Следуйте инструкции, чтобы быстро и правильно установить *FindFace Lite*.

Для корректной работы сервиса необходимо:

- Подготовить камеру видеонаблюдения, опираясь на требуемые характеристики (*ШАГ 1*).
- Проверить конфигурацию сервера на соответствие требованиям FindFace Lite (*ШАГ 2*).
- Подготовить сервер, установив Docker Engine и Docker Compose (*ШАГ 3*).

Когда подготовка завершена:

- Загрузить файл инсталлятора FindFace Lite и лицензию сервиса на сервер (*ШАГ 4*).
- Установить FindFace Lite (*ШАГ 5*).

Удачи!

Если у вас возникнут вопросы, пожалуйста, напишите нам на **support@ntechlab.com**

Глава 5

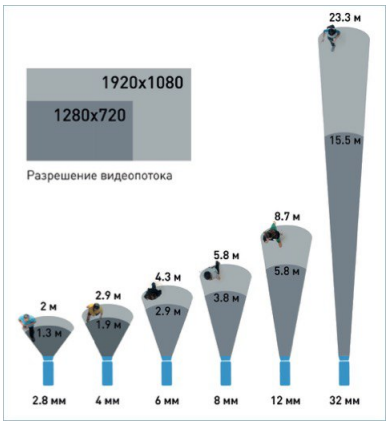
ШАГ 1. Требования к камерам видеонаблюдения: характеристики и установка

5.1 Распознавание по лицу

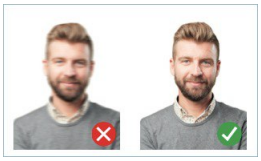
5.1.1 Характеристики камер видеонаблюдения

1. Минимальная плотность пикселей для идентификации – 500 пикселей/м (примерно соответствует ширине лица 80 пикселей).

2. Фокусное расстояние объектива должно выбираться таким образом, чтобы при заданном расстоянии до объектов съемки обеспечивалась необходимая плотность пикселей. На рисунке ниже приведен пример расчета фокусного расстояния объектива от расстояния между камерой и объектами съемки. Для расчета фокусного расстояния для конкретной камеры требуется использовать калькуляторы или методологию, предоставляемые производителем камеры.



3. Экспозиция должна быть настроена таким образом, чтобы лица были резкими ("в фокусе"), не смазанными и равномерно освещенными (не засвеченными и не слишком темными).



4. В зависимости от условий освещения (яркая засветка, слишком яркое или слишком тусклое освещение) рекомендуется использовать камеры с аппаратным WDR (Wide Dynamic Range) или другими технологиями, обеспечивающими компенсацию встречной засветки и/или слабой освещенности (BLC, HLC, DNR, высокая светочувствительность, Smart ИК-подсветка, AGC и др.).

5. Сжатие видео: большинство видеоформатов и кодеков, которые могут быть декодированы FFmpeg FFmpeg.

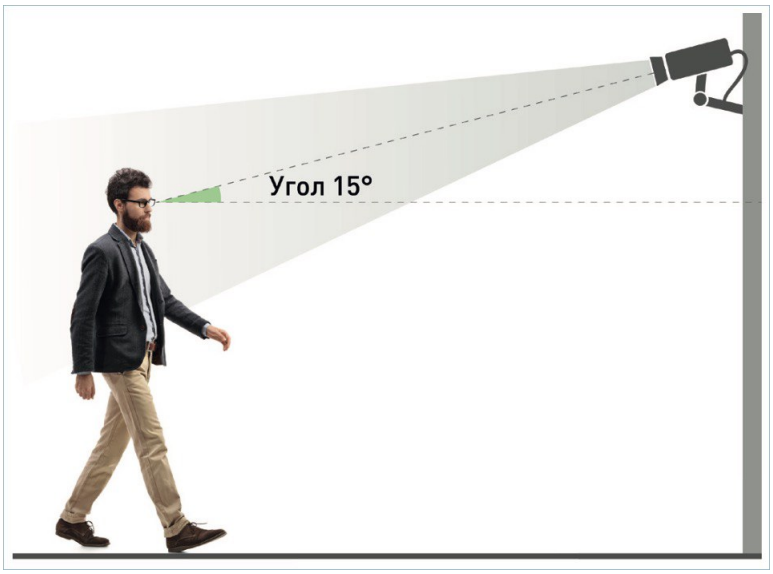
6. Протоколы передачи видеопотока: RTSP, HTTP.

Совет: Для расчета точной конфигурации оборудования в соответствии с вашими целями обратитесь к нашим специалистам по адресу support@ntechlab.com.

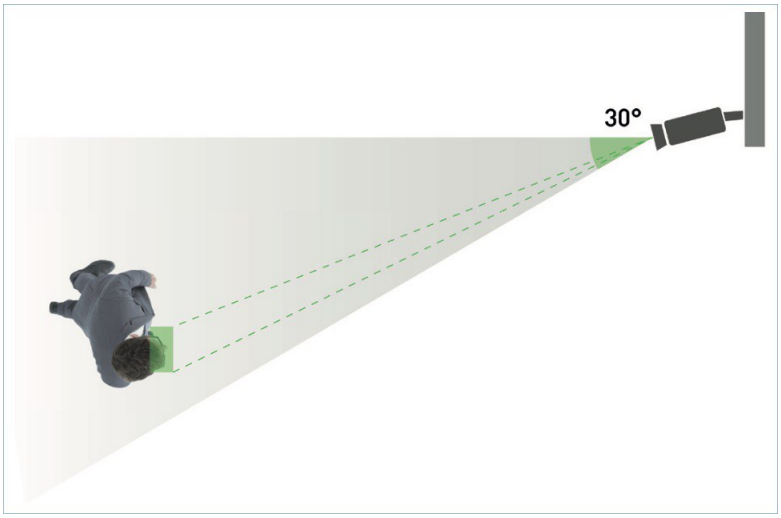
5.1.2 Установка камер видеонаблюдения

1. Для корректной детекции в видеопотоке установите камеру таким образом, чтобы в ее поле зрения обязательно появлялось лицо каждого человека, входящего в зону наблюдения.

2. Угол вертикального наклона видеокамеры не должен превышать 15°. Вертикальный наклон — это отклонение оптической оси видеокамеры от горизонтальной плоскости, расположенной на уровне середины лица человека среднего роста (160 см).



3. Угол горизонтального отклонения не должен превышать 30°. Горизонтальное отклонение — это отклонение оптической оси видеокамеры от вектора движения основного потока объектов распознавания.



5.2 Распознавание автомобиля

5.2.1 Характеристики камер видеонаблюдения

5.2.1.1 Характеристики камер видеонаблюдения

Для корректной работы FindFace Lite требуется следующая конфигурация.

Требования к объекту в кадре

Параметр	Минимальные требования	Рекомендуемые требования
Размер объекта: ширина ТС	>= 80 пикселей	>= 120 пикселей
Размер объекта: ширина ГРЗ	>= 100 пикселей	>= 150 пикселей
Размер объекта: ширина ТС + ГРЗ	>= 340 пикселей	>= 340 пикселей
Допустимое перекрытие объекта	<= 30%	<= 15%

Требования к камерам (к цифровому изображению)

Параметр	Минимальные требования	Рекомендуемые требования
Размер матрицы	>= 1/2,8	>= 1/1,8
Фокусное расстояние	>= 1,5 мм	>= 4 мм
Светочувствительность (цвет)	<= 0.1 люкс	<= 0,05 люкс
Поддержка протокола TCP	Да	Да
Разрешение потока	>= 720x576	>= 1920x1080
Качество потока	3000-4000 кбит/с	>= 4000 кб/с
Частота кадров	>= 15	>= 50-50
Скорость затвора	до 1/100	до 1/500
Поддержка H.264	H.264	H.264,H.265
Регулировка частоты опорного кадра	Да	Да
Поддержка WDR	Да	Да (до 120 дБ)
Регулировка диафрагмы	Неважно	Да
Регулировка фокусного расстояния	Неважно	Да
Механический ИК-фильтр	Неважно	Да
Поддержка ONVIF	Неважно	Да

Монтаж камеры (допустимые повороты объекта в кадре)

Параметр	Минимальные требования	Рекомендуемые требования
Угол вертикального наклона камеры	<= 45°	<= 30°
Угол горизонтального наклона камеры (ТС)	Неважно	Неважно
Угол горизонтального наклона камеры (ГРЗ)	<= 30°	<= 15°

Требования к освещению

Параметр	Минимальные требования	Рекомендуемые требования
Освещение в зоне распознавания	>= 150 люкс	>= 200 люкс
Компенсация обратной засветки	<= 200 люкс	<= 100 люкс

Требования к потоку для камеры

Настройки	Параметр	Рекомендуемые требования
Экспозиция	Iris mode	auto
	Auto iris level	50
	Exposure time	1/200
	Gain	25
Фокус камеры	Фокус камеры	Настройка вручную под конкретную сцену
*Настройка работы с засветкой**	BLC	OFF
	WDR	OFF
	HLC	ON (для камер шлагбаумов, если не требуется собирать атрибуты автомобиля)

5.2.1.2 Прочие характеристики

ИК-подсветка

Рекомендуемая опция для камеры. Номерная пластина отлично отражает свет, и в условиях недостаточной освещённости, свет от ИК прожектора камеры будет отлично подсвечивать номер автомобиля.

Примечание: характеристики оборудования для ИК-подсветки, которые указывает производитель (например 10,20,50м) это дальность подсветки до полного угасания.

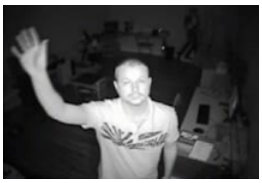
Таким образом, эффективная дальность, как правило, на 30-40% ниже. Важно это учитывать.

SmartIR

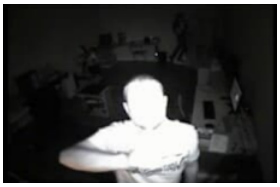
При наличии в камере ИК прожектора, функция SmartIR может значительно улучшить качество изображения в условиях плохой освещенности.

SmartIR позволяет снижать интенсивность подсветки, если объект находится слишком близко и кадр получается переэкспонированным.

Smart IR



Без Smart IR



WDR (Wide Dynamic Range)

Используйте настройку этого параметра для регулировки баланса белого.

BLC (Blacklight Compensation)

Используйте настройку этого параметра для устранения проблем фоновой засветки.

HLC (Highlight Compensation)

Используйте HLC настройку для компенсации светлых участков. HLC автоматически обнаруживает сильные источники света и уменьшает засветку, значительно улучшая четкость ярких областей.

При активации HLC, камера будет учитывать яркие области, похожие на прожектор, и соответствующим образом корректировать экспозицию.

С функцией HLC камера будет пытаться правильно экспонировать всю сцену, уменьшая яркость светлых участков.

5.2.2 Установка камер видеонаблюдения

Объект в кадре должен быть детализированным, в фокусе (детали четко видны), без размытости и с высоким уровнем контрастности.

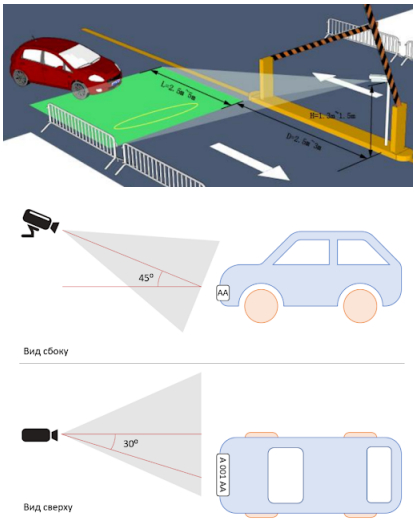
Для правильной работы видеоаналитики используйте рекомендации ниже.

5.2.2.1 Общие рекомендации по установке

- **Расстояние от камеры до места распознавания может быть условно любым**. В зависимости от дальности подбираются камеры с соответствующим объективом.
- **Камера должна быть установлена к неподвижной жесткой конструкции**.
- **Избегайте попадания солнечных лучей или избытка света** в объектив камеры, это может приводить к засветки изображения.
- **Объект в кадре** должен быть полностью виден. Центральная ось камеры должна приближаться к центру зоны распознавания, чтобы объект находился в центре кадра.
- **Разрешение и объектив камеры** должны обеспечивать резкое, не замыленное изображение объекта, а также отсутствие видимого смаза и дисторсии изображения.

5.2.2.2 Установка при шлагбауме

Параметры	Требования
Высота установки (H)	1.3-1.5 м
Дистанция от камеры до нижнего угла изображения (D)	2.5 - 3 м.
Дистанция видимой зоны кадра (L)	2.5-5м



Глава 6

ШАГ 2. Требования к серверу и ПО

6.1 Требования к серверу

Для работы *FindFace Lite* требуется различное аппаратное обеспечение в зависимости от типа обрабатываемой информации (изображения или видеопотоки) и от количества используемых камер и *терминалов доступа*.

6.1.1 Обработка изображений

Для обработки изображения, во время которого *терминал доступа* фиксирует лицо и отправляет результат в FindFace Lite, требования к серверу следующие:

Количество устройств	Физические ядра Intel >2.4 ГЦ	RAM, ГБ
1 – 8	4	6
8 – 16	4 – 6	8
16 – 24	4	8

6.1.2 Обработка HD (720p) видеопотоков

Для обработки HD (720p) видеопотоков с 20 FPS, требования к серверу следующие:

Количество устройств	Физические ядра Intel >2.4 ГЦ	RAM, ГБ
1	4	8
5	8	10
10	14	16

6.1.3 Обработка FHD (1080p) видеопотоков

Для обработки FHD (1080p) видеопотоков с 20 – 25 FPS, требования к серверу следующие:

Количество устройств	Физические ядра Intel >2.4 ГЦ	RAM, ГБ
1	6	8
5	16	10
10	24	16

6.2 Требования к программному обеспечению

ПО	Рекомендации
Операционная система	Ubuntu 18.04 x64, CentOS 7 и другие похожие ОС.
Командная строка	Поддерживается использование только командных строк Linux.
Docker Engine	Версия, начиная с 19.03
Docker Compose	Версия, начиная с 2.2.3
Для корректной установки Docker, пожалуйста, следуйте инструкции *в Шаге 3*.	
NVIDIA Container Toolkit	Требуемая версия 1.7.0+ (nvidia-docker2 >= 2.8.0). Требование актуально только для **GPU-сервера**.

Глава 7

ШАГ 3. Подготовка сервера

7.1 Подготовка CPU-сервера

Чтобы подготовить CPU-сервер, установите Docker Engine (19.03+) и Docker Compose (2.2.3+) .

Перед первичной установкой Docker Engine (19.03+) and Docker Compose (2.2.3+) подключите репозиторий Docker.

Шаги установки программного обеспечения (от **репозитория Docker** до **Docker Engine** и **Docker Compose**) описаны ниже для Ubuntu OS и CentOS.

7.1.1 Ubuntu OS

1. Обновите пакетный менеджер `apt` и установите пакеты, необходимые для шифрования данных и использования репозитория через HTTPS:

```
sudo apt-get update
sudo apt-get install \
    ca-certificates \
    curl \
    gnupg \
    lsb-release
```

2. Добавьте GPG ключ, предоставленный Docker. Используйте команду ниже, она уже содержит ключ:

```
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -
↪o /etc/apt/keyrings/docker.gpg
```

3. Подключите репозиторий, используя команду:

```
echo \
"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.
↪gpg] https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /
↪dev/null
```

4. Обновите пакетный менеджер `apt`:

```
sudo apt-get update
```

5. Установите **Docker** и **Docker Compose Plugin**:

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-
↪plugin
```

6. Проверьте корректность установки **Docker**, используя команду, которая загружает тестовый образ и открывает его в контейнере:

```
sudo docker run hello-world
```

После запроса контейнер запустится, отобразится сообщение об успешной операции, затем контейнер автоматически остановится.

```
~
❯ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:faa03e786c97f07ef34423fccceeec2398ec8a5759259f94d99078f264e9d7af
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

7. Убедитесь, что **Docker Compose Plugin** установлен корректно, для этого используйте команду проверки версии:

```
docker compose version
```

```
~
❯ docker compose version
Docker Compose version v2.2.3
```

Все требования для установки FindFace Lite соблюдены, пожалуйста, перейдите на следующий *ШАГ*, чтобы загрузить файл инсталлятора и лицензию на сервер.

7.1.2 CentOS

1. Установите пакеты `yum-utils` и установите репозиторий. Используйте команду ниже:

```
sudo yum install -y yum-utils
sudo yum-config-manager \
    --add-repo \
    https://download.docker.com/linux/centos/docker-ce.repo
```

2. Установите **Docker** и **Docker Compose Plugin**:

```
sudo yum install docker-ce docker-ce-cli containerd.io docker-compose-plugin
```

Примечание: Если будет предложено принять ключ GPG, убедитесь, что его публичный код соответствует значению **060A 61C5 1B55 8A7F 742B 77AA C52F EB6B 621E 9F35**, и если да, примите его.

---

3. Запустите Docker, используя команду:

```
sudo systemctl start docker
```

4. Проверьте корректность установки **Docker**, используя команду, которая загружает тестовый образ и открывает его в контейнере:

```
sudo docker run hello-world
```

После запроса контейнер запустится, отобразится сообщение об успешной операции, затем контейнер автоматически остановится.

```
~
❯ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:faa03e786c97f07ef34423fccceeec2398ec8a5759259f94d99078f264e9d7af
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

5. Убедитесь, что **Docker Compose Plugin** установлен корректно, для этого используйте команду проверки версии:

```
docker compose version
```

Все требования для установки FindFace Lite соблюдены, пожалуйста, перейдите на следующий *ШАГ*, чтобы загрузить файл инсталлятора и лицензию на сервер.

7.2 Подготовка GPU-сервера

Чтобы подготовить GPU-сервер, установите NVIDIA Container Toolkit.

Перед первичной установкой NVIDIA Container Toolkit, проверьте сервер на наличие следующих требований:

- Драйверы NVIDIA Linux >= 418.81.07 (обратите внимание, что более старые драйверы не поддерживаются, чтобы установить актуальные драйверы, перейдите на официальный сайт NVIDIA);
- Архитектура NVIDIA GPU >= Kepler;
- Версия Docker >= 19.03.

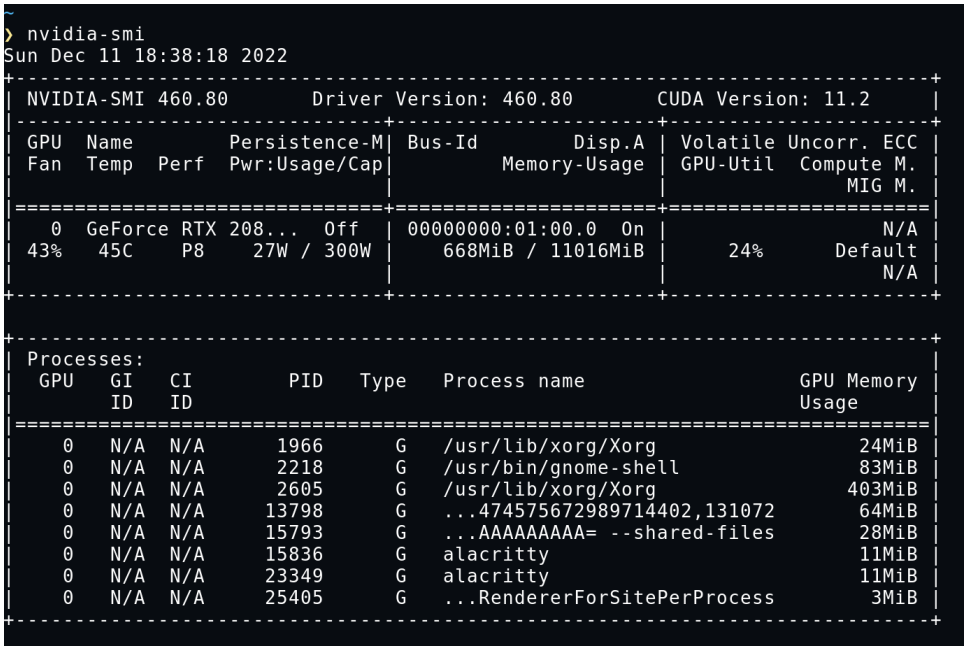
Шаги установки программного обеспечения (от проверки GPU сервера на соответствие требованиям до установки **Docker** и **NVIDIA Container Toolkit**) описаны ниже для Ubuntu OS и CentOS.

7.2.1 Ubuntu OS

1. Проверьте версию драйверов сервера, используя команду ниже:

```
nvidia-smi
```

Значение параметра **Driver Version:** должно быть >= 418.81.07.



Если значение версии не совпадает с требуемым, пожалуйста, обновите ПО, используя официальную инструкцию NVIDIA .

2. Проверьте модель видеокарты, используя команду ниже:

```
nvidia-smi -L
```

3. Убедитесь, что архитектура видеокарты >= **Kepler**. Для этого воспользуйтесь таблицей ниже:

Архитектура (от старой к новой)	**Модель**
Fermi	GeForce 400 and 500: GTX 480, GTX 470, GTX 580, GTX 570;
Kepler	GeForce 600 and 700: Nvidia GTX 680, 670, 660, GTX 780, GTX 770;
Maxwell	GeForce 900: GTX 960, GTX 970, GTX 980;
Pascal	GeForce 1000: GTX 1050, 1050 Ti, 1060, 1080;
Turing	GeForce RTX 2000 and GTX 1600: GTX 1660, GTX 1650, RTX 2060, RTX 2080;
Ampere	GeForce RTX 3080, RTX 3090, RTX 3070, etc.

4. Установите последнюю версию программного обеспечения **Docker**, используя команду ниже:

```
  curl https://get.docker.com | sh \

&& sudo systemctl --now enable docker
```

> **Предупреждение:** Если команда не работает, пожалуйста, установите Docker, следуя шагам установки, описанным в *разделе установки Docker на CPU-сервер*.

5. Добавьте **репозиторий NVIDIA** и **GPG ключ**:

```
distribution=$(. /etc/os-release;echo $ID$VERSION_ID) \
  && curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg␣
↪--dearmor -o /usr/share/keyrings/nvidia-container-toolkit-keyring.gpg \
  && curl -s -L https://nvidia.github.io/libnvidia-container/$distribution/
↪libnvidia-container.list | \
        sed 's#deb https://#deb [signed-by=/usr/share/keyrings/nvidia-
↪container-toolkit-keyring.gpg] https://#g' | \
        sudo tee /etc/apt/sources.list.d/nvidia-container-toolkit.list
```

6. Обновите пакетный менеджер `apt`:

```
sudo apt-get update
```

7. Установите пакет **nvidia-docker2** и все, что к нему относится, используя команду ниже:

```
sudo apt-get install -y nvidia-docker2
```

8. Перезапустите **Docker daemon**, чтобы завершить установку:

```
sudo systemctl restart docker
```

9. Проверьте наличие компонента “runtime” в конфигурационном файле:

```
cat /etc/docker/daemon.json
```

   Если в выводе присутствует компонент **nvidia-container-runtime**, установка прошла успешно.

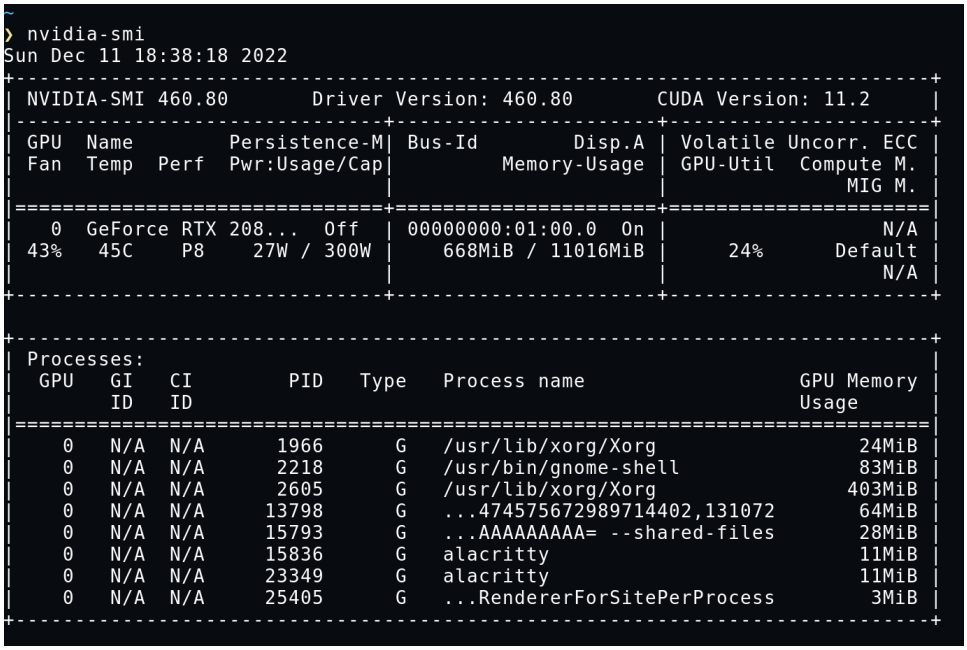
Все требования для установки FindFace Lite соблюдены, пожалуйста, перейдите на следующий *ШАГ*, чтобы загрузить файл инсталлятора и лицензию на сервер.

7.2.2 CentOS

1. Проверьте версию драйверов сервера, используя команду ниже:

```
nvidia-smi
```

   Значение параметра **Driver Version:** должно быть >= 418.81.07.



Если значение версии не совпадает с требуемым, пожалуйста, обновите ПО, используя официальную инструкцию NVIDIA .

2. Проверьте модель видеокарты, используя команду ниже:

```
nvidia-smi -L
```

3. Убедитесь, что архитектура видеокарты >= **Kepler**. Для этого воспользуйтесь таблицей ниже:

Архитектура (от старой к новой)	**Модель**
Fermi	GeForce 400 and 500: GTX 480, GTX 470, GTX 580, GTX 570;
Kepler	GeForce 600 and 700: Nvidia GTX 680, 670, 660, GTX 780, GTX 770;
Maxwell	GeForce 900: GTX 960, GTX 970, GTX 980;
Pascal	GeForce 1000: GTX 1050, 1050 Ti, 1060, 1080;
Turing	GeForce RTX 2000 and GTX 1600: GTX 1660, GTX 1650, RTX 2060, RTX 2080;
Ampere	GeForce RTX 3080, RTX 3090, RTX 3070, etc.

4. Добавьте **репозиторий Docker**:

```
sudo yum-config-manager --add-repo=https://download.docker.com/linux/centos/
↪docker-ce.repo
```

5. Установите пакет `containerd.io`:

```
sudo yum install -y https://download.docker.com/linux/centos/7/x86_64/stable/
↪Packages/containerd.io-1.4.3-3.1.el7.x86_64.rpm
```

6. Установите последнюю версию программного обеспечения **Docker**, используя команду ниже:

```
sudo yum install docker-ce -y
```

7. Убедитесь, что Docker запущен. Для этого запустите команду:

```
sudo systemctl --now enable docker
```

8. Проверьте установку Docker с помощью команды “hello-world”:

```
sudo docker run --rm hello-world
```

После запроса контейнер запустится, отобразится сообщение об успешной операции, затем контейнер автоматически остановится.

```
~
❯ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:faa03e786c97f07ef34423fccceeec2398ec8a5759259f94d99078f264e9d7af
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

9. После установки **Docker** продолжите установку программного обеспечения **NVIDIA**. Добавьте репозиторий и GPG ключ, используя команду ниже:

```
distribution=$(. /etc/os-release;echo $ID$VERSION_ID) \
    && curl -s -L https://nvidia.github.io/libnvidia-container/$distribution/
↪libnvidia-container.repo | sudo tee /etc/yum.repos.d/nvidia-container-
↪toolkit.repo
```

10. Обновите пакетный менеджер `yam`:

```
sudo yum clean expire-cache
```

11. Установите пакет `nvidia-docker2` и все, что к нему относится, используя команду ниже:

```
sudo yum install -y nvidia-docker2
```

12. Перезапустите **Docker daemon**, чтобы завершить установку:

```
sudo systemctl restart docker
```

13. Проверьте наличие компонента “runtime” в конфигурационном файле:

```
cat /etc/docker/daemon.json
```

Если в выводе присутствует компонент **nvidia-container-runtime**, установка прошла успешно.

Все требования для установки FindFace Lite соблюдены, пожалуйста, перейдите на следующий *ШАГ*, чтобы загрузить файл инсталлятора и лицензию на сервер.

Глава 8

ШАГ 4. Загрузка файла инсталлятора FindFace Lite и лицензии на сервер

В этом шаге мы попросим вас загрузить файл инсталлятора и лицензию сервиса на ту машину, которую вы планируете использовать для работы с FindFace Lite. Вы можете использовать как физические сервер или ПК, так и виртуальные, соответствующие *требованиям*.

Чтобы загрузить файл инсталлятора и лицензию с локальной машины на виртуальную, используйте один из способов: *scp* команду или *один из SFTP менеджеров*.

Предупреждение: Важно поместить файл инсталлятора и лицензию в одну директорию.

8.1 Использование команды `scp`

Команда scp устанавливает защищенное соединение между двумя локациями с помощью SSH, что позволяет безопасно копировать файлы с одной машины на другую.

Предупреждение: Перед использованием команды `scp` убедитесь, что:

- Подключение по SSH доступно на обеих машинах: на той, с которой копируете и на той, куда копируете.
- У вас есть доступ типа “root” на обеих машинах.

Формат команды scp может отличаться в зависимости от типа аутентификации, которую вы используете для подключения по SSH.

Ниже расположена информация о том:

- как выглядит *общий формат команды scp*;
- *как выглядит команда, если для SSH аутентификации используется логин и пароль*;
- *как выглядит команда, если для SSH аутентификации используется пара ключей*.

8.1.1 Общий формат команды scp

```
scp [[user@]src_host:]file1 [[user@]dest_host:]file2
```

Где:

- `scp` инициализирует команду и устанавливает защищенное соединение.
- `src_host` — путь к файлу на машине, откуда вы копируете.
- `dest_host` — путь на машине, куда вы копируете файл.

8.1.2 Формат команды scp для SSH аутентификации с использованием логина и пароля

Чтобы скопировать файл с исходной машины на целевую, используйте следующий формат команды `scp`:

```
scp location/file_name.ext username@destination_host:/location
```

Где:

- `scp` инициализирует команду и устанавливает защищенное соединение.
- `location/file_name.ext` — путь до файла, который вы копируете, с указанием его полного названия.
- `username@destination_host` — данные для подключения к целевой машине: *username* — имя пользователя, *destination_host* — IP-адрес машины.
- `/location` — путь на машине, куда вы копируете файл.

Пример команды:

scp NtechLab_license_512b25bdcd334b44b87ccf5f089215b9.lic azureuser@00.00.000.000:home/azureuser

---

После исполнения команды необходимо ввести пароль. Обратите внимание, что пароль не отображается в процессе ввода.

Результат выполненной команды:

azureuser@00.00.000.000's password:

NtechLab_license_512b25bdcd334b44b87ccf5f089215b9.lic 100% 3672KB 126.6KB/s 00:29

8.1.3 Формат команды scp для SSH аутентификации с использованием пары ключей

Чтобы скопировать файл с исходной машины на целевую, используйте следующий формат команды `scp`:

```
scp -i ~/.ssh/private_key_name location/file_name.ext username@destination_host:/location
```

Где:

- `scp` инициализирует команду и устанавливает защищенное соединение.
- `~/.ssh/private_key_name` — путь к файлу приватного ключа в папке **ssh**. Эта часть команды отвечает за установление безопасного соединения с сервером.
- `location/file_name.ext` — путь до файла, который вы копируете, с указанием его полного названия.
- `username@destination_host` — данные для подключения к целевой машине: *username* — имя пользователя, *destination_host* — IP-адрес машины.
- `/location` — путь на машине, куда вы копируете файл.

Пример команды:

scp -i ~/.ssh/private NtechLab_license_512b25bdcd334b44b87ccf5f089215b9.lic azureuser@00.00.000.000:/home/azureuser

---

Результат выполненной команды:

azureuser@00.00.000.000:

NtechLab_license_512b25bdcd334b44b87ccf5f089215b9.lic 100% 3672KB 126.6KB/s 00:29

8.2 Использование файлового менеджера SFTP

1. Установите один из файловых менеджеров, например, FileZilla Client (доступен для Windows, Linux и Mac OS).
2. Подключитесь к серверу, используя имя пользователя и IP-адрес сервера.
3. Перенесите файлы, которые необходимо скопировать, используя графический интерфейс.

Глава 9

ШАГ 5. Установка FindFace Lite

9.1 Установка

1. Измените настройки *файла инсталлятора*, отметив его как “исполняемый”. Используйте команду ниже:

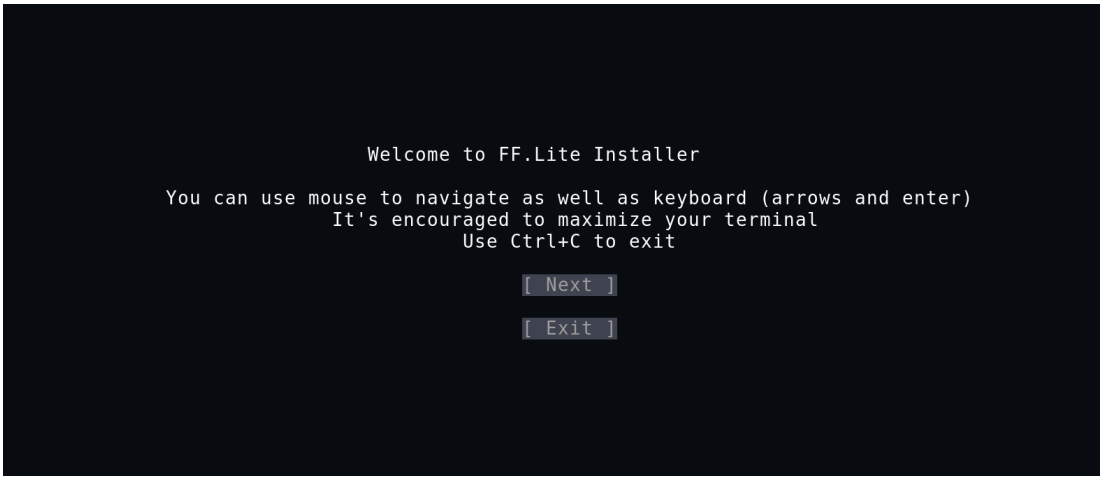
```
chmod +x fflite-{cpu|gpu}-master-g{git_hash}.run
```

Где `fflite-{cpu|gpu}-master-g{git_hash}.run` — название файла инсталлятора.

2. Запустите файл:

```
sudo  ./fflite-{cpu|gpu}-master-g{git_hash}.run
```

3. Откроется интерфейс инсталлятора внутри командной строки. Нажмите **[Next]**.

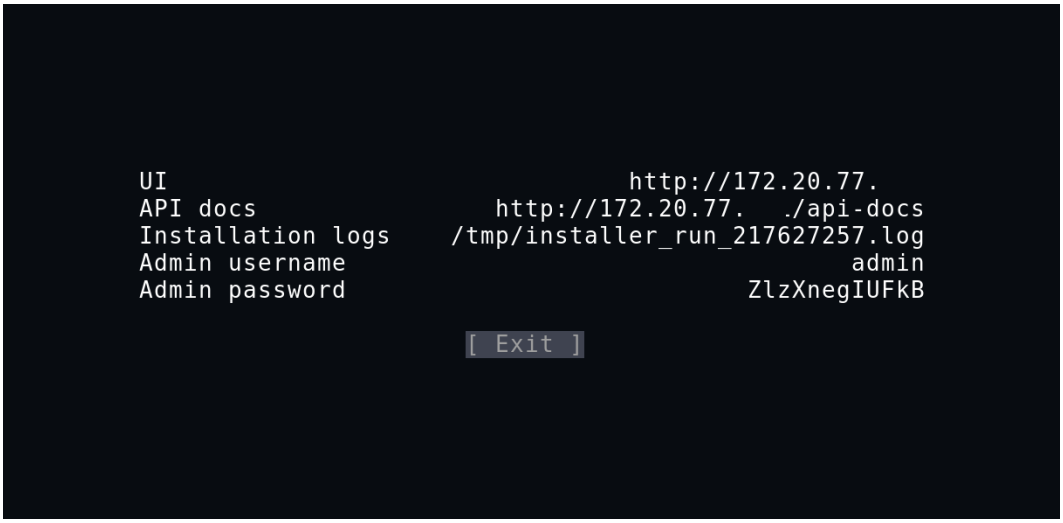


4. Дождитесь окончания проверки ПО. Нажмите **[Next]**.

5. После успешной проверки начнется установка компонентов. Дождитесь окончания установки и нажмите **Enter**.

6. Отобразится информация для авторизации и работы с сервисом.

Предупреждение: Обязательно сохраните отображенную информацию.



7. Установка FindFace Lite завершена.

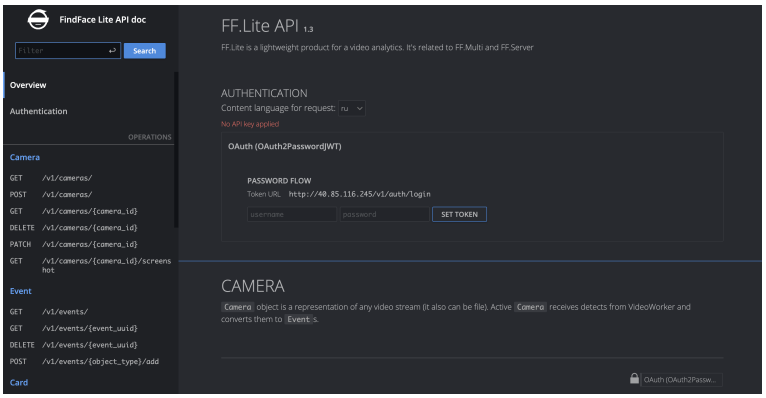
   Нажмите **[Exit]**, чтобы выйти из инсталлятора.

   В терминале отобразится путь до файла с логом установки FindFace Lite, в нем можно найти данные для авторизации в сервис, если вы забыли сохранить их на предыдущем шаге.

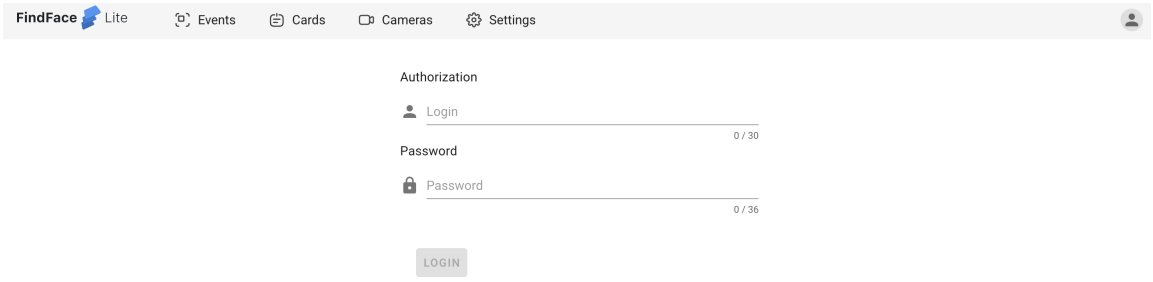
9.2 После установки

Начните знакомство с FindFace Lite:

- API документация расположена по адресу `http://<your_hostname>/api-docs`. Изучить описание запросов и того, как пользоваться интерактивной API документацией можно *в статье*.



- Интерфейс FindFace Lite расположен по адресу `http://<your_hostname>`.



Глава 10

Настройка конфигурации FindFace Lite

Вы можете внести персонализированные настройки в FindFace Lite в конфигурационном файле сервиса, расположенном в **FFlite -> api_config.yml**.

Конфигурационный файл содержит следующие блоки информации:

- **app** — конфигурация API;
- **eapi** — адрес сервиса eapi;
- **eapi_license_plate** — адрес сервиса eapi_license_plate;
- **vm** — адрес и данные для подключения к виртуальной машине;
- **db** — адрес и данные для подключения к базе данных.

10.1 Настройка конфигурации

Откройте его, используя текстовый редактор (например: *nano*, *vim* и тд.) и внесите изменения в необходимые настройки.

Ниже расположена информация о каждом блоке настроек с описанием возможных параметров и их значений.

Предупреждение: Пожалуйста, изучите описание настроек конфигурационного файл, чтобы корректно вносить изменения.

Примените новые параметры, перезагрузив `сервис api`. Для этого используйте команду:

```
docker compose restart api
```

10.2 Настройки блока App

10.2.1 Возможные значения

Настройки	Возможные значения
host:	**0.0.0.0** — значение по умолчанию.
port:	**8000** — значение по умолчанию.
debug:	**false** (по умолчанию) — режим отладки выключен;
	true — режим отладки включен.
router_base_url:	**http://nginx** — значение по умолчанию.
media_root:	**/uploads** — значение по умолчанию.
fullframe_root	**/fullframe** – значение по умолчанию.
normalized_root:	**/normalized** – значение по умолчанию.
save_fullframe:	**false** – полноразмерные изображения не будут сохраняться.
	true (по умолчанию) — полноразмерные изображения будут сохраняться.
save_normalized:	**false** (по умолчанию) – преобразованные изображения не будут сохраняться.
	true— преобразованные изображения будут сохраняться.
secret_key:	**change_me** — значение по умолчанию.
max_event_age_days:	**20** – значение по умолчанию.
face_confidence_threshold:	**0.714** – значение по умолчанию.
car_confidence_threshold:	**0.65** — значение по умолчанию.
webhook_workers_num:	**10** — значение по умолчанию.
exit_on_availability_check_fail:	**false** – API сервис предпринимает попытки получить необходимые ресурсы д
	true (по умолчанию) — API сервис прекратит работу, если один из необходи
event_creation_token:	**change_me** — значение по умолчанию.
event_creation_response_type:	**serialized** (по умолчанию) – ответ будет включать информацию о созданном
	id – ответ будет включать ID созданного Event`a.
	serialized_verbose — ответ будет включать информацию о созданном Event
face_features:	**headpose** – положение головы;
	medmask – распознавание медицинской маски;
	liveness – применение технологии “лайвнес”, при которой происходит оценка
car_features	**orientation** — распознавание положения автомобиля.
	special_types – распознавание типа автомобиля.
	license_plate_visibility —распознавание номерного знака автомобиля.
liveness_source	**eapi** — использование сервиса “eapi” для получения данных о ”живости объе
	vw – использование сервиса “videoworker” для получения данных о ”живости
auth_enabled:	**false** — авторизация выключена;
	true (по умолчанию) – авторизация включена.
access_token_expire_minutes:	**43200** — значение по умолчанию.
dedup_enabled:	**false** – дублирование Event’ов выключено;
	true (по умолчанию) – дублирование Event’ов включено.
save_dedup_events:	**false** (по умолчанию) – дубликаты Event’ов будут сохранены.
	true— дубликаты ивентов не будут сохранятся.
face_dedup_interval:	**5** – значение по умолчанию.
face_dedup_confidence:	**0.9** — значение по умолчанию.
car_dedup_interval:	**5** – значение по умолчанию.
car_dedup_confidence:	**0.9** — значение по умолчанию.

10.2.2 Описание настроек

Настройки	Описание
host:	Информация о хосте
port:	Информация о порте
debug:	Управление режимом отладки. На момент версии 1.3 доступно управление **только журналом отладки**.
router_base_url:	Базовый URL роутера. Пожалуйста, изменяйте, только если уверены в результате.
media_root:	Указание корневой директории для хранения изображений объектов **Objects**.
fullframe_root:	Указание корневой директории для полноразмерных изображений, полученных из обработчика видеопотоков VideoWorker (vw).
normalized_root:	Корневая директория для преобразованных изображений, где они хранятся для миграций между моделями в базе данных.
save_fullframe:	Настройки сохранения полноразмерных изображений.
save_normalized:	Настройки сохранения преобразованных изображений.
secret_key:	Секретный ключ, необходимый для выполнения операций, требующих проверки безопасности.
max_event_age_days:	Максимальное время хранения Event'ов. После окончания указанного в настройке времени Event будет удален.
face_confidence_threshold:	Значение порога уверенности, согласно которому Event совпадает или не совпадает с Card при распознавании и обработке изображений людей. Если значение порога уверенности превышает установленное значение, Event соотносится Card.
car_confidence_threshold:	Значение порога уверенности, согласно которому Event совпадает или не совпадает с Card при распознавании и обработке изображений автомобилей. Если значение порога уверенности превышает установленное значение, Event соотносится с Card.
webhook_workers_num:	Количество одновременных обработчиков вебхуков, отправляющих запросы.
exit_on_availability_check_fail:	Прекращение работы сервиса в случае недоступности необходимых ресурсов.
event_creation_token:	Токен, используемый для аутентификации внешнего детектора для создания Event`а (/{object_type}/add). Токен JWT не используется в этом запросе.
event_creation_response_type:	Параметр содержания ответа на запрос на о создании Event`а (/{object_type}/add).
face_features:	Управление функциональностью FindFace Lite для распознавания лиц. Функциональность активна, если она указана в значении. После активации функциональности недавно созданные Event`ы будут обработаны согласно новой настройке.
car_features	Управление функциональностью FindFace Lite для распознавания автомобилей. Функциональность активна, если она указана в значении. После активации функциональности недавно созданные Event`ы будут обработаны согласно новой настройке.
liveness_source	Источник данных для применения технологии "лайвнесс".
auth_enabled:	Управление авторизацией. Обратите внимание, что все API запросы (с некоторыми исключениями) требуют заголовок *Authorization* с указанием токена авторизации *JWT <token>*.
access_token_expire_minutes:	Время жизни токена авторизации.
dedup_enabled:	Управление дублированием Event'ов.
save_dedup_events:	Настройки сохранения дублирующихся Event'ов.
face_dedup_interval:	Временной интервал в секундах, в течение которого несколько Event'ов с одним человеком отмечаются как дубликаты.
face_dedup_confidence:	Порог совпадения между двумя дублирующими Event'ами.
car_dedup_interval:	Временной интервал в секундах, в течение которого несколько Event'ов с одной машиной отмечаются как дубликаты.
car_dedup_confidence:	Порог совпадения между двумя дублирующими Event'ами.

10.3 Настройки блока EAPI

Настройки	Возможные значения	Описание
host:	**eapi** — значение по умолчанию.	Информация о хосте
port:	**18666** — значение по умолчанию.	Информация о порте

10.4 Настройки блока License plate EAPI

Настройки	Возможные значения	Описание
host:	**eapi-license-plate** — значение по умолчанию.	Информация о хосте
port:	**18667** — значение по умолчанию.	Информация о порте

10.5 Настройки блока VM

Настройки	Возможные значения	Описание
host:	**vm** — значение по умолчанию.	Информация о хосте
port:	**18810** – значение по умолчанию.	Информация о порте
token:	**GOOD_TOKEN** — значение по умолчанию.	Токен для процессов, связанных с виртуальной машиной. Должен совпадать с токеном из конфигурационного файла *vm.conf*.

10.6 Настройки блока DB

Настройки	Возможные значения	Описание
host:	**postgres** — значение по умолчанию.	Информация о хосте
port:	**5432** – значение по умолчанию.	Информация о порте
user:	**fflite** — значение по умолчанию.	Данные для доступа к базе данных с названием из настройки **database**.
password:	**fflite** — значение по умолчанию.	
database:	**fflite** — значение по умолчанию.	Название базы данных.

Глава 11

API

FindFace Lite API находится на странице `http:// <your_hostname>/api-docs`. Страница интерактивна, вы можете делать запросы и получать ответы прямо на ней.

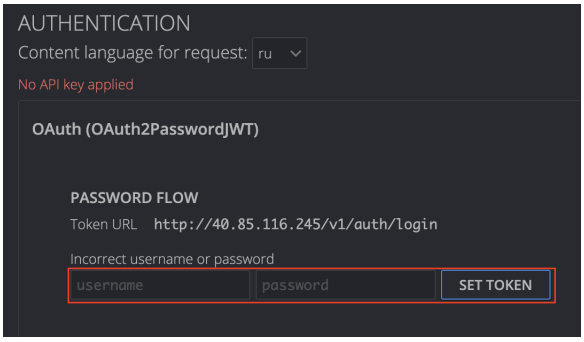
API документация позволяет читать, создавать, обновлять и удалять любые сущности, а также содержит описание всех методов и параметров.

В этой статье описаны функциональность FindFace Lite API и то, как использовать интерактивную API документацию.

11.1 Подготовка к использованию API

Перед использованием FindFace Lite API выполните авторизацию, создав JWT-токен в разделе **AUTHENTICATION**.

Введите **username** и **password** из *ШАГА 5* блока **С чего начать** и нажмите кнопку **SET TOKEN**.

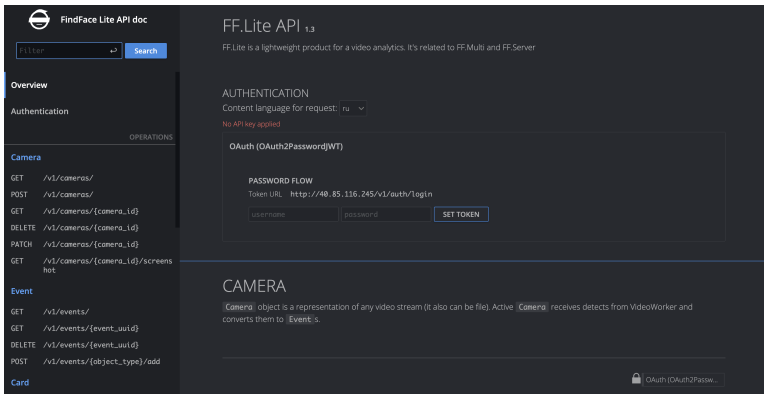


После аутентификации можно начать использование интерактивной FindFace Lite API.

Примечание: Для применения API запросов вне интерактивной API документации используйте сгенерированный токен.

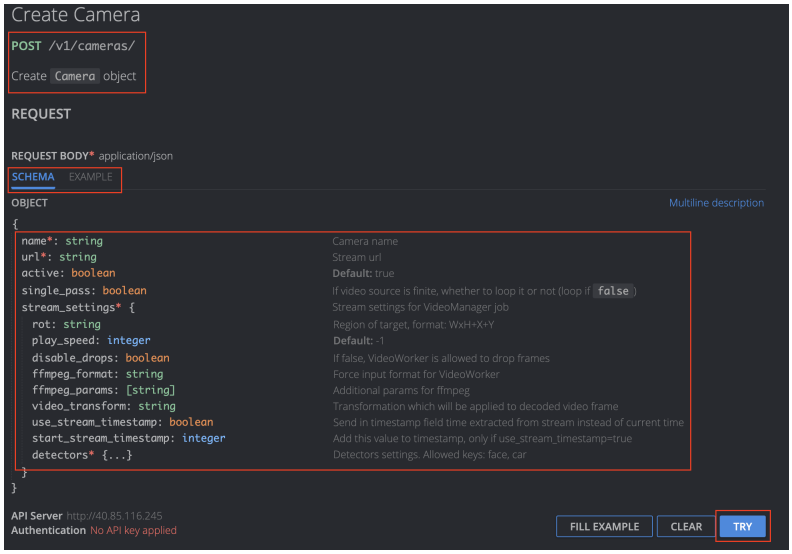
11.2 Использование API

Страница с API разделена на 2 части: слева список операций, справа поля для выполнения операции.



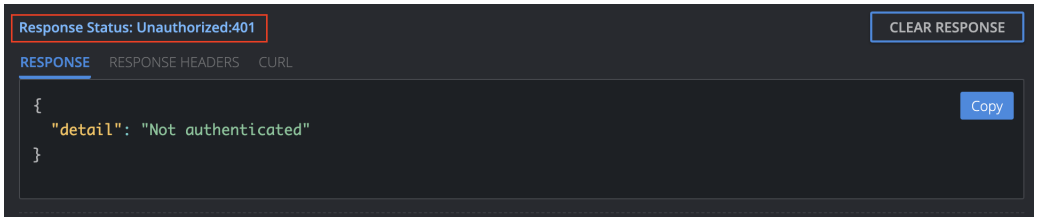
Каждая операция содержит блок **Request** и блок **Response**:

- Блок **Request** описывает операции, включая **request schema** — структуру запроса с интерпретацией каждого параметра, **request example** — пример запроса и кнопку **TRY** для отправки запроса.



- Блок **Response** описывает **schema** — структуру ответа и **examples** — примеры каждого варианта ответа для используемой операции.

После нажатия кнопки **TRY** запрос отправится, отобразится блок ответа с **response status** — статусом запроса и подробной информацией о нем.



11.3 Функциональность FindFace Lite API

API документацию можно разделить на смысловые блоки, которые содержат все доступные операции по управлению функциональностью FindFace Lite: получение, изменение и удаление сущностей.

1. **Процесс распознавания**
    - Операции раздела **Camera** управляют объектом **Camera**, который является образом видеопотока или файла, передающего видео. Активный объект Camera получает информацию из обработчика видеопотоков *VideoWorker* и конвертирует ее в Event'ы.
    - Операции раздела **Event** управляют объектом **Event**, который является результатом фиксации лица или автомобиля, попадающего в кадр камеры. Активный объект Camera получает информацию из обработчика видеопотоков VideoWorker и конвертирует ее в Event'ы. Вы или сторонняя система также можете создавать их вне автоматического процесса, используя POST-запрос.
    - Операции раздела **Card** управляют объектом **Card**, который является профилем реального человека или автомобиля. Card может быть двух типов *face* или *car*.
    - Операции раздела **Object** управляют объектом **Object**, который соответствует реальному лицу или автомобилю. Чтобы создать объект необходимо добавить соответствующие изображение и ссылку на Card, к которому необходимо привязать Object.

2. **Взаимодействие с внешними системами**
   - Операции раздела **Webhook** управляют отправкой уведомлений о совпадении по Event во внешние системы.
3. **Аутентификация и настройки пользователя**
   - Операции раздела **Auth** описывают методы для аутентификации.
   - Операции раздела **User** управляют пользователями FindFace Lite
4. **Системные операции**
   - Раздел **Misc** содержат прочие запросы, связанные с работой сервиса.
   - Раздел **Pipline** – это внутренний метод для обработчика видеопотоков *VideoWorker* и обычно не используется в общем процессе.

Глава 12

Периферийные устройства

12.1 Что такое периферийные устройства

Периферийные устройства — это физические устройства (например, *терминалы контроля доступа*), которые могут отправлять изображения в FindFace Lite для распознавания объектов. Для того, чтобы эти устройства получили результаты распознавания, необходимо настроить *вебхуки*.

Процесс распознавания с периферийных устройств такой же, как и с камер видеонаблюдения. Разница заключается в том, что Event'ы создаются непосредственно с периферийного устройства при помощи POST-запроса, а не из системы FindFace Lite.

12.2 Подготовка к процессу распознавания

Перед интеграцией периферийного устройства пройдите аутентификацию и укажите *Object'ы* и *Card'ы* , которые будут использоваться для распознавания данных из созданных *Event'ов* .

12.2.1 Пройдите аутентификацию

Для выполнения всех операций (кроме тех, которые связаны с Event'ами) авторизуйтесь в системе, используя POST-запрос **/v1/auth/login**. Для работы с Event`ами используется токен, указанный в *конфигурационном файле*.

Для параметров **username** и **password** используйте данные, полученные по завершению на *ШАГА 5*.

Пример запроса:

```
{
  "username": "login",
  "password": "password"
}
```

Пример успешного ответа:

```
{
   "access_token": "token"
}
```

12.2.2 Создайте объект Card

Card используется для хранения нескольких Object'ов под единым профилем человека или автомобиля. В процессе распознавания Card рассматривается как *результат распознавания*.

Чтобы создать Card, используйте POST-запрос **/v1/cards/**. Все параметры описаны ниже.

Параметры	Типы значений	Описание
name	строка	Название Card.
active	true	Объект Card выключен.
	false	Объект Card включен.
type	face	Объект Card создан для хранения информации о лице.
	car	Объект Card создан для хранения информации об автомобиле.
wiegand	строка	Код интерфейса Wiegand в системе контроля доступа.

Пример запроса:

```
{
  "name": "test card",
  "active": true,
  "type": "face",
  "wiegand": "test wiegand code"
}
```

Пример успешного ответа:

```
{
  "name": "test card2",
  "active": true,
  "type": "face",
  "wiegand": "test wiegand cod2e",
  "id": 2,
  "objects": []
}
```

12.2.3 Создайте объект Object

Objects соответствует реальному лицу или автомобилю. Чтобы создать объект необходимо добавить соответствующие изображение и ссылку на Card, используя POST-запрос **/v1/objects/**.

Все запросы описаны ниже.

Параметры	Типы значений	Описание
card_id	число	Идентификатор объекта Card, с которым необходимо связать Object. К одной Card может быть привязано несколько Object'ов.
type	enum	Тип объекта, который вы хотите создать. **Car** — изображение автомобиля, **face** — изображение лица, **license_plate** — изображение номерного знака автомобиля.
input_file	строка	Укажите название файла, который содержит лицо человека или изображение автомобиля, которые вы хотите добавить в качестве Object'ов в систему.

Пример запроса:

```
{
  "card_id": "2",
  "type": "face",
  "input_file": "somehash.jpg"
}
```

Пример успешного ответа:

```
{
  "id": 1,
  "emben": "vV1yPfc2izy...de8vY/bvNXLfDw=",
  "type": "face",
  "card": 4,
  "filename": "somehash.jpg"
}
```

12.3 Интеграция периферийного устройства

Чтобы интегрировать периферийное устройство, используйте API. Все необходимые шаги и связанные операции описаны ниже.

12.3.1 Создайте объект Camera

Создайте объект *Camera*, используя POST-запрос `/v1/cameras/`.

Примечание: Объект Camera не используется в процессе распознавания при помощи периферийных устройств. Он необходим только для создания объектов Event.

Ниже приведено описание параметров, необходимых для создания объекта Camera. Параметры, которые не включены в таблицу, описывают настройки для обработки видеопотоков. Вы можете не обращать внимание на них, т.к. при использовании периферийных устройств в FindFace Lite отправляются изображения, а не видеопотоки.

Параметры	Описание
name	Название объекта Camera. Выберите любое.
url:	URL видеопотока. У вас нет видеопотока, поэтому укажите любое значение, начинающееся с `rtmp://`.
active	Статус активности объекта Camera. Установите значение “выключено”, для этого укажите `false`.
stream_settings:	Настройки для видеопотоков с камеры видеонаблюдения. Необходимо заполнить только обязательный параметр — **detectors**.
detectors	Настройки распознавания. Необходимо включить параметр в запрос, но необязательно его заполнять. Можно оставить параметр пустым.

Пример запроса:

```
{
  "name": "Edge device",
  "url": "rtmp://none",
  "active": false,
  "stream_settings": {
      "detectors": {
      }
  }
}
```

Пример успешного ответа:

```
{
  "name": "Edge device",
  "url": "rtmp://none",
  "active": false,
  "single_pass": false,
  "stream_settings": {
      "rot": "",
      "play_speed": -1,
      "disable_drops": false,
      "ffmpeg_format": "",
      "ffmpeg_params": [],
      "video_transform": "",
      "use_stream_timestamp": false,
      "start_stream_timestamp": 0,
      "detectors": {}
  },
  "id": 2,
  "status": "UNKNOWN"
}
```

12.3.2 Настройте периферийное устройство

Предупреждение: После выполнения настроек из этой инструкции периферийное устройство будет готово для отправки данных в FindFace Lite на распознавание. FindFace Lite также будет подготовлен для распознавания. Чтобы настроить отправку результатов распознавания из системы в периферийное устройство, настройте *вебхук*.

Настройте отправку POST-запросов `/v1/events/{object_type}/add` периферийным устройством в FindFaceLite. В случае с периферийными устройствами, Event — это изображение зафиксированного лица или автомобиля, попавших в зону периферийного устройства.

Ниже описаны параметры, которые периферийное устройство должно отправлять в POST-запросе к FindFace Lite API.

Параметры	Типы значений	Описание
object_type	face	Параметр пути запроса */v1/events/{object_type}/add*, который указывает на объект распознавания.
	license_plate	
token	строка	Авторизация при помощи токена, указанного в параметре **event_creation_token** *конфигурационного файла*.
camera	число	Идентификатор объекта Camera, с которым связан созданный Event.
fullframe	бинарные данные	Изображение в любом из подходящих форматов (jpeg, png, etc.).
rotate	true	Технология автоматического поворота изображения для подготовки к распознаванию включена. Система пытается развернуть изображение, если объект на нем занимает перевернутое положение.
	false	Технология автоматического поворота изображения для подготовки к распознаванию выключена.
timestamp	дата-время	Дата и время в формате **yyyy-MM-dd'T'HH:mm:ss. SSSXXX**, например: **2000-10-31T01:30:00.000-05:00**.
mf_selector	all	Параметр фиксации множества объектов включен. Распознаются все объекты на изображении.
	biggest	Параметр фиксации множества объектов выключен. Распознается только самый большой объект на изображении.
roi	числа	Аббревиатура **roi** используется для обозначения **области распознавания**. Укажите значения в формате [слева, сверху, справа, снизу], где значения в скобках — это числа в градусах. Указанными значениями формируется область распознавания.

Пример запроса:

```
{
    "object_type": "face",
    "token": "change_me",
    "camera": 2,
    "fullframe": "somehash.jpg",
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    "rotate": true,
    "timestamp": "2000-10-31T01:30:00.000-05:00",
    "mf_selector": "biggest",
    "roi": 15,20,12,14
}
```

Пример успешного ответа:

Ответ будет отличаться в зависимости от значений, установленных для параметра `event_creation_response_type` в *конфигурационном файле*.

- Отображается только ID Event'а.

```
{
  "events": [
    "cc04cc9c-f355-4121-80c4-94a02eec652a",
    "c7d51db3-5b52-4318-9565-e2651308c1a6"
  ]
}
```

- Отображается полная информация о созданном Event'е, включая информацию о совпавшем Card, путь до полноразмерного изображения и др.

```
{
  "events": [
    {
      "bbox_bottom": 97,
      "bbox_left": 170,
      "bbox_right": 214,
      "bbox_top": 39,
      "bs_type": "realtime",
      "camera": 1,
      "card": null,
      "confidence": null,
      "created_date": "2022-12-29 13:02:07.910724+00:00",
      "emben":
↪"bmY3Pff9Grt1Ah09lp8kvn+a6Tw8SZs8K5xtvLOjtrxFtJ+9d5WIPH3PHL39acg9oNWhu4Mv2j2VjPo8QqDjubiFkz05Bou9
↪aQBvfdqFD2/
↪woc9j03iO5U3vT1P6ya9BfNyPUCBkz1Smmm8CIPvPRPxWTzWXxo8DwRGvMxfp7zRhGw8KyZzPtwoCTOBx7C9AcKWvflgUb2NL
↪BD7OU8M9grEbPVTZCb3mgXg+LxfEvdm6uL2wLhO8BU6yPQhREzOkj1M98tY+PNbA9D1MDCO7Tp4dPjh7n7zZAQS9/
↪JFzPWEJCb1CTwU+3deFvQb/hz2YaAa+QbjpvdOUFb7HVtG9NEhLPMfS1rouTk49f6DTOy9r/
↪j373aG7hBa3PW/
↪eJT5Zuz08cO+2PZXVDL3hemE8sYaOPY7XtjO6NAG+Stw4vrhQhb2KtJe9J8hCvTM/
↪Gr4vrOm9S3jJvfnBHz2QVgk97HuzvH2V/
↪71lQsk9l4UTvqE3XD1ssos9ErvcvY5CCr6ftJ+9EOarPOKvKrOJFa69sdyCvQPcuj2nLtO89I+iuzS+aj1bTgY+qT4dPtREVz
↪rVPA==",
      "features": {
        "headpose": {
          "pitch": 11.207756,
          "roll": 3.1429868,
          "yaw": 4.652887
        },
        "liveness": 0.6584981
```

(continues on next page)

(продолжение с предыдущей страницы)


```
      },
      "fullframe": "2022/12/29/13/c436c2d92c4c627d5c6d13f9f1d9555a.jpg",
      "type": "face",
      "uuid": "2f25dd19-d0cd-4b44-9147-69a7dc57450e"
    },
    {
      "bbox_bottom": 75,
      "bbox_left": 57,
      "bbox_right": 100,
      "bbox_top": 15,
      "bs_type": "realtime",
      "camera": 1,
      "card": 1,
      "confidence": 0.8306973576545715,
      "created_date": "2022-12-29 13:02:07.917896+00:00",
      "emben": "yRukvSOo1r2Bcm699SGTO/
↪SPBz64MvK9xINqPYDge708nSW9ba4BvAVgLz77ctw7A2OhvZ5LRD2rA1488DLnvSKQXT0ER367zf20vdypqb2Lhog8nIxjPa7
↪wKlvVlwr724Ll09LdqYPeOq170bDua8Zn2lPNy8dr0TF9G7VhMXPT6yQz0aCI49h8OJPQVTKb25oB+9x9++PEKCFj4uq0i8bE
↪PR3mrbxs+rW7TQqNO4QcFz2oI407H4nfvY/
↪nQD57Y2m9ItFMPZQKibzobRi6cf9wPT1itz0lkw89qUv0vS9RVDxjGoC9E3SiOxqSsbzjnyc+P4ZnPpFjEz5XMZE8IuILvYvg
↪A9waicOx9On7z0kW+9k7EmvIxuuLwdPPo9t5H3vQETLz6FcGQ+1fOqvfkwKz0rfSO9ckoivV65k70xw5296raIvYnnE70gaYa
↪",
      "features": {
        "headpose": {
          "pitch": -2.1808214,
          "roll": -0.5856089,
          "yaw": -4.5041146
        },
        "liveness": 0.5574532
      },
      "fullframe": "2022/12/29/13/4ef96c620d738d87c00aaaaa12fccca2.jpg",
      "type": "face",
      "uuid": "9b718d45-919a-490f-9fe6-b2af58cbf83a"
    }
  ]
}
```


- Отображается полная информация о созданном Event'е (включая информацию о совпавшем объекте Card, путь до полноразмерного изображения и др.), а также полная информация о Card.


```
{
  "events": [
    {
      "bbox_bottom": 97,
      "bbox_left": 170,
      "bbox_right": 214,
      "bbox_top": 39,
      "bs_type": "realtime",
      "camera": {
        "active": false,
        "id": 1,
        "name": "test camera",
```


(continues on next page)

(продолжение с предыдущей страницы)


```
        "single_pass": false,
        "status": "DISABLED",
        "url": "rtmp://test"
      },
      "card": null,
      "confidence": null,
      "created_date": "2022-12-29 13:48:34.624541+00:00",
      "emben":
↪"bmY3Pff9Grt1Ah09lp8kvn+a6Tw8SZs8K5xtvLOjtrxFtJ+9d5WIPH3PHL39acg9oNWhu4Mv2j2VjPo8QqDjubiFkz05Bou9
↪aQBvfdqFD2/
↪woc9j03iO5U3vT1P6ya9BfNyPUCBkz1Smmm8CIPvPRPxWTzWXxo8DwRGvMxfp7zRhGw8KyZzPtwoCTOBx7C9AcKWvflgUb2NL
↪BD7OU8M9grEbPVTZCb3mgXg+LxfEvdm6uL2wLhO8BU6yPQhREz0kj1M98tY+PNbA9D1MDC07Tp4dPjh7n7zZAQS9/
↪JFzPWEJCb1CTwU+3deFvQb/hz2YaAa+QbjpvdOUFb7HVtG9NEhLPMfS1rouTk49f6DTOy9r/
↪j373aG7hBa3PW/
↪eJT5Zuz08cO+2PZXVDL3hemE8sYa0PY7Xtj06NAG+Stw4vrhQhb2KtJe9J8hCvTM/
↪Gr4vr0m9S3jJvfnBHz2QVgk97HuzvH2V/
↪71lQsk9l4UTvqE3XD1ssos9ErvcvY5CCr6ftJ+9EOarPOKvKr0JFa69sdyCvQPcuj2nLtO89I+iuzS+aj1bTgY+qT4dPtREVz
↪rVPA==",
      "features": {
        "headpose": {
          "pitch": 11.207756,
          "roll": 3.1429868,
          "yaw": 4.652887
        },
        "liveness": 0.6584981
      },
      "fullframe": "2022/12/29/13/5e870f4f9dbd1e27652f6384663b8cab.jpg",
      "type": "face",
      "uuid": "df0821b4-6e52-4b66-abd2-0f642e2a090a"
    },
    {
      "bbox_bottom": 75,
      "bbox_left": 57,
      "bbox_right": 100,
      "bbox_top": 15,
      "bs_type": "realtime",
      "camera": {
        "active": false,
        "id": 1,
        "name": "test camera",
        "single_pass": false,
        "status": "DISABLED",
        "url": "rtmp://test"
      },
      "card": {
        "active": true,
        "id": 1,
        "name": "test card",
        "objects": [],
        "type": "face",
        "wiegand": "test wiegand code"
      },
```


(continues on next page)

(продолжение с предыдущей страницы)


```
      "confidence": 0.8306973576545715,
      "created_date": "2022-12-29 13:48:34.633562+00:00",
      "emben": "yRukvSOo1r2Bcm699SGTO/
↪SPBz64MvK9xINqPYDge708nSW9ba4BvAVgLz77ctw7A2OhvZ5LRD2rA1488DLnvSKQXTOER367zf20vdypqb2Lhog8nIxjPa7
↪wKlvVlwr724Ll09LdqYPeOq170bDua8Zn2lPNy8dr0TF9G7VhMXPT6yQz0aCI49h8OJPQVTKb25oB+9x9++PEKCFj4uqOi8bE
↪PR3mrbxs+rW7TQqNO4QcFz2oI407H4nfvY/
↪nQD57Y2m9ItFMPZQKibzobRi6cf9wPT1itz0lkw89qUv0vS9RVDxjGoC9E3SiOxqSsbzjnyc+P4ZnPpFjEz5XMZE8IuILvYvg
↪A9waicOx9On7z0kW+9k7EmvIxuuLwdPPo9t5H3vQETLz6FcGQ+1fOqvfkwKz0rfSO9ckoivV65k70xw5296raIvYnnE70gaYa
↪",
      "features": {
        "headpose": {
          "pitch": -2.1808214,
          "roll": -0.5856089,
          "yaw": -4.5041146
        },
        "liveness": 0.5574532
      },
      "fullframe": "2022/12/29/13/6041c2a71f4e2020d4cbaa52ce9b41f8.jpg",
      "type": "face",
      "uuid": "4dad4c16-f1cd-4ff1-a18a-268b71c1dbec"
    }
  ]
}
```

Глава 13

Вебхуки

Вебхук — механизм отправки уведомлений при наступлении в системе события, на которое подписано клиентское приложение.

Вебхуки можно использовать для решения разных задач, например, для уведомления пользователя об определенном событии, вызова определенных действий на целевом веб-сайте, при решении задач безопасности, таких как удаленное автоматическое управление доступом и др.

Например, если вы настроили периферийное устройство и хотите получать результаты распознавания для верификации объекта.

Для того, чтобы FindFace Lite отправлял HTTP-запросы при наступлении определенных событий, создайте и настройте вебхук.

13.1 Пройдите аутентификацию

Чтобы пройти аутентификацию в системе, используйте POST-запрос **/v1/auth/login**.

Для параметров **username** и **password** используйте данные, полученные на *ШАГЕ 5*.

Пример запроса:

```
{
  "username": "login",
  "password": "password"
}
```

Пример успешного ответа:

```
{
  "access_token": "token"
}
```

13.2 Создайте Вебхук

Чтобы создать вебхук, используйте POST-запрос **/v1/webhooks/**. Ниже описаны все параметры и их значения:

Параметры	Типы значений	Описание
name	строка	Название вебхука.
active:	true	Значение по-умолчанию, которое означает, что вебх
	false	Возможное значение, которое означает, что вебхук
target:	строка	Целевой URL для отправки сообщение о событии.
filters	группа значений [filters]	Набор фильтров, основывая на которых сообщения
type_in	строка	Если созданный Event соответствует указанным ти
camera_in:	число или несколько чисел	Если созданный Event связан с указанными в значе
card_in:	число или несколько чисел	Если созданный Event связан с указанными в значе
confidence_gte:	число от 0 до 1	Если результат распознавания больше или равен у
confidence_lte:	число от 0 до 1	Если результат распознавания меньше или равно у
matched:	true	Сообщение на целевой URL будет отправлено тольк
	false	Сообщение на целевой URL будет отправлено тольк
bs_type_in:	overall	Сообщение на целевой URL будет отправлено тольк
	realtime	Сообщение на целевой URL будет отправлено тольк
Yaw, Pitch, Roll		Параметры означают угол наклона относительно о
yaw_lte:	число	Если положение головы по параметру **yaw** меньше
yaw_gte:	число	Если положение головы по параметру **yaw** больше
pitch_lte:	число	Если положение головы по параметру **pitch** меньш
pitch_gte:	число	Если положение головы по параметру **pitch** больш
roll_lte:	число	Если положение головы по параметру **roll** меньше
roll_gte:	число	Если положение головы по параметру **roll** больше
liveness_lte:	число от 0 до 1	Если результат проверки на “живость” меньше или
liveness_gte:	число от 0 до 1	Если результат проверки на “живость” больше или
medmask	object	Если в конфигурационном файле указана функцио
name	enum	Доступные параметры, по которым можно фильтро
		none — на лице нет медицинской маски;
		correct – медицинская маска надета корректно;
		incorrect – медицинская маска надета некорректно
confidence_lte:	число от 0 до 1	Если значение уверенности в правильности результ
confidence_gte:	число от 0 до 1	Если значение уверенности в правильности результ
orientation	object	Если в конфигурационном файле указана функцио
name	enum	Доступные параметры, по которым можно фильтро
		back – задняя часть автомобиля;
		side – боковая часть автомобиля;
		front – передняя часть автомобиля.
confidence_lte:	число от 0 до 1	Если значение уверенности в правильности результ
confidence_gte:	число от 0 до 1	Если значение уверенности в правильности результ
special_type	object	Если в конфигурационном файле указана функцио
name	enum	Доступные параметры, по которым можно фильтро
		not_special – неспециальное транспортное средств
		police – полиция,
		ambulance — скорая,
		road_service — дорожный спецтранспорт,

Параметры	Типы значений	Описание
		gas_service — газовая служба,
		rescue_service — МЧС,
		other_special — все прочие специальные транспор
		taxi — такси,
		route_transport — публичный транспорт,
		car_sharing — каршеринг,
		military — военные службы.
confidence_lte:	число от 0 до 1	Если значение уверенности в правильности результ
confidence_gte:	число от 0 до 1	Если значение уверенности в правильности результ
license_plate_number	строка	Если результат распознавания номера автомобиля
license_plate_visibility	object	**license_plate_visibility** — обязательная функция
name	enum	Доступные параметры, по которым можно фильтро
		partly_visible_no_text – номерной знак без тек
		fully_visible_no_text — номерной знак без текс
		invisible — номерной знак невидим,
		partly_visible — номерной знак с текстом и части
		fully_visible — номерной знак с текстом и полнос
confidence_lte:	число от 0 до 1	Если значение уверенности в правильности результ
confidence_gte:	число от 0 до 1	Если значение уверенности в правильности результ
license_plate_event_number	строка	Фильтр можно применить только к типу Event'а **li**
send_attempts:	число	Количество попыток отправки сообщения на целево

Пример запроса:

```
{
  "name": "test webhook",
  "active": true,
  "target": "http://localhost/webhok_test",
  "filters": {
    "camera_in": [
      1,
      2
    ],
    "card_in": [
      4,
      5
    ],
    "confidence_gte": 0.75,
    "confidence_lte": 0.79,
    "matched": true,
    "bs_type_in": [
      "overall",
      "realtime"
    ],
    "yaw_lte": 3.5,
    "yaw_gte": 3.5,
    "pitch_lte": -4.2,
    "pitch_gte": -4.2,
    "roll_lte": 1.8,
    "roll_gte": 1.8,
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    "liveness_lte": 0.44,
    "liveness_gte": 0.44
  },
  "send_attempts": 3
}
```

Пример успешного ответа:

```
{
  "name": "test webhook",
  "active": true,
  "target": "http://localhost/webhok_test",
  "filters": {
    "camera_in": [
      1,
      2
    ],
    "card_in": [
      4,
      5
    ],
    "confidence_gte": 0.75,
    "confidence_lte": 0.79,
    "matched": true,
    "bs_type_in": [
      "overall",
      "realtime"
    ],
    "yaw_lte": 3.5,
    "yaw_gte": 3.5,
    "pitch_lte": -4.2,
    "pitch_gte": -4.2,
    "roll_lte": 1.8,
    "roll_gte": 1.8,
    "liveness_lte": 0.44,
    "liveness_gte": 0.44
  },
  "send_attempts": 3,
  "id": 1
}
```

Глава 14

Интеграция с Sigur

14.1 Основное про интеграции с Sigur

FindFace Lite может использоваться как самостоятельный продукт, а также как дополнительный продукт, интегрированный со СКУД.

Интеграция FindFace Lite расширяет возможности используемого оборудования и систем контроля доступа.

14.2 Инструкция по интеграции с Sigur

Интеграция работает так, что при генерации события в FindFace Lite, совпадающего с карточкой, встроенный плагин отправляет запрос в Sigur на открытие турникета.

Предупреждение: В интеграции с Sigur предусмотрено только распознавание по лицу.

Чтобы настроить интеграцию с Sigur необходимо:

1. Установить и настроить FindFace Lite.
2. Выполнить настройку плагина интеграции Sigur в конфигурационном файле **ffpacs-config.yml**.
3. Настроить интеграцию в панели администратора Sigur.
4. Настроить вебхук для отправки Event'ов.

14.2.1 Установка FindFace Lite.

1. Чтобы корректно установить FindFace Lite, пройдите шаги установки, описанные в блоке *С чего начать*.
2. После установки вы можете внести персонализированные настройки в работу сервиса, например в:
   - Функциональность сервиса: настроить источник данных для применения технологии *Лайвнес* при распознавании лица.
   - Управление сервисом: настроить секретный ключ, необходимый для выполнения операций, требующих проверки безопасности или корневую директорию для полноразмерных изображений, полученных из обработчика видеопотоков VideoWorker (vw).

   Настройки расположены в конфигурационном файле сервиса **FFlite -> api_config.yml**.

   Подробное описание и рекомендации по внесению изменений читайте в статье *Настройка конфигурации*.

14.2.2 Настройка плагина интеграции Sigur в FindFace Lite

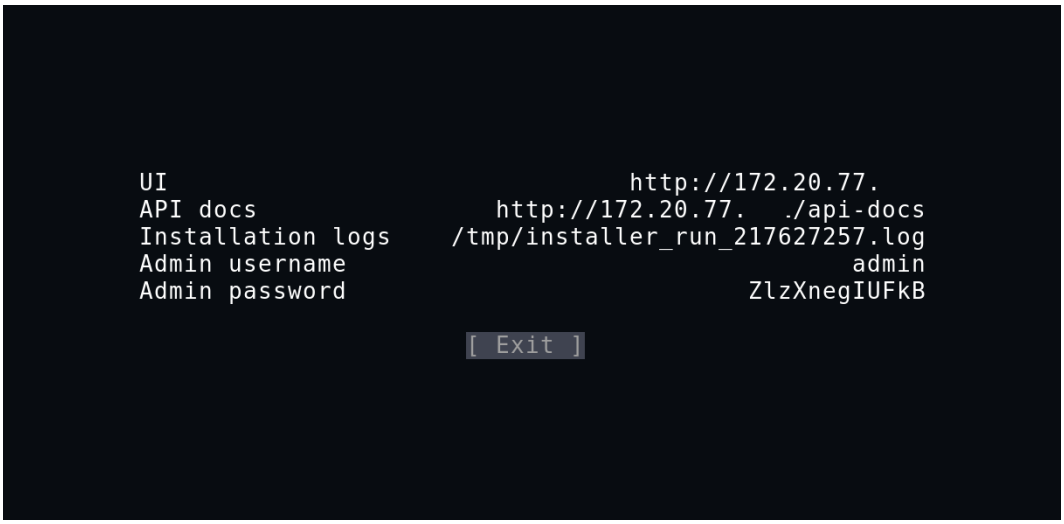
FindFace Lite разработан с учетом интеграций со СКУД.

Файл инсталлятора FindFace Lite содержит конфигурации для установки и настройки самой системы и специальный модуль, содержащий плагин Sigur, с помощью которого FindFace Lite интегрируется со СКУД.

Для настройки плагина:

1. Откройте файл конфигурации **FFlite -> ffpacs-config.yml**, используя текстовый редактор (например: nano, vim и тд.).
2. В открывшемся файле внесите изменения в поля, ответственные за авторизацию в FindFace Lite: **user** – имя пользователя и **password** – пароль.

   Логин и пароль от FindFace Lite был сформирован в конце установки продукта.



   Если вы забыли или не сохранили эти данные, на сервере с FindFace Lite:

   - Перейдите в папку **/tmp** из корневой директории, здесь хранятся логи.
   - Откройте файл логов установки сервиса, пример логов: **installer_run_080236265.log**.

- Ближе к концу файла будут указаны логин и пароль для созданных пользователей.

```
[azureuser@fflite-demo:/tmp$ ls -alh
total 196K
drwxrwxrwt 11 root      root      4.0K Mar 16 04:08 .
drwxr-xr-x 23 root      root      4.0K Mar  3 06:13 ..
drwxrwxrwt  2 root      root      4.0K Oct 25 08:53 .ICE-unix
drwxrwxrwt  2 root      root      4.0K Oct 25 08:53 .Test-unix
drwxrwxrwt  2 root      root      4.0K Oct 25 08:53 .X11-unix
drwxrwxrwt  2 root      root      4.0K Oct 25 08:53 .XIM-unix
drwxrwxrwt  2 root      root      4.0K Oct 25 08:53 .font-unix
drwx------  8 azureuser azureuser 4.0K Oct 31 14:16 installer970165156
-rw-------  1 root      root      9.5K Oct 27 13:05 installer_run_080236265.log
```

3. В подразделе плагина sigur, активируйте интеграцию, установив **true** в параметре **enable** блока **sigur**:

```
sigur:
  enable: true
```

4. После выполненных настроек, перезапустите модуль интегрции, запустив следующую команду:

```
docker compose restart ffpacs
```

14.2.3 Настройка вебхука

Создайте вебхук по *инструкции из статьи*.

Обязательные параметры:

Параметр	Значение
name	Любое имя, например, *fflite integration*
active	Активируйте вебхук, установив *true*
target	Укажите адрес и порт интеграционного модуля: *http://172.17.46.129:9998/detect*

Совет: Остальные параметры могут быть установлены по желанию, напрмиер, фильтры отправки вебхкука.

Пример вебхука:

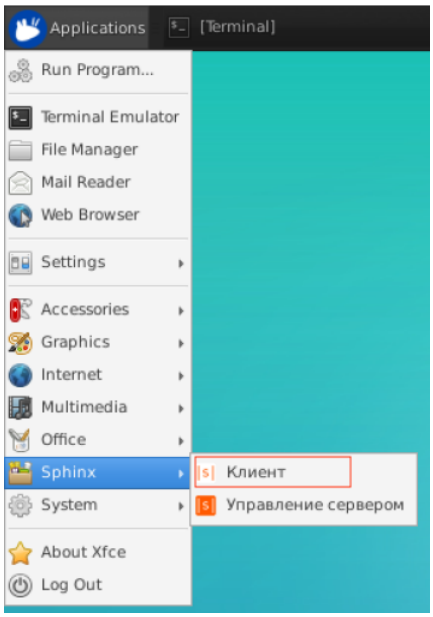
```
POST "http://172.17.46.129/v1/webhooks/"

{
   "name": "test webhook",
   "active": true,
   "target": "http://172.17.46.129:9998/detect", <- адрес, порт пакса
   "filters": {},
   "send_attempts": 1,
   "id": 1
}
```

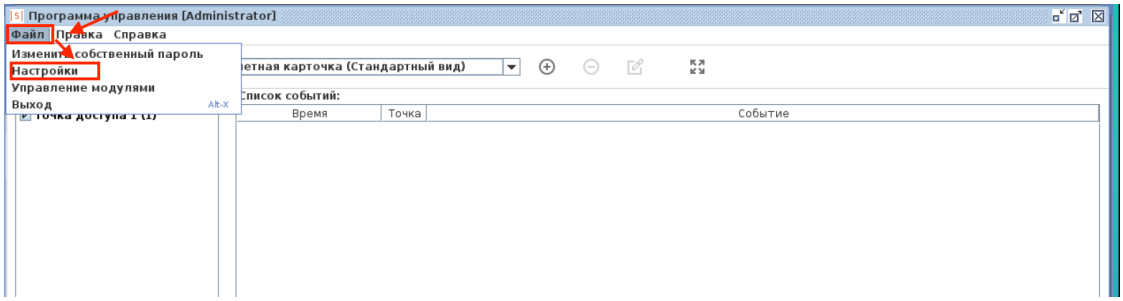
14.2.4 Настройка интеграции в Sigur

Последний шаг для настройки интеграции — добавлене данных о FindFace Lite в систему Sigur.

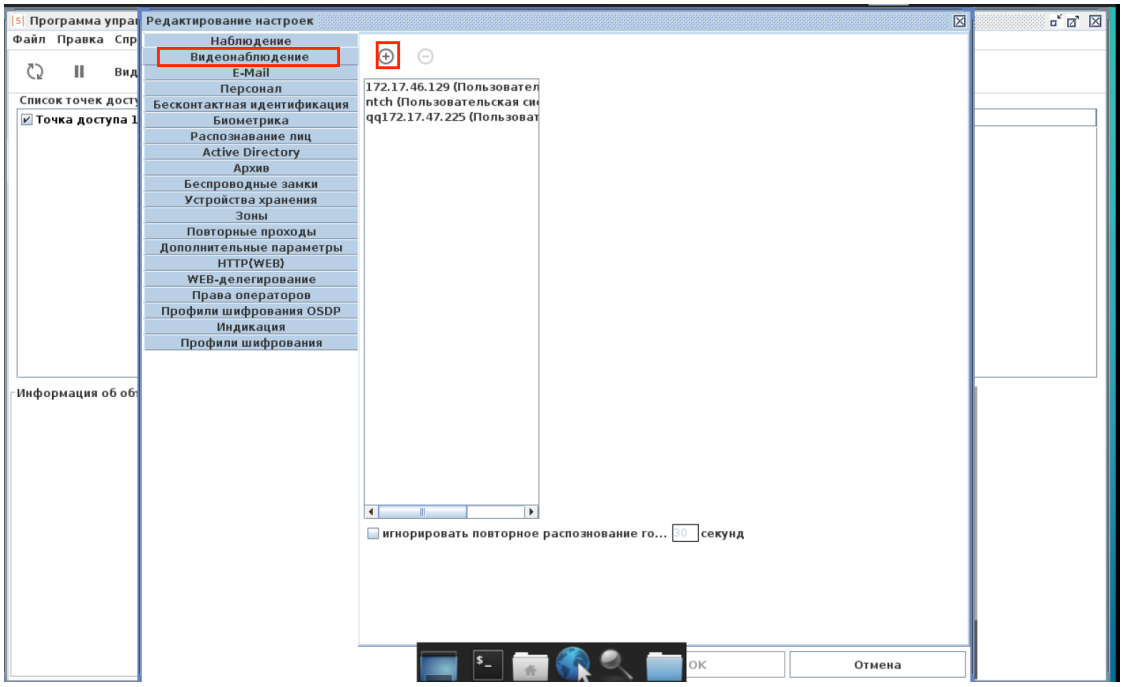
1. Откройте клиент Sigur.



2. Выберите **Файл** -> **Настройки**.

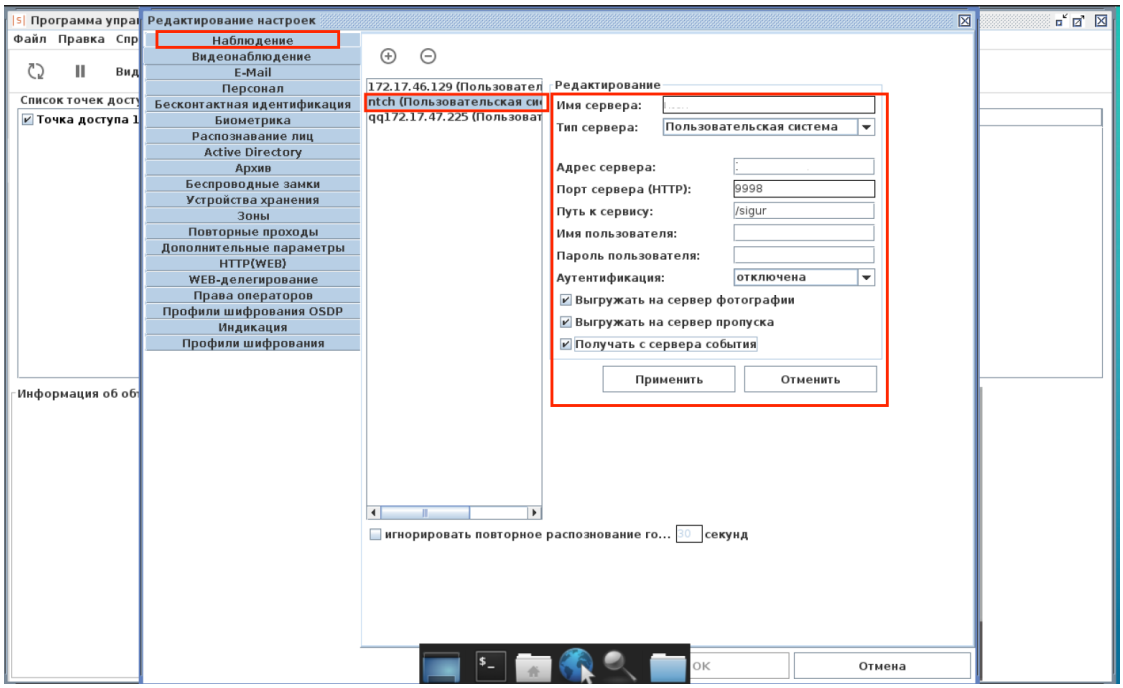


3. В разделе **Видеонаблюдение** добавьте новую пользовательскую систему, нажав +.



4. Введите данные конфигурации FindFace Lite:

Поле	Значение	Описание
Имя сервера	Любое имя	Название интегрируемой пользовательской системы
Тип сервера	*Пользовательская система*	Необходимы набор настроек для интеграции сервиса
Адрес сервера	URL	URL сервера, на которой установлен FindFace Lite
Порт сервера (HTTP)	*9998*	Порт сервера плагина
Путь к сервису	*/sigur*	Путь до сервиса внутри системы
Имя пользователя	Пусто	Данные о пользователе. Не используется
Пароль пользователя	Пусто	Данные о пользователе. Не используется
Аутентификация	*отключена*	Управление аутентификацией в системе
Выгружать на сервер фотографии	Активно	Порт сервера плагина
Выгружать на сервер пропуска	Активно	Путь до сервиса внутри системы
Получать с сервера события	Активно	Путь до сервиса внутри системы



5. Подключите камеры, добавленные в FindFace Lite к Sigur.

   В Sigur есть возможность разделения камер по направлению: «на вход» и «на выход».

   Для этого откройте **Оборудование** -> **Точка доступа** -> **Видеонаблюдение**:

   - В разделе **Камера «на выход»** добавьте камеру, которая будет фиксировать людей, покидающих место пребывания.
   - В разделе **Камера «на вход»** добавьте камеру, которая будет фиксировать людей, входящих на территорию людей.

   Настройки, которые необходимо внести:

Поле	Описание
Система	Выберите из списка пользовательскую систему, добавленную на предыдущем шаге.
Камера	Выберите из списка камеру, добавленную на стороне FindFace Lite.
Отступ	
Распознавание автомобильных номеров	Оставьте неактивным.
Разрешить верификацию по лицу	Активируйте.
Включить идентификацию по лицу	Активируйте

Настройка интеграции завершена.

Алфавитный указатель