ИНФРАСТРУКТУРА БИОМЕТРИЧЕСКОЙ ПЛАТФОРМЫ

ЕДИНАЯ БИОМЕТРИЧЕСКАЯ СИСТЕМА

Методические рекомендации

по подключению биометрических процессоров

к Единой биометрической системе

Версия 1.35

Москва 2025

История документа

Версия	Дата	Автор	Комментарии
1.0	05.12.2017	Шульженко С.Н.	Создание документа
1.1	19.06.2018	Курочкин В.С.	Корректировка разделов: −3 Основы взаимодействия; −4 Основные требования и рекомендации; −ПРИЛОЖЕНИЕ А; −ПРИЛОЖЕНИЕ Б.
1.2	19.07.2018	Сеначин С.В.	Корректировка разделов 4.1 – 4.3
1.3	30.07.2018	Сеначин С.В.	Корректировка раздела 4.3: изменены ограничения контейнера с ППО БО
1.4	22.11.2018	Сеначин С.В.	Добавлен раздел 4.4 «Требования по журналированию».
1.5	29.01.2019	Сеначин С.В.	Корректировка раздела 4.3: удалены ограничения контейнера с ППО БО по ресурсам (vCPU, vRAM, vHDD)
1.6	30.06.2020	Шишков А.А.	Корректировка Приложения А, п. «Метод «Извлечение биометрического шаблона»
1.7	21.09.2020	Шишков А.А.	Корректировка списка сокращений. В Приложение Б добавлены примеры text-instructions и passive-instructions. Корректировка описания.
1.8	18.01.2021	Самойлова Д.В.	Добавлено Приложение В «Описание интеграции БП биометрической идентификации с ЕБС».
1.9	16.03.2021	Самойлова Д.В.	Корректировка раздела «Точка доступа к API» в Приложениях
1.10	02.04.2021	Самойлова Д.В.	Корректировка разделов: 4.4. скорректированы требования по логированию; 4.5 изменена версия CentOS
1.11	23.06.2021	Самойлова Д.В.	Исправление ошибок в Приложении Б и В
1.12	28.07.2021	Самойлова Д.В.	Исправлены грамматические ошибки, добавлена сноска к ошибке в Приложении А
1.13	23.12.2021	Соловьёв Н.А.	Добавлено правило игнорирования параметров в методах add и update
1.14	15.02.2022	Шишков А.А.	Исправлены разделы 4.3-4.5
1.15		Шумский А.	Изменён раздел 4
1.16		Шумский А.	Изменён раздел 4
1.17		Шумский А.	Изменён раздел 4
1.18		Шумский А.	Изменён раздел 4
1.19	25.05.2022	Маркин М.С.	Исправлен раздел 4
1.19.3	9.06.2022	Маркин М.С.	Конфигурирование БП через переменные окружения
1.19.4	28.06.2022	Маркин М.С.	Раздел 4. Лицензии только офлайн
1.20	15.07.2022	Кубенский И.Р.	Добавлены новые коды ошибок к API БП
1.21	30.07.2022	Маркин М.С.	Раздел 4. Исправлены опечатки
1.22	03.08.2022	Шляхова О.Ю.	Актуализировано Приложение Б
1.23	05.08.2022	Маркин М.С.	Настройка *{pathPrefix}* через переменные

			окружения
1.24	30.09.2022	Шляхова О.Ю.	1 Актуализирован раздел 4 2 Актуализировано Приложение Б
1.25	12.12.2022	Попов Д.В.	Актуализировано Приложение Б
1.26	14.03.2023	Гаврилов С.Р.	Обновление терминов в соответствии с 572-ФЗ
1.27	05.06.2023	Ковальчук М.А.	Актуализировано Приложение Б
1.28	28.07.2023	Ковальчук М.А.	Актуализирован раздел 4 Актуализировано приложение
1.29	19.12.2023	Попов Д.В. Ковальчук М.А.	1 Обновлены Приложение А, Приложение Б 2 Удалено Приложение В 3 Актуализирован раздел 4
1.30	04.04.2023	Ковальчук М.А.	1 Добавлен новый код ошибки к методам API (приложение А) 2 Скорректировано сообщение при ошибке LDE-002004 3 Скорректирован раздел 4.8
1.31	16.07.2024	Липницкий И.В.	Актуализировано 1. Раздел 4 2. Приложение А 3. Приложение Б
1.32	03.12.2024	Исайченко В.В.	Добавлено Приложение В Актуализирован п.4.4, добавлено описание поставок Обновлено содержание
1.33	30.01.2025	Исайченко В.В.	Обновлен п. 4.4. Скорректированы описания методов в приложении А (поправлены маркированные списки помеченные комментариями).
1.34	19.03.2025	Исайченко В.В.	Обновлено описание базового урла Приложение В. Скорректировано описание пункта «Поддерживаемые в запросах методы HTTP» согласно комментариям
1.35	08.10.2025	Гаврилов С.Р.	Откорректированы: −п. 4.2 (добавлена поставка CPU-, GPU-версии), −п. 4.5 (4 сервиса), −п. 4.6 (поддержка Docker Engine 25.X, Docker Compose 2.24.1, добавлено требование к .env), −4.8 (контр. сумма образа), −п. 4.9 (требование инстр. по созданию индексов, −4.11 (абзац о способе сведений о лицензии), −4.12 (добавлен абзац о graceful shutdown). −Приложения А, В (дробные значения от 0.0 до 1.0 для compare и verify, поддержка постоянных соединений (keep-alive)), −Приложение Б (поддержка постоянных соединений keep-alive) Добавлен п. 4.13 (версионность).

Содержание

Список сокращений

Термин	Определение
Аутентификация	Действия по проверке подлинности субъекта доступа в автоматизированной информационной системе
Биометрическая верификация	Процесс подтверждения биометрического заявления при сравнении
Биометрическая идентификация	Процесс поиска по базе данных биометрических регистраций, направленный на поиск и возврат идентификатора(ов) биометрического контрольного шаблона, связанного с одним индивидом.
Биометрическая проба	Биометрический образец или набор биометрических признаков, введённый в алгоритм для использования в качестве объекта сравнения с биометрическим контрольным шаблоном (биометрическими контрольными шаблонами)
Биометрическая регистрация	Действия по созданию и сохранению записи данных биометрической регистрации в соответствии с правилами биометрической регистрации
Биометрическая система	Система, предназначенная для биометрического распознавания людей, основанного на их поведенческих и биологических характеристиках
Биометрическая характеристика	Биологические и поведенческие характеристики человека, которые могут быть зарегистрированы и использованы в качестве отличительных, повторяющихся биометрических признаков для распознавания людей.
Биометрические данные	Биометрический образец или совокупность биометрических образцов на любой стадии обработки, например, биометрический контрольный шаблон, биометрическая проба, биометрический признак или биометрическое свойство

Термин	Определение
Биометрический контрольный шаблон	Один или более хранимых биометрических шаблонов, относящихся к субъекту биометрических данных и используемых в качестве объекта сравнения
Биометрический образец	Аналоговое или цифровое представление биометрических характеристик, предшествующее извлечению биометрических признаков
Биометрический признак	Цифровое представление информации (числа или метки), извлечённое из биометрических образцов и используемое для сравнения
Биометрический процессор	Обработчик запросов на выполнение биометрических операций.
Биометрический шаблон	Набор хранимых биометрических признаков, сравниваемых непосредственно с биометрическими признаками биометрической пробы
Биометрическое заявление	Заявление, что субъект сбора биометрических данных является или не является собственно источником установленного или неустановленного биометрического контрольного шаблона
Биометрическое распознавание	Автоматическое распознавание индивидов, основанное на их поведенческих и биологических характеристиках. Биометрическое распознавание включает в себя биометрическую верификацию и биометрическую идентификацию
Витальность	Свойство или состояние живого субъекта, подтверждаемое анатомическими характеристиками, непроизвольными реакциями, физиологическими функциями, произвольными реакциями или поведенческими характеристиками субъекта
Единая биометрическая система (ЕБС)	Государственная информационная система «Единая система идентификации и аутентификации

Термин	Определение
	физических лиц с использованием биометрических персональных данных», которая содержит биометрические персональные данные физических лиц, векторы единой биометрической системы и иную предусмотренную в соответствии с частью 16 статьи 4 Федерального закона №572-ФЗ информацию, которая используется в целях осуществления идентификации, аутентификации с использованием биометрических персональных данных физических лиц, а также в иных правоотношениях в случаях, установленных законодательством Российской Федерации, и оператором которой является определенная Правительством Российской Федерации
Запись данных биометрической регистрации	Запись данных, связанная с субъектом биометрических данных, содержащая не биометрические данные, и связанная с идентификатором (идентификаторами) биометрического контрольного шаблона
Идентификатор биометрического контрольного шаблона	Указатель на запись данных биометрического контрольного шаблона в базе данных биометрических контрольных шаблонов
Мультимодальный режим	Режим работы биометрической системы, при котором процесс биометрического распознавания происходит одновременно по нескольким различным биометрическим характеристикам
Обнаружение витальности / Liveness detection	Измерение и анализ анатомических характеристик, непроизвольных и произвольных реакций субъекта с целью определения того, что биометрический образец получен от живого субъекта
Пользователь	Человек, взаимодействующий с биометрической системой с целью биометрической регистрации или удаленной идентификации его личности
Поставщик БДн (ЕБС)	Участник биометрического взаимодействия,

Термин	Определение
	имеющий право и осуществляющий регистрацию Пользователей в ЕБС.
Поставщик БП	Компания – производитель биометрических процессоров, поставляющая их под своей маркой
Потребитель БДн (ЕБС)	Участник биометрического взаимодействия, имеющий право и осуществляющий удаленную идентификацию Пользователей с использованием биометрического распознавания в ЕБС.
Сбор биометрических данных	Получение и запись в воспроизводимой форме сигнала биометрической характеристики (биометрических характеристик) непосредственно от человека, или от представления биометрической характеристики (биометрических характеристик)
Сравнение	Оценка, вычисление или измерение степени схожести и различия между биометрическим образцом и биометрическим контрольным шаблоном
Степень схожести	Количественный показатель, характеризующий схожесть извлеченных из биометрического образца признаков с биометрическим контрольным шаблоном.
Удаленная идентификация	Идентификация пользователей, в рамках требований Федерального закона от 07.08.2001 №115-ФЗ, осуществляемая по удалённым каналам связи, без визита пользователя в офис кредитной организации
Унимодальный режим	Режим работы биометрической системы, при котором процесс биометрического распознавания происходит по какой-либо одной биометрической характеристике (например, по записи голоса)
Участник БВ	Участник биометрического взаимодействия, имеющий право осуществлять биометрическую регистрацию и/или удаленную идентификацию в соответствии с ФЗ-115. А также поставщики БП

Термин	Определение
Web (Веб) приложение	Клиент-серверное приложение, в котором клиентом выступает интернет-браузер, а сервером — интернет-сервер
БДн	Биометрические данные
БКШ	Биометрический контрольный шаблон
БО	Биометрические образцы
БП	Биометрический процессор
ДКО	Дистанционные каналы обслуживания (WEB и мобильные приложения)
ЕБС, Система	Единая биометрическая система
КО	Кредитные организации
ОС	Операционная система
ППО	Прикладное программное обеспечение
ФГИС ЕСИА, ЕСИА	Федеральная государственная информационная система «Единая система идентификации и аутентификации в инфраструктуре, обеспечивающей информационно-технологическое взаимодействие информационных систем, используемых для предоставления государственных и муниципальных услуг в электронной форме», обеспечивающая санкционированный доступ к информации, содержащейся в информационных системах
ФГИС СМЭВ, СМЭВ	Федеральная государственная информационная Система Межведомственного Электронного Взаимодействия

1 ВВЕДЕНИЕ

1.1 Назначение документа

Требования, указанные в документе, следует рассматривать в дополнение к требованиям, содержащимся в нормативно-правовых документах, регламентирующих работу ЕБС.

В рамках документа рассматриваются следующие вопросы:

− подключение к ЕБС поставщиков БП биометрического распознавания;

− подключение к ЕБС поставщиков БП обнаружения витальности.

Описываемые в документе правила являются обязательными к применению участниками БВ ЕБС.

Для описания требований к участникам БВ используются следующие соглашения, выделенные жирным шрифтом:

− может – разрешено, но необязательно;

− не может – запрещено;

− должен – обязательно.

1.2 Нормативные ссылки

Данный документ разработан в целях реализации и во исполнение:

– Федерального закона от 07.08.2001 № 115-ФЗ «О противодействии легализации (отмыванию) доходов, полученных преступным путём, и финансированию терроризма»;

– Федерального закона от 07.07.2003 № 126-ФЗ «О связи».

2 ОБЩЕЕ ОПИСАНИЕ СИСТЕМЫ

Целью создания Системы является обеспечение возможности проведения удалённой идентификации пользователей по биометрическим характеристикам для исполнения требований, установленных федеральными законами от 07.08.2001 № 115-ФЗ «О противодействии легализации (отмыванию) доходов, полученных преступным путём, и финансированию терроризма» (далее – Федеральный закон № 115-ФЗ) и от 07.07.2003 № 126-ФЗ «О связи» (далее – Федеральный закон № 126-ФЗ).

Система обеспечивает уровень надёжности, необходимый для удалённой идентификации физических лиц КО – потребителями БДн, с дальнейшим оказанием им банковских услуг.

Система обеспечивает возможность решения следующих задач:

1. Сбор БДн как в офисах КО – поставщиков БДн, так и удалённо;

2. Передачу БО от КО - поставщиков БДн в ЕБС;

3. Хранение БО в Системе;

4. Формирование биометрических шаблонов из БО с приданием им статуса БКШ;

5. Хранение БКШ в Системе;

6. Получение БО от пользователей через ДКО КО -потребителей БДн;

7. Проверку полученных БО на соответствие требованиям к качеству и защиты от попыток фальсификации;

8. Сравнение БО с БКШ для проведения процедуры удаленной идентификации в ЕБС;

9. Взаимодействие ЕБС с ДКО КО - потребителей БДн;

10. Взаимодействие ЕБС с ЕСИА и СМЭВ.

Разработанная Система обеспечивает мультимодальный и унимодальный режимы работы. Список биометрических характеристик (модальностей), используемых для осуществления процесса удаленной идентификации состоит из следующих модальностей:

− модальность 1: аудиозапись голоса;

− модальность 2: фотоизображение лица.

Размер биометрических образцов может быть не более 20 Мбайт.

3 ОСНОВЫ ВЗАИМОДЕЙСТВИЯ

3.1 Описание процесса «Подключение БП биометрического распознавания»

При подключении к Системе нового БП биометрического распознавания запускается процедура создания из хранимых в Системе БО, прошедших проверку качества, новых биометрических шаблонов подключаемого БП биометрического распознавания. Полученные биометрические шаблоны сохраняются в соответствующих БКШ в дополнение к хранимым биометрическим шаблонам ранее подключённых к Системе БП биометрического распознавания.

3.2 Описание процесса «Подключение БП обнаружения витальности»

При подключении к Системе нового БП обнаружения витальности необходимо согласование предоставляемых БП описания и формата необходимых действий обнаружения витальности с возможностями клиентских приложений ЕБС.

4 ОСНОВНЫЕ ТРЕБОВАНИЯ И РЕКОМЕНДАЦИИ

4.1 Общая схема размещения и взаимодействия подключаемых БП биометрического распознавания и БП обнаружения витальности с ЕБС

БП размещаются в защищенном контуре ЕБС в кластере docker-swarm. Для обеспечения отказоустойчивости должно быть развернуто не менее двух экземпляров с ППО БП. Тип масштабирования - горизонтальное: на этапе испытаний ППО БП определяются нагрузочные характеристики БП. При необходимости увеличения количества обрабатываемых запросов в секунду развертываются и включаются в балансировку дополнительные сервера БП.

Взаимодействие ЕБС с БП осуществляется Подсистемой балансировки и обработки запросов, которая обеспечивает реализацию следующих функций:

– обработка входящих запросов для передачи в БП и получение ответов от них;

– балансировка нагрузки на подключённые БП в соответствии с типом биометрических данных, политикой балансировки и установленных приоритетов использования БП.

Интеграция БП с Подсистемой балансировки запросов должна быть реализована посредством REST API. Каждый поставщик БП должен реализовать REST API, согласно спецификациям:

– ПРИЛОЖЕНИЕ А – для БП биометрического распознавания;

– ПРИЛОЖЕНИЕ Б – для БП обнаружения витальности;

– ПРИЛОЖЕНИЕ В – для БП биометрической верификации.

4.2 Аппаратное обеспечение

В поставку обязательно должна быть включена версия дистрибутива БП, полностью функционирующая на базе центрального процессора (CPU).

Дополнительно ППО БП может поддерживать выполнение вычислений на графических процессорах (GPU). На данный момент в эксплуатации Nvidia T4 и Nvidia A10. Версия дистрибутива БП под GPU может быть включена в поставку по возможности.

По запросу поставщика БП и в случае наличия технической возможности могут быть предоставлены доступы к наборам инструкций AVX, AVX2, доступ к GPU. Специальные требования (при наличии таковых) по настройке должны быть указаны в Инструкции Администратора.

4.3 Системное программное обеспечение

ППО БП должно поддерживать исполнение в среде контейнеризации Docker. Для запуска, перезапуска, остановки ППО БП используются средства управления контейнерами среды Docker.

ППО БП должно развёртываться в кластере docker-swarm.

Лицензия на ППО БП, при наличии таковой, должна проверяться только в офлайн режиме.

Настройка перенаправления порта приложения на порт хоста должна быть только у сервиса БП, который принимает запросы от ЕБС, сервиса БД и сервиса лицензирования, если он есть.

4.4 Требования к поставке дистрибутива БП:

Доступны следующие варианты поставки распознавания/верификации:

1. Для некоторых проектов, согласно регламенту, может быть сборка, которая содержит только БП верификации.
2. В остальных случаях поставка должна содержать БП верификации и БП распознавания. Варианты такой поставки:
    a. Два комплекта образов: отдельные сборка БП верификации и сборка БП распознавания. Обязательное условие: образы из сборки верификации не должны включать сервисы и иные сущности, необходимые для работы идентификации. Поставки должны быть совместимы по используемым векторам.
    b. Единая сборка, но с двумя комплектами файлов запуска. Первый для верификации, второй для распознавания. Обязательное условие: файлы запуска должны регулировать количество запускаемых сервисов, volumes и иных сущностей. Например, запуск в режиме верификации должен включать сущности, которые требуются только для работы верификации. То есть, компоненты, которые нужны для работы идентификации (сервисы, базы данных, configs, volumes и т. д.), не должны разворачиваться при запуске поставки в режиме верификации.

Обязательные условия:

1. Образы (images) в виде файлового архива tgz должны выкладываться в корень на FTP-сервер ftp.ebs.ru.

Имя архива образа: {имя-модуля}_{версия}.tar.gz, где:

– имя модуля – наименование модуля, входящего в поставку БП

– версия ППО БП

2. В корне должна быть создана папка с номером версии дистрибутива БП, а в ней с типом поставки БП, в которой содержатся:

– docker-compose.yml файл;

Если у вас разные версии БП под cpu и gpu – 2 compose файла с указанием [cpu|gpu] в имени файла. Если в образе содержится версия и для CPU и GPU – 1 compose файл и описание настройки (переключения CPU/GPU) в документации.

– .env файл;

– файл контрольных сумм sha256, в формате .csv;

– файл с release notes в свободном формате с указанием версии, даты релиза, ключевых изменений, ограничений, новых требований по запуску и т. п. (если по первой версии нет значимой информации в контексте тестирования и эксплуатации БП, достаточно просто указать версию ПО, дату релиза и краткое описание основных реализованных функций);

– документация;

– файлы лицензий для ключей защиты ППО БП (при необходимости);

– Dockerfile-<app-name> (при необходимости).

Файлы запуска .env и docker-compose.yml должны быть сохранены с использованием символов Unix (LF).

Пример структуры каталогов:

```
├── api-data_v1.0.17.tar.gz
├── api-data_v1.0.20.tar.gz
├── pipeline-data_v2.0.20.tar.gz
├── core-cpu_v1.0.1.tar.gz
├── nginx_v1.2.tar.gz
├── redis_v1.7.11.tar.gz
├── nbp-integration_v1.4.10.tar.gz
├── v1.0.1
│   ├── identification
│   │   ├── docker-compose.yml
│   │   ├── .env
│   │   ├── sha256.csv
│   │   ├── admin-docs.pdf
│   │   ├── licence
│   │   ├── Dockrefile-embeder
│   │   └── Dockerfile-redis
│   ├── liveness-photo-gpu
│   │   ├── docker-compose.yml
│   │   ├── .env
│   │   └── …
│   ├── liveness-photo-cpu
│   │   ├── docker-compose.yml
│   │   ├── .env
│   │   └── ...
│   └── verification-photo-cpu
│       ├── docker-compose.yml
│       └── ...
└── v1.0.2
    ├── identification
    │   ├── docker-compose.yml
    │   └── ...
    ├── liveness-photo-cpu
    │   ├── docker-compose.yml
    │   └── ...
    └── verification-photo-gpu
        ├── docker-compose.yml
        └── ...
```

4.5 Требования к образам (контейнерам)

– Контейнеры на базе Linux.

– Все приложения входящие в дистрибутив поставки должны быть в отдельных контейнерах, без supervisor.

Необходимо исключить ситуации, когда все сервисы помещены в один контейнер. ПО отвечающее за расчёт БКШ должно быть отдельно от БД с векторами.

Например, для биометрического распознавания имеем 4 сервиса:

–BD – mysql;

–api – nginx;

–core – расчет БКШ;

–nbp-integration – nginx с api для EBS.

При помещении всех данных сервисов в один контейнер с supervisor, при развёртывании в swarm с «deploy: replicas: 10» будем иметь 10 разных баз данных с векторами. Биометрическое распознавание работать будет только по процессам верификации, т.к. по round robin наш запрос попадёт в реплику, возможно не содержащую требуемый вектор. Либо мы будем иметь 10 копий одной и той же базы.

Правильно сделать каждый сервис в отдельном контейнере и при необходимости реплицировать только core.

Если вы используете образы сторонних разработчиков с supervisor и контейнеры, запущенные с этих образов, в рамках вашего дистрибутива возможно реплицировать, — оставьте.

– Контейнеры отвечающие за обработку вектора должны иметь возможность масштабирования, путём запуска нескольких копий

– Софт внутри контейнера, по возможности, должен запускаться под непривилегированным пользователем (не root);

– Образы должны быть "чистыми", без инструментов разработки, кэшей, исходников и т.д.

– Писать все журналируемые события на уровне контейнеров (STDOUT)

– Конфигурироваться ППО БП должно через переменные окружения (environment)

– В новых поставках следует указывать новую версию ПО. Тэги образов (TAGS) должны соответствовать новой версии.

Желательно выполнение следующих рекомендаций:

– В compose не используйте ENTRYPOINT, CMD, ARG. Прописывайте их в Dockerfile при сборке образа.

– В образы надо установить пакет tzdata. Timezone устанавливать через enviropment TZ=Europe/Moscow, а не монтировать /etc/localtime.

– Использовать совместимые с openshift образы, например для mariadb - https://docs.openshift.com/online/pro/using_images/db_images/mariadb.html.

Openshift при запуске подменяет uid пользователя. Это непривилегированный пользователь, и UID, который будет использоваться во время выполнения ENTRYPOINT, заранее неизвестен. С точки зрения технического проектирования это означает, что каталоги и файлы, в которые могут быть записаны процессы в контейнере, должны принадлежать корневой группе и быть доступны для чтения/записи с GID=0. Файлы, которые должны быть выполнены, также должны иметь групповые разрешения на выполнение https://cloud.redhat.com/blog/a-guide-to-openshift-and-uids

4.6 Требования к docker-compose файлу

– docker-compose.yml должен быть написан для работы в кластере docker-swarm. Поддержка Docker Engine 25.X, Docker Compose 2.24.1.

– Переменные (environment) должны выть вынесены в отдельный файл .env. *Не используйте для передачи окружения в сервисы директиву env_file и всё содержимое файла .env. Для каждого сервиса должны быть явно перечислены только те переменные из этого файла, которые требуются для работы данного сервиса. Например:*

```
services:

  webapp:
```

```
            environment:
                - DEBUG_MODE=${DEBUG_MODE}
                - API_VERSION=${API_VERSION}
                - PATH_PREFIX=${PATH_PREFIX}
                - TZ=${TZ}
```

– Все сервисы НЕ должны использовать папки/файлы из хостовой системы напрямую, только через volumes: *https://docs.docker.com/storage/volumes/* . *Не используйте «volumes: type: bind», для того чтобы не копировать вручную ваши файлы на все ноды swarm; используйте «volumes: - type: volume» (и при необходимости инициализируйте ваши volume отдельными контейнерами);*

– Настройте метки (labels):

app.part-off: ${VENDOR}

app.version: ${VERSION}

– Настройте healthcheck (должен быть настроен для каждого сервиса).

– Настройте логирование, в .env должны вынесены переменные: "max-size", "max-file"

Пример запуска:

```
env $(cat .env | grep ^[A-Z] | xargs) docker stack deploy --with-registry-auth -c <vendor-version>-
stack.yml <vendor_name>

Пример файла docker compose (<vendor-version>-stack.yml):

networks:
  net:
    driver: overlay


volumes:
  api_data:
  db_data:
  pipeline_data:


x-image: &core-app
  image: ${REGISTRY_URl}/${VENDOR}/${VENDOR_CORE_IMAGE}:${VENDOR_CORE_TAG}
  labels: &def-labels
    app.part-off: ${VENDOR}
    app.version: ${VERSION}
  # https://docs.docker.com/compose/compose-file/compose-file-v3/#logging
  logging: &def-logging
    options:
      max-size: ${LOG_MAX_SIZE}
```

```
      max-file: ${LOG_MAX_FILE}

  volumes:

    - pipeline_data:${PIPELINE_STORAGE_PATH}
  environment:
    LC_ALL:${ LC_ALL}

    PYTHON_ENCODING:${PYTHON_ENCODING}

    LOG_LEVEL:${LOG_LEVEL}

    TZ:${TZ}
    PIPELINE_LOG_LEVEL:${PIPELINE_LOG_LEVEL}

    CORE_LOG_LEVEL:${PIPELINE_LOG_LEVEL}

  shm_size: 1g

  networks:

    - net



services:

  ##### initial containers #####

  pipeline-volume:

    image: ${REGISTRY_URl}/${VENDOR}/${PIPELINE_VOLUME_IMAGE}:${PIPELINE_VOLUME_TAG}

    labels: *def-logging

    volumes:

      - pipeline_data:${PIPELINE_DATA_PATH}

    deploy:

      mode: global

        restart_policy:

          max_attempts: 1



  api-volume:

    image: ${REGISTRY_URl}/${VENDOR}/${API_VOLUME_IMAGE}:${API_VOLUME_TAG}

    labels: *def-logging

    volumes:

      - api_data:${API_DATA_PATH}

    deploy:

      mode: global

        restart_policy:

          max_attempts: 1
```

```
##### work containers #####

redis:

  image: ${REGISTRY_URl}/${VENDOR}/${REDIS_IMAGE}:${REDIS_TAG}

  healthcheck:

    test: ["CMD", "redis-cli","ping"]

    interval: 30s

    timeout: 10s

    retries: 3

  labels: *def-logging

  logging: *vendor-logging

  deploy:

    mode: replicated

    replicas: 1

    restart_policy:

      condition: any
  environment:

    DB_DATA_PATH=${B_DATA_PATH}

    DB_PORT=${DB_PORT)

  networks:

    - net

  volumes:

    - db_data:${DB_STORAGE_PATH}




embeder:

  <<: *core-app

  healthcheck:

    test: ["CMD", "embeder- healthcheck"]

    interval: 30s

    timeout: 10s

    retries: 3

  deploy:

    mode: replicated

    replicas: ${CORE_COUNT}

    placement:

       max_replicas_per_node: ${REPLICAS_PER_NODE}
```

```
    restart_policy:
        condition: any


da:
  <<: *core-app
  healthcheck:
    test: ["CMD", "da-healthcheck"]
    interval: 30s
    timeout: 10s
    retries: 3
  environment:
    PIPELINE_DATA_PATH=${PIPELINE_DATA_PATH}
  deploy:
    mode: replicated
    replicas: ${CORE_COUNT}
    placement:
       max_replicas_per_node: ${REPLICAS_PER_NODE}
    restart_policy:
        condition: any



api:
  image: ${REGISTRY_URl}/${VENDOR}/${NGINX_IMAGE}:${NGINX_TAG}
  labels:
    app.part-off: ${VENDOR}
    app.version: ${VERION}
  labels: *def-logging
  logging: *vendor-logging
  healthcheck:
    test: ["CMD", "api-healthcheck"]
    interval: 30s
    timeout: 10s
    retries: 3
  deploy:
    mode: replicated
    replicas: 1
```

```
      restart_policy:
        condition: any
    environment:
      API_DATA_PATH=${API_DATA_PATH}
      API_PORT=${API_PORT}
    volumes:
      - api_data:${API_DATA_PATH}
    networks:
      - net


  nbp-integration:
    image: ${REGISTRY_URl}/${VENDOR}/${NPB_INTEGRATION_IMAGE}:${NPB_INTEGRATION_TAG}
    labels:
      app.part-off: ${VENDOR}
      app.version: ${VERION}
    logging:
      <<: *vendor-logging
    healthcheck:
      test: ["CMD", "nbp-healthcheck"]
      interval: 30s
      timeout: 10s
      retries: 3
    deploy:
      mode: replicated
      replicas: 1
      restart_policy:
        condition: any
    environment:
      NBP_INTEGR_PORT=${NBP_INTEGR_PORT}
    ports:
      - ${NBP_INTEGR_PORT}:80
    networks:
      - net
```

4.7 Требования к .env файлу

1. Через файл .env должна быть реализована возможность менять порты для сервиса, принимающего запросы от ЕБС, сервиса БД и для сервиса лицензирования при его наличии.
2. Должна быть реализована возможность настройки параметра «pathPrefix», в том числе с возможностью задания пустых значений (где {pathPrefix} – необязательная составная часть пути URL к API БП).
3. Обязательные параметры, должны быть с такими именами, используются в сценариях развёртывания:

3.1. Основные:

VENDOR="название БП"

VERSION="версия дистрибутива"

REGISTRY_URl="server:port" # Указывайте свой. При развёртывании, мы изменим на наш.

CORE_COUNT= # Количество копий контейнеров отвечающих за обработку векторов

REPLICAS_PER_NODE= # Максимальное количество реплик на каждой ноде

3.2. Настройки логирования:

LOG_MAX_SIZE= размер файла лога

LOG_MAX_FILE= количество фалов лога

3.3. Имена, тэги образов:

APP1_IMAGE= имя образа

APP1_TAG= тэг (версия)

APP2_IMAGE= …

APP2_TAG= …

4. Опционально. Другие параметры, конфигурационные переменные. Например:

LC_ALL=en_US.UTF-8

PYTHONIOENCODING=UTF-8\

LOG_LEVEL=ERROR

...

5. Если БП состоит из нескольких сервисов, и для каждого метода API требуется регулировать количество определенных сервисов, можно создать несколько соответствующих переменных типа CORE_COUNT и REPLICAS_PER_NODE, например:

ADAPTER_COUNT=…

ADAPTER_REPLICAS_PER_NODE=…

HANDLERS_COUNT=…

HANDLERS_REPLICAS_PER_NODE= …

…

Пример файла .env:

```
### General

VENDOR="Vendor-name"

VERSION="v1.0.1"

REGISTRY_URl="server:port/"

CORE_COUNT=3
REPLICAS_PER_NODE=1



### Logging

LOG_MAX_SIZE="30m"

LOG_MAX_FILE=3



### Images

PIPELINE_IMAGE=pipeline-data

PIPELINE_TAG=v2.0.20

API_VOLUME_IMAGE=api-data

API_VOLUME_TAG=v1.0.17

CORE_IMAGE=core-cpu

CORE_TAG=v1.0.1

NGINX_IMAGE=nginx

NGINX_TAG=v1.2

REDIS_IMAGE=redis

REDIS_TAG=v1.7.11

NPB_INTEGRATION_IMAGE=nbp-integration

NPB_INTEGRATION_TAG=v1.4.10



### Other

# All

LC_ALL=en_US.UTF-8

PYTHON_ENCODING=UTF-8

LOG_LEVEL=ERROR

TZ=Europe/Moscow

# Core
```

```
PIPELINE_LOG_LEVEL=${LOG_LEVEL}

CORE_LOG_LEVEL=${LOG_LEVEL}

# Redis

DB_DATA_PATH=/path/to/data

DB_PORT=6379

# Embeder

PIPELINE_DATA_PATH=/path/to/data

# API

API_DATA_PATH=/path/to/data

API_PORT=18080

# NBP

NBP_INTEGR_PORT=8090
```

4.8 Требования к файлу контрольных сумм

В файле контрольных сумм должны быть указаны все образы, используемые в данной версии дистрибутива и их контрольные суммы sha256 (соответствует IMAGE ID при выводе информации об образе). Разделитель точка с запятой.

Пример файла sha256.csv:

```
pipeline-data:v2.0.20; a182afb11a25df72709f77c57fc4b5cd724217d4302f810e571e1cae2fbb2e70

api-data:v1.0.17; 6da5cd9b2a3503c8c80966f92c6ab7893d1495e94abc9714c6c419d8597d00e1

core-cpu:v1.0.1; 7068e2b52d4379b8d6d603b545275e2e363254910b60cbebc024dc1d06d3c4dc

nginx:v1.2; 301827bf4200b4cc15a171cad08473f8993cc2d3a985155952ae5fea75735f28

redis:v1.7.11; 2b9dc857b15124d38c8a7da56d85cf8d456b5f95036d8256ecaabb162aab64c7

nbp-integration:v1.4.10; 1d844a019734d30ff6b0393d731d45a123e01c4c0aad8bc72a81c476505dc539
```

Посмотреть контрольную сумму докер-образа можно, выполнив команду:

```
docker image inspect --format='{{.Id}}' IMAGE [IMAGE...]
```

4.9 Требования к документации

Документация должна содержать следующую информацию:

– архитектура и взаимодействия между модулями в решении;

– порядок установки и настройки, в том числе настройки лицензирования ППО БП (если это требуется);

– порядок настройки файлов запуска на случай, если потребуется развернуть сервис БД вне кластера docker-swarm: для каких сервисов потребуется указать хост и порт сервиса БД, и иные настройки, которые может потребоваться выполнить для запуска БП;

– перечень особенностей при запуске сервисов, в случае если такие имеются (утилизация большого количества CPU при запуске и т.д.);

– рекомендации по масштабированию сервисов, включая БД (репликация, шардинг и т.д.). Рекомендации должны включать список компонентов, которые возможно масштабировать для обеспечения большей производительности, а также особенности масштабирования компонентов, в частности информацию о том, какие сервисы масштабируются динамически, а какие только при перезапуске БП;

– если в поставке есть БД для хранения векторов, описание запросов (SQL, API-запрос и т.п.) на получение количества сохраненных векторов;

– инструкция по созданию индексов в том случае, если это требуется для оптимизации работы БД (например, при больших объемах данных), и информация по индексу (тип, нюансы использования и т. д.);

– специальные требования по настройке, в частности, предоставление доступа к наборам инструкций AVX, AVX2, GPU;

– описание параметров конфигурации;

– описание api модулей;

– описание CLI, WebGUI ...

– типовые ошибки и способы их устранения.

4.10 Требования по журналированию

ППО БП должно журналировать события в STDOUT. Необходимо реализовать журналирование следующих типов событий:

– вызовы функций API, согласно спецификациям (Приложение А – для БП биометрического распознавания; Приложение Б – для БП обнаружения витальности; Приложение В – для БП биометрической верификации);

– изменения состояния (запуск, перезапуск, остановка) ППО БП;

– ошибки в работе ППО БП.

Записи журналов событий должны содержать следующие данные:

– тип события;

– критичность события;

– компонент, в котором возникло событие;

– дата и время возникновения события;

– текст сообщения, в т.ч. результат выполнения функции ППО БП (с указанием кода и описания ошибки, в случае ее возникновения).

Уровень журналирования должен быть достаточен для диагностирования возникающих при эксплуатации ошибок, в частности должны журналироваться причины возникновения ошибок методов API, указанных в Приложениях А и Б, достаточные для анализа проблемы.

Должна иметься возможность однозначно определить записи журнала, относящиеся к единичному вызову метода API (из Приложения А, Б или В). Например, путем присвоения всем связанным записям в логах сервисов одинакового уникального идентификатора, назначенному данному единичному вызову метода API.

4.11 Лицензирование

Лицензия на ППО БП, при наличии таковой, должна проверяться только в офлайн режиме.

При невозможности реализации сервера лицензирования в контейнере, его дистрибутив должен отвечать следующим требованиям:

- RPM пакет совместимый с RedOS 7.3;
- все файлы сервера лицензирования устанавливаются в каталог /opt/<vendor_name>-license-srv-<version>;
- должна быть реализована настройка порта.

Необходимо реализовать способ получения сведений о текущей лицензии, как минимум о ее статусе и сроке действия. Возможные варианты:

- веб-интерфейс сервиса лицензирования (при его наличии) или разработка такого интерфейса;
- API endpoint на стороне БП.

В документации к БП следует описать способ получения данных сведений (например, примеры запросов).

4.12 Настройка отказоустойчивости

Одним из обязательных условий прохождения тестирования является обеспечение отказоустойчивости, для этого должно быть развернуто не менее двух экземпляров с ППО БП. Также необходимо реализовать автоматическое восстановление сервисов с сохранением работоспособности API методов, в случае возникновения аварийных ситуаций, например: отключение одной из нод кластера, перезапуска сервиса docker. Настройка healthcheck и restart_policy должна быть произведена для каждого из сервисов.

По умолчанию БП размещаются в защищенном контуре ЕБС в кластере docker-swarm.

Дополнительно требуется подготовить файлы запуска (.env и docker-compose.yml) таким образом, чтобы при необходимости можно было развернуть сервис БД вне кластера swarm.

Например, для сервиса БД должна быть возможность указать перенаправление порта приложения на порт хоста, на котором он будет развернут. А для остальных сервисов БП, которые будут развернуты в кластере swarm и обращаться к БД, должна быть возможность указать хост и порт сервиса БД. И тогда при необходимости БД можно развернуть как отдельный проект посредством docker compose up на выделенном хосте. Таким образом, если потребуется обновить поставку и перезапустить основную группу сервисов, БД не будет затронута.

Аналогичные настройки требуются для сервиса лицензии при его наличии.

Все контейнеризированные приложения должны корректно обрабатывать сигнал прерывания или завершения, обеспечивая graceful shutdown: прекращать прием новых запросов, завершать текущие операции в пределах stop grace period, освобождать ресурсы и корректно завершать работу. Если для завершения вашего приложения требуется другой сигнал и период, отличные от значений по умолчанию, следует указать в документации к БП актуальные настройки.

Примеры настроек https://docs.docker.com/reference/compose-file/services:

```
stop_signal: SIGTERM      # отправки SIGTERM

stop_grace_period: 25s    # максимальное время ожидания, далее SIGKILL
```

4.13 Требования к версионности БП

Изменение версии нейронной сети, входящей в состав БП, влекущее за собой модификацию ее архитектуры, алгоритмов или весовых коэффициентов, и соответственно несовместимость биометрических шаблонов, должно сопровождаться увеличением мажорной версии биометрического процессора.

В документации следует описать принятый подход к версионированию приложения. Например, структура и формат версии, какой блок цифр используется для указания версии нейронной сети, патчей и т.д.

ТРЕБОВАНИЯ ДЛЯ ИНИЦИАЦИИ ПРОЦЕДУРЫ ПОДКЛЮЧЕНИЯ К ЕБС

Регистрация Поставщика БП: при подключении к ЕБС Поставщика БП необходимо пройти все процедуры в соответствии с Регламентом использования биометрических процессоров в Государственной информационной системы Единая биометрическая система.

ПРИЛОЖЕНИЕ А. Описание интеграции БП биометрического распознавания ЕБС

Общее описание API БП

Интеграция БП в составе ЕБС с Подсистемой балансировки запросов должна быть реализована посредством REST API. Архитектура REST (Representational State Transfer) подразумевает наличие клиент-серверной архитектуры. Клиент (здесь и далее в контексте описания REST API - Подсистема балансировки запросов) инициирует запросы к Серверу (здесь и далее в контексте описания REST API - БП), сервер обрабатывает их и возвращает ответ.

Каждый поставщик БП должен реализовать REST API, согласно спецификации, приведенной ниже.

REST API определяет набор методов, к которым Подсистема балансировки запросов может совершать запросы и получать ответы. Взаимодействие происходит по протоколу HTTP. Обязательна поддержка постоянных соединений (keep-alive).

Список используемых методов REST API, а также необходимые параметры и возвращаемые значения, приведены ниже в данном документе.

Версия API

Актуальная версия API БП: «v1».

Формат версии: префикс «v» и целое число.

Точка доступа к API

Базовый URL доступа к API:

http://{hostname}/{version}/{pathPrefix}/

Где:

− *{hostname}* – имя хоста и порт сервера развернутого БП;

− *{version}* – используемая версия API БП. Формат версии в пути HTTP запросов: префикс «v» и целое число; актуальная версия API подключаемых БП биометрического распознавания: «v1».

− *{pathPrefix}* – необязательная составная часть пути URL к API БП. Может использоваться, например, при установке на сервер БП нескольких модальностей. Должна быть реализована возможность настройки параметра «pathPrefix» через переменные окружения, в том числе с возможностью задания пустых значений.

Пример запроса[1]:

```
GET /v1/{pathPrefix}/health HTTP/1.1

     Host: bioproc1.ebs.ru
```

Поддерживаемые в запросах методы HTTP и типы контента

Сервер должен поддерживать следующие методы HTTP:

1 Здесь и далее в примерах запросов и ответов часть HTTP-заголовков не приводится для краткости.

− GET;

− POST.

Сервер должен поддерживать следующие типы контента запроса (HTTP-заголовок «Content-Type»):

− «multipart/form-data»;

− «application/octet-stream»;

− «image/jpeg» или «image/png» (для БП модальности «Фотоизображение лица»);

− «audio/wav» (для БП модальности «Аудиозапись голоса»).

Если тип контента запроса - «application/json», то входные параметры метода передаются в теле POST-запроса в формате JSON.

Если тип контента запроса - «multipart/form-data», то каждый входной параметр метода передается как отдельная часть составного содержимого HTTP-запроса и следует правилам для составных MIME-данных в соответствии с RFC 2045.

Порядок заголовков внутри части multipart - запроса и порядок самих частей multipart - запроса может быть изменен. Каждая часть должна содержать:

− заголовочное поле «Content-Disposition», имеющее значение «form-data»;

− атрибут «name» поля «Content-Disposition», имеющий значение, равное наименованию входного параметра (см. ниже таблицы с описанием входных параметров соответствующих методов);

- параметр «filename» поля «Content-Disposition» не является обязательным для передачи в запросе. В случае наличия параметра «filename», отсутствие в значении параметра расширения не должно завершаться с ошибкой. Вызов метода без параметра «filename» не должен завершаться с ошибкой;

− заголовочное поле «Content-Type», принимающая значение в зависимости от контента, передаваемого в части:

− БО, модальность «Фотоизображение лица»: «image/jpeg» или «image/png»;

− БО, модальность «Аудиозапись голоса»: «audio/wav»;

− биометрический шаблон: «application/octet-stream».

Следует отметить, что boundary (граница) — это последовательность байтов, которая не должна встречаться внутри закодированного представления данных части.

Если тип контента запроса - «application/octet-stream», «image/jpeg», «image/png», «audio/wav», то метод принимает на вход только один бинарный параметр.

В каждом HTTP-запросе присутствует набор обязательных параметров, но могут быть определены дополнительные параметры, требуемые только для конкретного метода. Текстовые значения параметров передаются в кодировке UTF-8.

Для процесса идентификации каждая запись ассоциирована со своим уникальным идентификатором “template_id” (String) и значением ее Биометрического шаблона (БШ).

Используемые коды ответов HTTP

Используемые Системой коды ответов HTTP приведены в таблице ниже.

Коды ответа	Описание
200 OK	Вызов метода завершился успешно. Ответ Сервера включен в HTTP BODY.
400 Bad Request	Вызов метода завершился с ошибкой на стороне Клиента. Код ошибки включен в HTTP BODY.
500 Internal Server Error	Вызов метода завершился с ошибкой на стороне Сервера. Код ошибки включен в HTTP BODY.

Все успешные ответы:

1. содержат код ответа HTTP 200;

2. возвращают:

- JSON объект со значениями выходных параметров метода в HTTP BODY. Тип контента - «application/json»;
- бинарный контент (например, если метод возвращает биометрический шаблон) с указанием типа контента в HTTP-заголовке «Content-Type»;
- составной контент (например, если метод возвращает степень схожести и биометрический шаблон), с указанием типа контента «multipart/form-data» в HTTP-заголовке «Content-Type».

Все ответы с ошибкой:

- содержат код ответа HTTP 400 или 500;
- возвращают JSON объект с описанием ошибки в HTTP BODY. Тип контента «application/json».

Пример ответа с ошибкой:

```
HTTP/1.1 500 Internal Server Error
Content-Type: application/json; charset=UTF-8
{
        "code": "BPE-001001",
        "message": "Внутренняя ошибка биометрического процессора"
}
```

Методы API БП

Метод «Извлечение биометрического шаблона» - extract

Метод предназначен для извлечения биометрического шаблона из БО.

Тип контента HTTP-запроса:

− «image/jpeg» или «image/png» (для БП модальности «Фотоизображение лица»);

− «audio/wav» (для БП модальности «Аудиозапись голоса»).

Вызов метода: POST /*{версия}*/*{pathPrefix}*/extract

Входные параметры:

Наименование параметра	Значение	Описание
–[2]	Stream	Обязательный параметр. БО для извлечения биометрического шаблона.[3] В заголовочном поле «Content-Type» должен быть указан тип контента соответствующей модальности БО.

Пример запроса (модальность «Фотоизображение лица»):

```
POST /v1/{pathPrefix}/extract HTTP/1.1
Content-Type: image/jpeg

{Поток байт БО}
```

Успешный ответ метода:

В случае успешного ответа, метод возвращает биометрический шаблон.

Тип контента HTTP-ответа: «application/octet-stream».

Выходные параметры:

Наименование параметра	Значение	Описание
– [4]	Stream	Биометрический шаблон, полученный из БО.

Пример:

```
HTTP/1.1 200 OK
Content-Type: application/octet-stream

{Поток байт биометрического шаблона}
```

2 Здесь и далее наименование параметра отсутствует и не передается в запросе

3 Здесь и далее порядок байтов в БО и шаблонах «little-endian»

4 Здесь и далее наименование параметра отсутствует и не передается в ответе

Ошибки метода:

В случае возникновения ошибки при обработке запроса, Система возвращает вызывающей стороне коды ответов HTTP и описания ошибок в HTTP BODY, согласно таблице ниже.

Код ответа HTTP	Значение параметра «code»	Описание
400	BPE-002001	Неверный Content-Type HTTP-запроса
400	BPE-002002	Неверный метод HTTP-запроса
400	BPE-002003	Не удалось прочитать биометрический образец
400	BPE-003001	Не удалось извлечь биометрический шаблон
400	BPE-003002	Сообщение отличается в зависимости типа контента: • audio: На биометрическом образце отсутствует голос • image: На биометрическом образце отсутствует лицо
400	BPE-003003	Только для типа контента image. На биометрическом образце присутствует более, чем одно лицо * Для определения лица следует руководствоваться следующими требованиями: Расстояние между центрами глаз (IOD - inter-ocular distance): • IOD > 16px • IOD > 5% от ширины кадра Оба условия накладываются одновременно. Все, что не соответствует предъявленным требованиям, не должно считаться лицом.
500	BPE-001001	Внутренняя ошибка биометрического процессора

Метод «Сравнение биометрических шаблонов» - compare

Метод предназначен для сравнения двух биометрических шаблонов: шаблона, извлеченного из БО, и БКШ.

Тип контента HTTP-запроса: «multipart/form-data».

Вызов метода: POST /*{версия}*/*{pathPrefix}*/compare

Входные параметры:

Наименование параметра	Значение	Описание
bio_feature	Stream	Обязательный параметр. Биометрический шаблон, полученный из БО при биометрической верификации.
bio_template	Stream	Обязательный параметр. БКШ, с которым производится сравнение.

Пример запроса:

```
POST /v1/{pathPrefix}/compare HTTP/1.1
Content-Type: multipart/form-data; boundary=f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Length: {длина тела сообщения}

--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="bio_feature"
Content-Type: application/octet-stream

{Поток байт шаблона, полученный из БО при биометрической верификации}

--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="bio_template"
Content-Type: application/octet-stream

{Поток байт БКШ}
--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH--
```

Успешный ответ метода:

В случае успешного ответа, метод возвращает результат сравнения двух биометрических шаблонов.

Тип контента HTTP-ответа: «application/json».

Выходные параметры:

Наименование параметра	Значение	Описание
score	Number	Обязательный параметр. Степень схожести с поступившим на сравнение биометрическим шаблоном, может принимать только

		дробные значения от 0.0 до 1.0 включительно.

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
  "score": 0.8
}
```

Ошибки метода:

В случае возникновения ошибки при обработке запроса, Система возвращает вызывающей стороне коды ответов HTTP и описания ошибок в HTTP BODY, согласно таблице ниже.

Код ответа HTTP	Значение параметра «code»	Описание
400	BPE-002001	Неверный Content-Type HTTP-запроса
400	BPE-002002	Неверный метод HTTP-запроса
400	BPE-002004	Не удалось прочитать биометрический шаблон
400	BPE-002005	Неверный Content-Type части multiparted HTTP-запроса
500	BPE-001001	Внутренняя ошибка биометрического процессора

Метод «Биометрическая верификация» - verify

Является объединением двух методов, описанных выше. Метод извлекает биометрический шаблон из БО и сравнивает полученный биометрический шаблон с БКШ, переданным в качестве параметра.

Тип контента HTTP-запроса: «multipart/form-data».

Вызов метода: POST /*{версия}*/*{pathPrefix}*/verify

Входные параметры:

Наименование параметра	Значение	Описание
bio_template	Stream	Обязательный параметр.

		БКШ, с которым производится сравнение.
sample	Stream	Обязательный параметр. БО для извлечения биометрического шаблона. В заголовочном поле «Content-Type» части составного типа контента должен быть указан тип контента соответствующей модальности БО (см. раздел «Поддерживаемые в запросах методы HTTP и типы контента»).

Пример запроса (модальность «Фотоизображение лица»):

```
POST /v1/{pathPrefix}/verify HTTP/1.1
Content-Type: multipart/form-data; boundary=f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Length: {длина тела сообщения}

--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="bio_template"
Content-Type: application/octet-stream

{Поток байт БКШ}

--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="sample"
Content-Type: image/jpeg

{Поток байт БО}
--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH--
```

Успешный ответ метода:

В случае успешного ответа, метод возвращает результат сравнения двух биометрических шаблонов и биометрический шаблон.

Тип контента HTTP-ответа: «multipart/form-data».

Выходные параметры:

Наименование параметра	Значение	Описание
score	Number	Обязательный параметр.

Наименование параметра	Значение	Описание
		Степень схожести с поступившим на сравнение биометрическим шаблоном, может принимать только дробные значения от 0.0 до 1.0 включительно.
bio_feature	Stream	Обязательный параметр. Биометрический шаблон, полученный из БО (входной параметр «sample»).

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: multipart/form-data; boundary=f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Length: {длина тела сообщения}

--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="score"
Content-Type: application/json
{
  "score": 0.8
}

--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="bio_feature"
Content-Type: application/octet-stream

{Поток байт Биометрического шаблона}
--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH--
```

Ошибки метода:

В случае возникновения ошибки при обработке запроса, Система возвращает вызывающей стороне коды ответов HTTP и описания ошибок в HTTP BODY, согласно таблице ниже.

Код ответа HTTP	Значение параметра «code»	Описание
400	BPE-002001	Неверный Content-Type HTTP-запроса
400	BPE-002002	Неверный метод HTTP-запроса

Код ответа HTTP	Значение параметра «code»	Описание
400	BPE-002003	Не удалось прочитать биометрический образец
400	BPE-002004	Не удалось прочитать биометрический шаблон
400	BPE-002005	Неверный Content-Type части multiparted HTTP-запроса
400	BPE-003001	Не удалось извлечь биометрический шаблон
400	BPE-003002	Сообщение отличается в зависимости типа контента: • audio: На биометрическом образце отсутствует голос • image: На биометрическом образце отсутствует лицо
400	BPE-003003	Только для типа контента image. На биометрическом образце присутствует более, чем одно лицо * Для определения лица следует руководствоваться следующими требованиями: Расстояние между центрами глаз (IOD - inter-ocular distance): • IOD > 16px • IOD > 5% от ширины кадра Оба условия накладываются одновременно. Все, что не соответствует предъявленным требованиям, не должно считаться лицом.
500	BPE-001001	Внутренняя ошибка биометрического процессора

Метод «Проверка состояния БП» - health

Метод возвращает текущее состояние работоспособности БП. Реализация метода должна обеспечить проверку всех зависимостей БП, необходимых для работоспособности методов API БП путем вызова всех соответствующих внутренних функций на контрольных примерах. Метод должен возвращать положительный ответ только в случае успешного прохождения проверки всех внутренних функций.

Подсистема мониторинга ЕБС обрабатывает ответ метода следующим образом:

− БП работает корректно: в случае получения от метода успешного ответа (см. таблицу ниже), параметр «status» равен 0;

− БП неработоспособен: в случае получения от метода успешного ответа (см. таблицу ниже), параметр «status» равен 3;

− REST API БП неработоспособно (внутренняя ошибка): в случае получения от метода ошибки с HTTP-кодом 500;

– анализирует время отклика метода для сбора статистики производительности и утилизации, генерации событий мониторинга при превышении пороговых значений.

Вызов метода: GET /{версия}/{pathPrefix}/health

Входные параметры:

Отсутствуют.

Пример запроса:

```
GET /v1/{pathPrefix}/health HTTP/1.1
     Host: bioproc1.ebs.ru
```

Успешный ответ метода:

В случае успешного ответа, метод возвращает сообщение со следующими параметрами.

Тип контента HTTP-ответа: «application/json».

Выходные параметры:

Наименование параметра	Значение	Описание
status	Number	Обязательный параметр. Возможные значения параметра: 0: биометрический процессор работает корректно (данный статус обязателен для реализации); 1: Minor – в работе биометрического процессора возникло событие низкой критичности (данный статус необязателен для реализации); 2: Major – в работе биометрического процессора возникло событие высокой критичности (данный статус необязателен для реализации); 3: Critical – биометрический процессор неработоспособен (данный статус обязателен для реализации);
message	String	Необязательный параметр. Строка длиной не более 128 символов. Если значение параметра «status» не равно нулю (1-3), то рекомендуется передать данный параметр и включить в него текст с описанием возникшего события. Данное событие будет показано сотруднику дежурной смены в подсистеме мониторинга.

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
  "status": 3,
  "message": "Функция сравнения биометрических шаблонов не ответила на контрольный запрос"
}
```

Ошибки метода:

В случае возникновения ошибки при обработке запроса, Система возвращает вызывающей стороне коды ответов HTTP и описания ошибок в HTTP BODY, согласно таблице ниже.

Код ответа HTTP	Значение параметра «code»	Описание
400	BPE-002006	Неверный запрос
500	BPE-001002	Внутренняя ошибка REST API биометрического процессора

Метод «Добавление записи с БКШ в базу данных Поставщика БП» - add

Метод предназначен для передачи связки БКШ с идентификатором БКШ для добавления записи в базу данных Поставщика БП.

Тип контента HTTP-запроса:

- Content-Type: multipart/form-data
- X-Request-ID: <string> - Идентификатор запроса (UUID).

Вызов метода: POST /*{версия}*/*{pathPrefix}*/add

Входные параметры:

Наименование параметра	Значение	Описание
Часть multipart для передачи БДн		
template	часть multipart	Обязательный параметр. Содержит БКШ. Заголовочные поля: • Content-Disposition: form-data; name="template" • Content-Type: application/octet-stream

Наименование параметра	Значение	Описание
Часть multipart для передачи информации о БКШ		
metadata	часть multipart	Обязательный параметр. Часть multipart для передачи идентификатора БКШ. Заголовочные поля: • Content-Disposition: form-data; name="metadata" • Content-Type: application/json
template_id	string	Обязательный параметр. Хранится как строковый тип. Является внутренним (ЕБС) идентификатором БКШ определенного вендора (идентификатор может содержать не только цифры, но и буквы).

В заголовке части запроса с именем "metadata" может приходить кодировка текста charset=UTF-8 (Пример: Content-Type: application/json; **charset=UTF-8**). В этом случае данный параметр необходимо игнорировать.

Пример запроса:

```
POST /v1/{pathPrefix}/add
Content-Type: multipart/form-data; boundary=f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
X-Request-ID: 4896c91b-9e61-3129-87b6-8aa299028058
Content-Length: {длина тела сообщения}

--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="template"
Content-Type: application/octet-stream

{Поток байт биометрического шаблона}
--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="metadata"
Content-Type: application/json
{
"template_id": "12345"
}
--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH--
```

Успешный ответ метода:

В случае успешного ответа, метод возвращает HTTP-ответ 200 OK с пустым BODY.

Пример ответа:

```
HTTP/1.1 200 OK
```

Ошибки метода:

В случае возникновения ошибки при обработке запроса, Система возвращает вызывающей стороне коды ответов HTTP и описания ошибок в HTTP BODY, согласно таблице ниже.

Код ответа HTTP	Значение параметра «code»	Описание
400	BPE-002001	Неверный Content-Type HTTP-запроса
400	BPE-002002	Неверный метод HTTP-запроса
400	BPE-002004	Не удалось прочитать биометрический шаблон
400	BPE-002005	Неверный Content-Type части multiparted HTTP-запроса
400	BPE-00005	Не удалось прочитать метаданные
400	BPE-00502	Запрос не содержит обязательных данных {название данных/параметров}
400	BPE-00506	Недопустимое значение параметра {название параметра}
400	BPE-00507	Шаблон не добавлен. Запись с данным идентификатором уже существует в базе
500	BPE-001001	Внутренняя ошибка биометрического процессора

Метод «Запрос обновления БКШ в базе данных Поставщика БП» - update

Метод предназначен для передачи связки БКШ с идентификатором для обновления БКШ в базе данных Поставщика БП.

Тип контента HTTP-запроса:

- Content-Type: multipart/form-data
- X-Request-ID: <string> - Идентификатор запроса (UUID).

Вызов метода: POST /*{версия}*/*{pathPrefix}*/update

Входные параметры:

Наименование параметра	Значение	Описание
Часть multipart для передачи БДн		
template	часть multipart	Обязательный параметр. Содержит БКШ. Заголовочные поля: • Content-Disposition: form-data; name="template" • Content-Type: application/octet-stream
Часть multipart для передачи дополнительной информации		
metadata	часть multipart	Обязательный параметр. Часть multipart для передачи идентификатора БКШ для обновления. Заголовочные поля: • Content-Disposition: form-data; name="metadata" • Content-Type: application/json
template_id	string	Обязательный параметр. Внутренний (ЕБС) идентификатор обновляемого БКШ (идентификатор может содержать не только цифры, но и буквы).

В заголовке части запроса с именем "metadata" может приходить кодировка текста charset=UTF-8 (Пример: Content-Type: application/json; **charset=UTF-8**). В этом случае данный параметр необходимо игнорировать.

Пример запроса:

```
POST /v1/{pathPrefix}/update HTTP/1.1
Content-Type: multipart/form-data; boundary=f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
X-Request-ID: 4896c91b-9e61-3129-87b6-8aa299028058
Content-Length: {длина тела сообщения}


--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="template"
Content-Type: application/octet-stream

{Поток байт биометрического шаблона}
--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="metadata"
Content-Type: application/json
```

```
{
"template_id": "12345"
}
--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH--
```

Успешный ответ метода:

В случае успешного ответа, метод возвращает HTTP-ответ 200 OK с пустым BODY.

Пример ответа:

```
HTTP/1.1 200 OK
```

Ошибки метода:

В случае возникновения ошибки при обработке запроса, Система возвращает вызывающей стороне коды ответов HTTP и описания ошибок в HTTP BODY, согласно таблице ниже.

Код ответа HTTP	Значение параметра «code»	Описание
400	BPE-002001	Неверный Content-Type HTTP-запроса
400	BPE-002002	Неверный метод HTTP-запроса
400	BPE-002004	Не удалось прочитать биометрический шаблон
400	BPE-002404	Не найден биометрический шаблон с заданным идентификатором
400	BPE-002005	Неверный Content-Type части multiparted HTTP-запроса
400	BPE-00005	Не удалось прочитать метаданные
400	BPE-00502	Запрос не содержит обязательных данных {название данных/параметров}
400	BPE-00506	Недопустимое значение параметра {название параметра}
500	BPE-001001	Внутренняя ошибка биометрического процессора

Метод «Удаление (деактивация) записи в базе данных с БКШ» - delete

Метод предназначен для передачи идентификатора БКШ для удаления записи из базы данных Поставщика БП.

Тип контента HTTP-запроса:

− Content-Type: application/json

− X-Request-ID: <string> - Идентификатор запроса (UUID).

Вызов метода: POST /*{версия}*/*{pathPrefix}*/delete

Входные параметры:

Наименование параметра	Значение	Описание
template_id	string	Обязательный параметр. Идентификатор БКШ (идентификатор может содержать не только цифры, но и буквы).

Пример запроса:

```
POST /v1/{pathPrefix}/delete
Content-Type: application/json
X-Request-ID: 4896c91b-9e61-3129-87b6-8aa299028058
{
“template_id”: “123456”
}
```

Успешный ответ метода:

В случае успешного ответа, метод возвращает HTTP-ответ 200 OK с пустым BODY.

Пример ответа:

```
HTTP/1.1 200 OK
```

Ошибки метода:

В случае возникновения ошибки при обработке запроса, Система возвращает вызывающей стороне коды ответов HTTP и описания ошибок в HTTP BODY, согласно таблице ниже.

Код ответа HTTP	Значение параметра «code»	Описание
400	BPE-002001	Неверный Content-Type HTTP-запроса
400	BPE-002002	Неверный метод HTTP-запроса
400	BPE-002006	Неверный запрос
400	BPE-002404	Не найден биометрический шаблон с заданным идентификатором
400	BPE-00005	Не удалось прочитать метаданные

Код ответа HTTP	Значение параметра «code»	Описание
400	BPE-00502	Запрос не содержит обязательных данных {название данных/параметров}
400	BPE-00506	Недопустимое значение параметра {название параметра}
500	BPE-001001	Внутренняя ошибка биометрического процессора

Метод «Поиск ближайших совпадений» - match

Метод предназначен для поиска максимально похожих на присланный биометрический шаблон идентификаторов БКШ в базе данных Поставщика БП.

Тип контента HTTP-запроса:

− Content-Type: multipart/form-data

− X-Request-ID: <string> - Идентификатор запроса (UUID)

Вызов метода: POST /*{версия}*/*{pathPrefix}*/match

Входные параметры:

Наименование параметра	Значение	Описание
Часть multipart для передачи БДн		
template	часть multipart	Обязательный параметр. Содержит биометрический шаблон для поиска схожих в базе данных Поставщика БП. Заголовочные поля: • Content-Disposition: form-data; name=”template” • Content-Type: application/octet-stream
Часть multipart для передачи дополнительной информации		
metadata	часть multipart	Обязательный параметр. Часть multipart для передачи параметров поиска. Заголовочные поля: • Content-Disposition: form-data; name=”metadata” • Content-Type: application/json
threshold	Number	Обязательный параметр.

Наименование параметра	Значение	Описание
		Пороговое значение степени схожести, может принимать только дробные значения от 0.0 до 1.0 включительно. Результаты, у которых степень схожести ниже, чем значение порогового значения, не нужны в выдаче.
Limit	Number	Обязательный параметр. Максимальное количество записей в выдаче. Может принимать значение от 1 до 50 (включительно).

В заголовке части запроса с именем "metadata" может приходить кодировка текста charset=UTF-8 (Пример: Content-Type: application/json; **charset=UTF-8**). В этом случае данный параметр необходимо игнорировать.

Пример запроса:

```
POST /v1/{pathPrefix}/match HTTP/1.1
Content-Type: multipart/form-data; boundary=f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
X-Request-ID: 4896c91b-9e61-3129-87b6-8aa299028058
Content-Length: {длина тела сообщения}


--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name=”template”
Content-Type: application/octet-stream

{Поток байт биометрического образца}
--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name=”metadata”
Content-Type: application/json
{
“threshold“: 0.3,
“limit”: 5
}
--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH--
```

Успешный ответ метода:

В случае успешного ответа, метод возвращает HTTP-ответ 200 OK, в HTTP BODY включены наборы прошедших пороговые значения степени схожести пар идентификаторов и значения степени схожести.

БП выдаёт найденный(е) идентификатор(ы) биометрического(их) шаблона(ов) и степень схожести.

В ответе может быть возвращен более чем один набор значений в случае нахождения нескольких совпадений выше порогового значения.

Набор совпадений должен быть отсортирован по убыванию степени схожести.

Если не найдено ни одного совпадения, то в ответ должен прийти пустой массив.

Если в базе БП отсутствуют записи, то в ответ должен прийти пустой массив

Выходные параметры:

Наименование параметра	**Значение**	**Описание**
template_id	string	Обязательный параметр. Внутренний (ЕБС) идентификатор БКШ.
Similarity	Number	Обязательный параметр. Степень схожести с поступившим на сравнение биометрическим шаблоном, может принимать только дробные значения от 0.0 до 1.0 включительно.

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
[
{
“template_id”: “1134569”,
“similarity”: 0.6
},
{
“template_id”: “1134567”,
“similarity”: 0.54
},
{
“template_id”: “1134568”,
“similarity”: 0.45
},
{
“template_id”: “1134570”,
```

```
“similarity”: 0.3001076
},
{
“template_id”: “1134566”,
“similarity”: 0.3
}
]
```


Ошибки метода:

В случае возникновения ошибки при обработке запроса, Система возвращает вызывающей стороне коды ответов HTTP и описания ошибок в HTTP BODY, согласно таблице ниже.

Код ответа HTTP	Значение параметра «code»	Описание
400	BPE-002001	Неверный Content-Type HTTP-запроса
400	BPE-002002	Неверный метод HTTP-запроса
400	BPE-002004	Не удалось прочитать биометрический шаблон
400	BPE-002005	Неверный Content-Type части multiparted HTTP-запроса
400	BPE-00005	Не удалось прочитать метаданные
400	BPE-00502	Запрос не содержит обязательных данных {название данных/параметров}
400	BPE-00506	Недопустимое значение параметра {название параметра}
500	BPE-001001	Внутренняя ошибка биометрического процессора

Метод «Идентификация» - identify

Является объединением методов extract и match. Метод извлекает биометрический шаблон из БО и сравнивает полученный биометрический шаблон с БКШ в базе данных Поставщика БП.

Тип контента HTTP-запроса:

− Content-Type: multipart/form-data

− X-Request-ID: <string> - Идентификатор запроса (UUID)

Вызов метода: POST /*{версия}*/*{pathPrefix}*/identify

Входные параметры:

Наименование параметра	Значение	Описание
Часть multipart для передачи БДн		
<modality>	часть multipart	Обязательный параметр. Содержит биометрический образец для поиска схожих с ним БШ. modality может принимать значения: • photo • audio Заголовочные поля: • Content-Disposition: form-data; name="fieldName"; filename="filename.jpg", Content-Type: • «image/jpeg» • «image/png» • «audio/wav»
Часть multipart для передачи информации о пользователе		
metadata	часть multipart	Обязательный параметр. Часть multipart для передачи параметров поиска. Заголовочные поля: • Content-Disposition: form-data; name="metadata" • Content-Type: application/json
threshold	Number	Обязательный параметр. Пороговое значение степени схожести, может принимать только дробные значения от 0.0 до 1.0 включительно. Результаты, у которых степень схожести ниже, чем значение порогового значения, не нужны в выдаче.
limit	Number	Обязательный параметр. Максимальное количество записей в выдаче. Может принимать значение от 1 до 50 (включительно).

В заголовке части запроса с именем "metadata" может приходить кодировка текста charset=UTF-8 (Пример: Content-Type: application/json; **charset=UTF-8**). В этом случае данный параметр необходимо игнорировать.

Пример запроса:

```
POST /v1/{pathPrefix}/identify HTTP/1.1
Content-Type: multipart/form-data; boundary=f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
X-Request-ID: 4896c91b-9e61-3129-87b6-8aa299028058
Content-Length: {длина тела сообщения}


--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="photo"; filename="Makar.jpg"
Content-Type: image/jpeg

{Поток байт биометрического образца}
--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="metadata"
Content-Type: application/json
{
"threshold": 0.3,
"limit": 5
}
--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH--
```

Успешный ответ метода:

В случае успешного ответа, метод возвращает HTTP-ответ 200 OK, в HTTP BODY включены наборы прошедших пороговые значения степени схожести пар идентификаторов и значения степени схожести.

БП выдаёт найденный(е) идентификатор(ы) биометрического(их) шаблона(ов) и степень схожести.

В ответе может быть возвращен более чем один набор значений в случае нахождения нескольких совпадений выше порогового значения.

Набор совпадений должен быть отсортирован по убыванию степени схожести.

Если не найдено ни одного совпадения, то в ответ должен прийти пустой массив.

Если в базе БП отсутствуют записи, то в ответ должен прийти пустой массив.

Выходные параметры:

Наименование параметра	Значение	Описание
template_id	string	Обязательный параметр. Внутренний (ЕБС) идентификатор БКШ.
similarity	Number	Обязательный параметр. Степень схожести с поступившим на сравнение биометрическим шаблоном, может принимать только дробные значения от 0.0 до 1.0 включительно.

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
[
{
"template_id": "1134569",
"similarity": 0.6
},
{
"template_id": "1134567",
"similarity": 0.54
},
{
"template_id": "1134568",
"similarity": 0.45
},
{
"template_id": "1134570",
"similarity": 0.3001
},
{
"template_id": "1134566",
"similarity": 0.3
}
]
```

Ошибки метода:

В случае возникновения ошибки при обработке запроса, Система возвращает вызывающей стороне коды ответов HTTP и описания ошибок в HTTP BODY, согласно таблице ниже.

Код ответа HTTP	Значение параметра «code»	Описание
400	BPE-002001	Неверный Content-Type HTTP-запроса
400	BPE-002002	Неверный метод HTTP-запроса
400	BPE-002003	Не удалось прочитать биометрический образец
400	BPE-002005	Неверный Content-Type части multiparted HTTP-запроса
400	BPE-00005	Не удалось прочитать метаданные
400	BPE-00502	Запрос не содержит обязательных данных {название данных/параметров}
400	BPE-00506	Недопустимое значение параметра {название параметра}
400	BPE-003001	Не удалось извлечь биометрический шаблон
400	BPE-003002	Сообщение отличается в зависимости типа контента: • audio: На биометрическом образце отсутствует голос • image: На биометрическом образце отсутствует лицо
400	BPE-003003	Только для типа контента image. На биометрическом образце присутствует более, чем одно лицо * Для определения лица следует руководствоваться следующими требованиями: Расстояние между центрами глаз (IOD - inter-ocular distance): • IOD > 16px • IOD > 5% от ширины кадра Оба условия накладываются одновременно. Все, что не соответствует предъявленным требованиям, не должно считаться лицом.
500	BPE-001001	Внутренняя ошибка биометрического процессора

ПРИЛОЖЕНИЕ Б. Описание интеграции БП обнаружения витальности с ЕБС

Общее описание API БП обнаружения витальности

Интеграция подключаемых БП обнаружения витальности в составе ЕБС с Подсистемой детектирования подделок должна быть реализована посредством REST API. Архитектура REST (Representational State Transfer) подразумевает наличие клиент-серверной архитектуры. Клиент (здесь и далее в контексте описания REST API - Подсистема детектирования подделок) инициирует запросы к Серверу (здесь и далее в контексте описания REST API – подключаемый БП обнаружения витальности), сервер обрабатывает их и возвращает ответ.

Каждый поставщик БП должен реализовать REST API согласно спецификации, приведенной ниже. REST API определяет набор методов, к которым Подсистема детектирования подделок может совершать запросы и получать ответы. Взаимодействие происходит по протоколу HTTP. Обязательна поддержка постоянных соединений (keep-alive).

Список используемых методов REST API, а также необходимые параметры и возвращаемые значения, приведены ниже в данном документе.

Каждый подключаемый БП обнаружения витальности, предоставляемый Поставщиком БП, должен поддерживать метод сбора БО, который согласовывается и регистрируется в ЕБС на этапе подключения БП обнаружения витальности. На текущий момент БП активного обнаружения витальности должен принимать на вход запись видео (файл видео), полученный с клиентского приложения ЕБС по сбору БО (web-форма, мобильное приложение). БП активного обнаружения витальности применяется для следующих мнемоник: "move-instructions" и "text-instructions".

На текущий момент БП пассивного обнаружения витальности должен принимать на вход фотоизображение лица, аудиозапись голоса или видеофайлы, полученные с клиентского приложения ЕБС по сбору БО (web-форма, мобильное приложение). БП пассивного обнаружения витальности применяется для мнемоники "passive-instructions".

Точка доступа к API

Базовый URL доступа к API:

http://{hostname}/{version}/{pathPrefix}/liveness/

Где:

− *{hostname}* – имя хоста и порт сервера, развернутого БП обнаружения витальности;

− *{version}* – используемая версия API подключаемого БП обнаружения витальности; формат версии в пути HTTP-запросов: префикс «v» и целое число; актуальная версия API подключаемых БП обнаружения витальности: «v1».

− *{pathPrefix}* – необязательная составная часть пути URL к API подключаемого БП обнаружения витальности. Может использоваться, например, при установке на сервер БП

обнаружения витальности, предоставляющих несколько типов действий обнаружения витальности. Должна быть реализована возможность настройки параметра URL «pathPrefix» через переменные окружения, в том числе с возможностью задания пустых значений.

Пример запроса[5]:

```
GET /v1/liveness/health HTTP/1.1
     Host: liveness1.ebs.ru
```

Поддерживаемые в запросах методы HTTP и типы контента

БП обнаружения витальности должен поддерживать следующие методы HTTP:

− GET;

− POST.

БП обнаружения витальности должен поддерживать следующие типы контента запроса (HTTP-заголовок «Content-Type»): «multipart/form-data».

Входные параметры метода передаются в виде строки запроса[6] (часть URL после знака «?», разделитель параметров — знак «&») с передаваемыми на сервер параметрами при использовании метода GET, либо в теле POST-запроса. В случае GET-запроса, параметры должны быть закодированы с помощью URL Encoding[7], т.к. для URL доступны только символы латинского алфавита. При наличии тела запроса (метод POST), его содержимое (входные параметры метода) должно быть передано в формате JSON[8].

Если тип контента POST-запроса - «multipart/form-data», то каждый входной параметр метода передается как отдельная часть составного содержимого HTTP-запроса и следует правилам для составных MIME-данных в соответствии с RFC 2045.

Порядок заголовков внутри части multipart - запроса и порядок самих частей multipart - запроса может быть изменен. Каждая часть должна содержать:

− заголовочное поле «Content-Disposition», имеющее значение «form-data»;

− атрибут «name» поля «Content-Disposition», имеющий значение, равное наименованию входного параметра (см. ниже таблицы с описанием входных параметров соответствующих методов);

- параметр «filename» поля «Content-Disposition» не является обязательным для передачи в запросе. В случае наличия параметра «filename», отсутствие в значении параметра расширения не должно завершаться с ошибкой. Вызов метода без параметра «filename» не должен завершаться с ошибкой;

5 Здесь и далее в примерах запросов и ответов часть HTTP-заголовков не приводится для краткости.

6 Согласно RFC 3984, раздел 3.4

7 Согласно RFC 3986, раздел 2.1

8 Согласно RFC 7159

– заголовочное поле «Content-Type», принимающая значение в зависимости от контента, передаваемого в части:

– выходные параметры метода: «application/json»;

– БО – файл с клиентского приложения ЕБС для обнаружения витальности: «video/mov», «image/jpeg», «image/png», «audio/wav».

Следует отметить, что boundary (граница) — это последовательность байтов, которая не должна встречаться внутри закодированного представления данных части.

В каждом HTTP-запросе присутствует набор обязательных параметров, но могут быть определены дополнительные параметры, требуемые только для конкретного метода. Текстовые значения параметров передаются в кодировке UTF-8.

Используемые коды ответов HTTP

Используемые Системой коды ответов HTTP приведены в таблице ниже.

Коды ответа	Описание
200 OK	Вызов метода завершился успешно. Ответ Сервера включен в HTTP BODY.
400 Bad Request	Вызов метода завершился с ошибкой на стороне Клиента. Код ошибки включен в HTTP BODY.
500 Internal Server Error	Вызов метода завершился с ошибкой на стороне Сервера. Код ошибки включен в HTTP BODY.

Все успешные ответы:

– содержат код ответа HTTP 200;

– возвращают JSON объект со значениями выходных параметров метода в HTTP BODY. Тип контента - «application/json».

Все ответы с ошибкой:

– содержат код ответа HTTP 400 или 500;

– возвращают JSON объект с описанием ошибки в HTTP BODY. Тип контента «application/json».

Пример ответа с ошибкой:

```
HTTP/1.1 500 Internal Server Error
Content-Type: application/json; charset=UTF-8

{
        "code": "LDE-001001",
        "message": "Внутренняя ошибка БП обнаружения витальности"
}
```

Методы API подключаемых БП обнаружения витальности

Метод «Обнаружение витальности» - detect

Метод получения БО на обнаружение витальности (liveness detection) и возврата результата обнаружения витальности.

Тип контента HTTP-запроса: «multipart/form-data».

Вызов метода: POST /{версия}/{pathPrefix}/liveness/detect

Входные параметры:

Наименование параметра	Значение	Описание
metadata	JSON-object	Обязательный параметр. Часть multipart для передачи описания применённого метода обнаружения витальности. Перечень возможных методов обнаружения витальности приведен в разделе «Перечень поддерживаемых методов обнаружения витальности».
metadata.mnemonic	String	Обязательный параметр. Содержит мнемонику согласованного метода обнаружения витальности (тип действия). Возможные варианты: – move-instructions; – text-instructions; – passive-instructions;
metadata.actions	list <actions>	Обязательный параметр. Содержит перечень и описание необходимых действий метода обнаружения витальности на стороне клиентского приложения.
actions.type	String	Обязательный параметр. Содержит описание совершаемого Пользователем действия
actions.duration	Long	Обязательный параметр. Содержит фактическое время, которое потребовалось на выполнение действия Пользователем (в миллисекундах)
actions.message	String	Обязательный параметр.

Наименование параметра	Значение	Описание
		Содержит инструкцию, описывающую действие, предлагаемое к совершению Пользователем
actions.text	String	Обязательный параметр. Используется для действия «numbers-digits». Содержит кодовую последовательность (неповторяющиеся цифры от 0 до 9) в текстовой форме, предлагаемую к прочтению Пользователем
bio_sample	String	Обязательный параметр. Часть multipart с БО, снятым с устройства Пользователя определенным ранее методом сбора. «Content-Type»: «video/mov», «image/jpeg», «image/png», «audio/wav».

Для Content-Type: audio/wav применяется:

- кодек сжатия аудио "codec_sound": "pcm_s16le",
- параметр "count_channel": "1",
- контейнер "format_sound": "wav"

Для Content-Type: video/mov применяется:

- максимальный размер видеофайла “video-size”: 209715200 байт
- максимальная длительность видео “video-length-sec”: 120 с.
- максимальное разрешение “video-width-px”: 6000 px
- максимальное разрешение “video-height-px”: 6000 px
- рекомендуемое минимальное разрешение[9]: 1280 x 720 (для горизонтальной ориентации) или 720 x 1280 (для вертикальной ориентации)
- минимальная длительность видео: 2c
- кодеки сжатия видео: h264, mjpeg, hevc, mpeg1video, mpeg2video, vp8, vp9, flv, mpeg4, msmpeg4v2, msmpeg4v3, text, rawvideo

Пример запроса move-instructions:

```
POST /v1/liveness/detect HTTP/1.1
Content-Type: multipart/form-data; boundary=f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH

----f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="metadata"
```

9 Также принимается разрешение 640 x 480 или ниже (для горизонтальной ориентации) или 480 x 640 (для вертикальной ориентации).

```
Content-Type: application/json

{
  "mnemonic": "move-instructions",
  "actions":[{
    "type": "BLINK",
    "duration": 6000,
    "message": "Пожалуйста, моргните"
  }, {
    "type": "SMILE",
    "duration": 6000,
    "message": "Пожалуйста, улыбнитесь"
  }]
}

----f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="bio_sample"
Content-Type: video/mov

{Поток байт БО}
----f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH----
```

Пример запроса text-instructions:

```
POST /v1/liveness/detect HTTP/1.1
Content-Type: multipart/form-data; boundary=f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH

----f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="metadata"
Content-Type: application/json

{
  "mnemonic": "text-instructions",
  "actions": [{
    "type": "numbers-digits",
    "duration": 7000,
    "message": "Произнесите цифры:",
    "text": "один два три четыре пять"
  }]
}

----f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="bio_sample"
```

```
Content-Type: video/mov
{Поток байт БО}
----f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH----
```

Пример запроса passive-instructions при Content-Type «image/jpeg» (или «image/png»):

```
POST /v1/liveness/detect HTTP/1.1
Content-Type: multipart/form-data; boundary=f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH

----f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="metadata"
Content-Type: application/json

{
  "mnemonic": "passive-instructions",
  "actions": [{
    "type": "photo-type",
    "duration": 0,
    "message": "Посмотрите ровно в камеру"
  }]
}

----f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="bio_sample"
Content-Type: image/jpeg
{Поток байт Биометрического образца}
----f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH----
```

Пример запроса passive-instructions при Content-Type «video/mov»:

```
POST /v1/liveness/detect HTTP/1.1
Content-Type: multipart/form-data; boundary=f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH

----f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="metadata"
Content-Type: application/json

{
  "mnemonic": "passive-instructions",
  "actions": [{
    "type": "photo-type",
    "duration": 6000,
    "message": "Посмотрите ровно в камеру"
  }]
```

```
}

----f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="bio_sample"
Content-Type: video/mov
{Поток байт Биометрического образца}
----f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH----
```

Пример запроса passive-instructions при Content-Type: audio/wav:

```
POST /v1/liveness/detect HTTP/1.1
Content-Type: multipart/form-data; boundary=f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH

----f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="metadata"
Content-Type: application/json

{
  "mnemonic": "passive-instructions",
  "actions": [{
    "type": "audio-type",
    "duration": 10000,
    "message": "Произнесите «два восемь»"
  }]
}

----f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="bio_sample"
Content-Type: audio/wav
{Поток байт Биометрического образца}
----f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH----
```

Успешный ответ метода:

В случае успешного ответа метод возвращает результат обнаружения витальности.

Тип контента HTTP-ответа: «application/json».

Выходные параметры:

Наименование параметра	Значение	Описание
passed	Boolean	Обязательный параметр. Результат проверки bio_sample на витальность в соответствии с указанными в запросе actions.

score	Float	Обязательный параметр. Количественная оценка результата проверки bio_sample на витальность в соответствии с указанными в запросы actions.
results	Array	Необязательный параметр Массив результатов проверки витальности на каждую атомарную проверку из полученных в запросе.
results.type	String	Тип конкретной атомарной проверки витальности, в соответствии с порядком, из полученных в запросе.
results.passed	Boolean	Признак успешности атомарной проверки витальности.

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
  "passed": false,
  "score": 0.1,
  "results": [
    {
      "type": "SMILE",
      "passed": false
    }
  ]
}
```

Ошибки метода:

В случае возникновения ошибки при обработке запроса Система возвращает вызывающей стороне коды ответов HTTP и описания ошибок в HTTP BODY, согласно таблице ниже.

Код ответа HTTP	Значение параметра «code»	Описание
400	LDE-002001	Неверный Content-Type HTTP-запроса
400	LDE-002002	Неверный метод HTTP-запроса
400	LDE-002004	Не удалось прочитать биометрический образец. [Подробное описание причины (при наличии)]
400	LDE-002005	Неверный Content-Type части multiparted HTTP-запроса

Код ответа HTTP	Значение параметра «code»	Описание
400	LDE-003002	Сообщение отличается в зависимости типа контента: • audio: На биометрическом образце отсутствует голос • image: На биометрическом образце отсутствует лицо • video: На биометрическом образце отсутствует лицо
400	LDE-003003	Только для типов контента image и video На биометрическом образце присутствует более, чем одно лицо * Для определения лица следует руководствоваться следующими требованиями: Расстояние между центрами глаз (IOD - inter-ocular distance): • IOD > 16px • IOD > 5% от ширины кадра Оба условия накладываются одновременно. Все, что не соответствует предъявленным требованиям, не должно считаться лицом.
400	LDE-003004	Не удалось прочитать metadata
500	LDE-001001	Внутренняя ошибка БП обнаружения витальности

Метод «Проверка состояния БП обнаружения витальности» - health

Метод возвращает текущее состояние работоспособности БП обнаружения витальности. Реализация метода должна обеспечить проверку всех зависимостей БП обнаружения витальности, необходимых для работоспособности методов API подключаемых БП обнаружения витальности путем вызова всех соответствующих внутренних функций на контрольных примерах. Метод должен возвращать положительный ответ только в случае успешного прохождения проверки всех внутренних функций.

Подсистема мониторинга ЕБС обрабатывает ответ метода следующим образом:

− БП обнаружения витальности работает корректно: в случае получения от метода успешного ответа (см. таблицу ниже), параметр «status» равен 0;

− БП обнаружения витальности неработоспособен: в случае получения от метода успешного ответа (см. таблицу ниже), параметр «status» равен 3;

− REST API БП обнаружения витальности неработоспособно (внутренняя ошибка): в случае получения от метода ошибки с HTTP-кодом 500;

– анализирует время отклика метода для сбора статистики производительности и утилизации, генерации событий мониторинга при превышении пороговых значений.

Вызов метода: GET /{версия}/{pathPrefix}/liveness/health

Входные параметры: Отсутствуют.

Пример запроса:

```
GET /v1/liveness/health HTTP/1.1
     Host: liveness1.ebs.ru
```

Успешный ответ метода:

В случае успешного ответа, метод возвращает сообщение со следующими параметрами.

Тип контента HTTP-ответа: «application/json».

Выходные параметры:

Наименование параметра	Значение	Описание
status	Number	Обязательный параметр. Возможные значения параметра: 0: БП обнаружения витальности работает корректно (данный статус обязателен для реализации); 1: Minor – в работе БП обнаружения витальности возникло событие низкой критичности (данный статус необязателен для реализации); 2: Major – в работе БП обнаружения витальности возникло событие высокой критичности (данный статус необязателен для реализации); 3: Critical – БП обнаружения витальности неработоспособен (данный статус обязателен для реализации)
message	String	Необязательный параметр. Строка длиной не более 128 символов. Если значение параметра «status» не равно нулю (1-3), то рекомендуется передать данный параметр и включить в него текст с описанием возникшего события. Данное событие будет показано сотруднику дежурной смены в подсистеме мониторинга.

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
  "status": 3,
  "message": "Функция обнаружения витальности не ответила на контрольный запрос"
}
```

Ошибки метода:

В случае возникновения ошибки при обработке запроса, Система возвращает вызывающей стороне коды ответов HTTP и описания ошибок в HTTP BODY, согласно таблице ниже.

Код ответа HTTP	Значение параметра «code»	Описание
400	LDE-002002	Неверный метод HTTP-запроса
400	LDE-002006	Неверный запрос
500	LDE-001002	Внутренняя ошибка REST API БП обнаружения витальности

Перечень поддерживаемых методов обнаружения витальности

Метод обнаружения витальности «move-instructions»

Описание действия данного метода состоит из следующих параметров:

− type – описание совершаемого Пользователем действия;

− duration – фактическое время, которое потребовалось на выполнение действия Пользователем (в миллисекундах);

− message – инструкция, описывающая действие, предлагаемое к совершению Пользователем.

Спецификация параметров:

Описание действия (type)	Время отображения (duration)	Инструкция Пользователю (message)
TURN_RIGHT		«Пожалуйста, посмотрите направо»
TURN_LEFT		«Пожалуйста, посмотрите налево»
TURN_UP		«Пожалуйста, посмотрите вверх»
TURN_DOWN		«Пожалуйста, посмотрите вниз»

Описание действия (type)	Время отображения (duration)	Инструкция Пользователю (message)
SMILE		«Пожалуйста, улыбнитесь»
BLINK		«Пожалуйста, моргните»
DISTORTION		«Пожалуйста, отдалите телефон и не спеша приближайте его к лицу»
RAISE_EYEBROW		«Пожалуйста, поднимите брови»

Пример метаданных «move-instructions»:

```
{
  "mnemonic": "move-instructions",
  "actions":[{
    "type": "BLINK",
    "duration": 6000,
    "message": "Пожалуйста, моргните"
  }, {
    "type": "SMILE",
    "duration": 6000,
    "message": "Пожалуйста, улыбнитесь"
  }]
}
```

Метод обнаружения витальности «text-instructions»

Описание действия данного метода состоит из следующих параметров:

– type – описание произносимого Пользователем текста (неповторяющиеся цифры от 0 до 9);

– duration – фактическое время, которое потребовалось на выполнение действия Пользователем (в миллисекундах);

– message – инструкция, описывающая действие, предлагаемое к совершению Пользователем;

– text - кодовая последовательность (неповторяющиеся цифры от 0 до 9) в текстовой форме, предлагаемую к прочтению Пользователем.

Пример метаданных «text-instructions»:

```
{
  "mnemonic": "text-instructions",
  "actions": [
```

```
    {
      "type": "numbers-digits",
      "duration": 7000,
      "message": "Произнесите цифры:",
      "text": "один два три четыре пять"
    }]
}
```

Метод обнаружения витальности «passive-instructions»

Описание действия данного метода состоит из следующих параметров:

– type – описание совершаемого Пользователем действия;

– duration – фактическое время, которое потребовалось на выполнение действия Пользователем (в миллисекундах);

– message – инструкция, описывающая действие, предлагаемое к совершению Пользователем;

Пример метаданных «passive-instructions» при Content-Type: audio/wav:

```
{
  "mnemonic": "passive-instructions",
  "actions": [{
    "type": "audio-type",
    "duration": 10000,
    "message": "Произнесите «два восемь»"
  }]
}
```

Пример метаданных «passive-instructions» при Content-Type: «image/jpeg» (или «image/png»):

```
{
  "mnemonic": "passive-instructions",
  "actions": [{
    "type": "photo-type",
    "duration": 0,
    "message": "Посмотрите ровно в камеру"
  }]
}
```

Пример метаданных «passive-instructions» при Content-Type: «video/mov»:

```
{
  "mnemonic": "passive-instructions",
  "actions": [{
```

```
    "type": "photo-type",
    "duration": 6000,
    "message": "Посмотрите ровно в камеру"
  }]
}
```

ПРИЛОЖЕНИЕ В. Описание интеграции БП биометрической верификации с ЕБС

Общее описание API БП

Интеграция БП в составе ЕБС с Подсистемой балансировки запросов должна быть реализована посредством REST API. Архитектура REST (Representational State Transfer) подразумевает наличие клиент-серверной архитектуры. Клиент (здесь и далее в контексте описания REST API - Подсистема балансировки запросов) инициирует запросы к Серверу (здесь и далее в контексте описания REST API - БП), сервер обрабатывает их и возвращает ответ.

Каждый поставщик БП должен реализовать REST API, согласно спецификации, приведенной ниже.

REST API определяет набор методов, к которым Подсистема балансировки запросов может совершать запросы и получать ответы. Взаимодействие происходит по протоколу HTTP. Обязательна поддержка постоянных соединений (keep-alive).

Список используемых методов REST API, а также необходимые параметры и возвращаемые значения, приведены ниже в данном документе.

Версия API

Актуальная версия API БП: «v1».

Формат версии: префикс «v» и целое число.

Точка доступа к API

Базовый URL доступа к API:

http://{hostname}/{version}/{pathPrefix}/

Где:

− *{hostname}* – имя хоста и порт сервера развернутого БП;

− *{version}* – используемая версия API БП. Формат версии в пути HTTP запросов: префикс «v» и целое число; актуальная версия API подключаемых БП биометрической верификации: «v1».

− *{pathPrefix}* – необязательная составная часть пути URL к API БП. Может использоваться, например, при установке на сервер БП нескольких модальностей. Должна быть реализована возможность настройки параметра «pathPrefix» через переменные окружения, в том числе с возможностью задания пустых значений.

Пример запроса[10]:

```
GET /v1/health HTTP/1.1

     Host: bioproc1.ebs.ru
```

Поддерживаемые в запросах методы HTTP и типы контента

Сервер должен поддерживать следующие методы HTTP:

− GET;

− POST.

Сервер должен поддерживать следующие типы контента запроса (HTTP-заголовок «Content-Type»):

− «multipart/form-data»;

− «application/octet-stream»;

− «image/jpeg» или «image/png» (для БП модальности «Фотоизображение лица»);

− «audio/wav» (для БП модальности «Аудиозапись голоса»).

Если тип контента запроса - «application/json», то входные параметры метода передаются в теле POST-запроса в формате JSON.

Если тип контента запроса - «multipart/form-data», то каждый входной параметр метода передается как отдельная часть составного содержимого HTTP-запроса и следует правилам для составных MIME-данных в соответствии с RFC 2045.

Порядок заголовков внутри части multipart - запроса и порядок самих частей multipart - запроса может быть изменен. Каждая часть должна содержать:

− заголовочное поле «Content-Disposition», имеющее значение «form-data»;

− атрибут «name» поля «Content-Disposition», имеющий значение, равное наименованию входного параметра (см. ниже таблицы с описанием входных параметров соответствующих методов);

− параметр «filename» поля «Content-Disposition» не является обязательным для передачи в запросе. В случае наличия параметра «filename», отсутствие в значении параметра расширения не должно завершаться с ошибкой. Вызов метода без параметра «filename» не должен завершаться с ошибкой;

− заголовочное поле «Content-Type», принимающая значение в зависимости от контента, передаваемого в части:

1. БО, модальность «Фотоизображение лица»: «image/jpeg» или «image/png»;
2. БО, модальность «Аудиозапись голоса»: «audio/wav»;
3. биометрический шаблон: «application/octet-stream».

10 Здесь и далее в примерах запросов и ответов часть HTTP-заголовков не приводится для краткости.

Следует отметить, что boundary (граница) — это последовательность байтов, которая не должна встречаться внутри закодированного представления данных части.

Если тип контента запроса - «application/octet-stream», «image/jpeg», «image/png», «audio/wav», то метод принимает на вход только один бинарный параметр.

В каждом HTTP-запросе присутствует набор обязательных параметров, но могут быть определены дополнительные параметры, требуемые только для конкретного метода. Текстовые значения параметров передаются в кодировке UTF-8.

Используемые коды ответов HTTP

Используемые Системой коды ответов HTTP приведены в таблице ниже.

Коды ответа	Описание
200 OK	Вызов метода завершился успешно. Ответ Сервера включен в HTTP BODY.
400 Bad Request	Вызов метода завершился с ошибкой на стороне Клиента. Код ошибки включен в HTTP BODY.
500м Internal Server Error	Вызов метода завершился с ошибкой на стороне Сервера. Код ошибки включен в HTTP BODY.

Все успешные ответы:

1. содержат код ответа HTTP 200;

2. возвращают:

- JSON объект со значениями выходных параметров метода в HTTP BODY. Тип контента - «application/json»;
- бинарный контент (например, если метод возвращает биометрический шаблон) с указанием типа контента в HTTP-заголовке «Content-Type»;
- составной контент (например, если метод возвращает степень схожести и биометрический шаблон), с указанием типа контента «multipart/form-data» в HTTP-заголовке «Content-Type».

Все ответы с ошибкой:

- содержат код ответа HTTP 400 или 500;
- возвращают JSON объект с описанием ошибки в HTTP BODY. Тип контента «application/json».

Пример ответа с ошибкой:

```
HTTP/1.1 500 Internal Server Error
Content-Type: application/json; charset=UTF-8
{
        "code": "BPE-001001",
        "message": "Внутренняя ошибка биометрического процессора"
```

```
}
```

Методы API БП

Метод «Извлечение биометрического шаблона» - extract

Метод предназначен для извлечения биометрического шаблона из БО.

Тип контента HTTP-запроса:

− «image/jpeg» или «image/png» (для БП модальности «Фотоизображение лица»);

− «audio/wav» (для БП модальности «Аудиозапись голоса»).

Вызов метода: POST /*{версия}*/*{pathPrefix}*/extract

Входные параметры:

Наименование параметра	Значение	Описание
–[11]	Stream	Обязательный параметр. БО для извлечения биометрического шаблона.[12] В заголовочном поле «Content-Type» должен быть указан тип контента соответствующей модальности БО.

Пример запроса (модальность «Фотоизображение лица»):

```
POST /v1/{pathPrefix}/extract HTTP/1.1
Content-Type: image/jpeg

{Поток байт БО}
```

Успешный ответ метода:

В случае успешного ответа, метод возвращает биометрический шаблон.

Тип контента HTTP-ответа: «application/octet-stream».

Выходные параметры:

Наименование параметра	Значение	Описание
– [13]	Stream	Биометрический шаблон, полученный из БО.

11 Здесь и далее наименование параметра отсутствует и не передается в запросе

12 Здесь и далее порядок байтов в БО и шаблонах «little-endian»

13 Здесь и далее наименование параметра отсутствует и не передается в ответе

Пример:

```
HTTP/1.1 200 OK
Content-Type: application/octet-stream

{Поток байт биометрического шаблона}
```

Ошибки метода:

В случае возникновения ошибки при обработке запроса, Система возвращает вызывающей стороне коды ответов HTTP и описания ошибок в HTTP BODY, согласно таблице ниже.

Код ответа HTTP	Значение параметра «code»	Описание
400	BPE-002001	Неверный Content-Type HTTP-запроса
400	BPE-002002	Неверный метод HTTP-запроса
400	BPE-002003	Не удалось прочитать биометрический образец
400	BPE-003001	Не удалось извлечь биометрический шаблон
400	BPE-003002	Сообщение отличается в зависимости типа контента: • audio: На биометрическом образце отсутствует голос • image: На биометрическом образце отсутствует лицо
400	BPE-003003	Только для типа контента image. На биометрическом образце присутствует более, чем одно лицо * Для определения лица следует руководствоваться следующими требованиями: Расстояние между центрами глаз (IOD - inter-ocular distance): • IOD > 16px • IOD > 5% от ширины кадра Оба условия накладываются одновременно. Все, что не соответствует предъявленным требованиям, не должно считаться лицом.
500	BPE-001001	Внутренняя ошибка биометрического процессора

Метод «Сравнение биометрических шаблонов» - compare

Метод предназначен для сравнения двух биометрических шаблонов: шаблона, извлеченного из БО, и БКШ.

Тип контента HTTP-запроса: «multipart/form-data».

Вызов метода: POST /*{версия}*/*{pathPrefix}*/compare

Входные параметры:

Наименование параметра	Значение	Описание
bio_feature	Stream	Обязательный параметр. Биометрический шаблон, полученный из БО при биометрической верификации.
bio_template	Stream	Обязательный параметр. БКШ, с которым производится сравнение.

Пример запроса:

```
POST /v1/{pathPrefix}/compare HTTP/1.1
Content-Type: multipart/form-data; boundary=f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Length: {длина тела сообщения}

--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="bio_feature"
Content-Type: application/octet-stream

{Поток байт шаблона, полученный из БО при биометрической верификации}

--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="bio_template"
Content-Type: application/octet-stream

{Поток байт БКШ}
--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH--
```

Успешный ответ метода:

В случае успешного ответа, метод возвращает результат сравнения двух биометрических шаблонов.

Тип контента HTTP-ответа: «application/json».

Выходные параметры:

Наименование параметра	Значение	Описание
score	Number	Обязательный параметр. Степень схожести.

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
  "score": 0.8
}
```

Ошибки метода:

В случае возникновения ошибки при обработке запроса, Система возвращает вызывающей стороне коды ответов HTTP и описания ошибок в HTTP BODY, согласно таблице ниже.

Код ответа HTTP	Значение параметра «code»	Описание
400	BPE-002001	Неверный Content-Type HTTP-запроса
400	BPE-002002	Неверный метод HTTP-запроса
400	BPE-002004	Не удалось прочитать биометрический шаблон
400	BPE-002005	Неверный Content-Type части multiparted HTTP-запроса
500	BPE-001001	Внутренняя ошибка биометрического процессора

Метод «Биометрическая верификация» - verify

Является объединением двух методов, описанных выше. Метод извлекает биометрический шаблон из БО и сравнивает полученный биометрический шаблон с БКШ, переданным в качестве параметра.

Тип контента HTTP-запроса: «multipart/form-data».

Вызов метода: POST /*{версия}*/*{pathPrefix}*/verify

Входные параметры:

Наименование параметра	Значение	Описание
bio_template	Stream	Обязательный параметр. БКШ, с которым производится сравнение.
sample	Stream	Обязательный параметр. БО для извлечения биометрического шаблона. В заголовочном поле «Content-Type» части составного типа контента должен быть указан тип контента соответствующей модальности БО (см. раздел «Поддерживаемые в запросах методы HTTP и типы контента»).

Пример запроса (модальность «Фотоизображение лица»):

```
POST /v1/{pathPrefix}/verify HTTP/1.1
Content-Type: multipart/form-data; boundary=f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Length: {длина тела сообщения}

--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="bio_template"
Content-Type: application/octet-stream

{Поток байт БКШ}

--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="sample"
Content-Type: image/jpeg

{Поток байт БО}
--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH--
```

Успешный ответ метода:

В случае успешного ответа, метод возвращает результат сравнения двух биометрических шаблонов и биометрический шаблон.

Тип контента HTTP-ответа: «multipart/form-data».

Выходные параметры:

Наименование параметра	Значение	Описание
score	Number	Обязательный параметр. Степень схожести.
bio_feature	Stream	Обязательный параметр. Биометрический шаблон, полученный из БО (входной параметр «sample»).

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: multipart/form-data; boundary=f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Length: {длина тела сообщения}

--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="score"
Content-Type: application/json
{
  "score": 0.8
}

--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH
Content-Disposition: form-data; name="bio_feature"
Content-Type: application/octet-stream

{Поток байт Биометрического шаблона}
--f3URHA_Xnhk0D8gW1iCGLPQk9_gjZr_ywsH--
```

Ошибки метода:

В случае возникновения ошибки при обработке запроса, Система возвращает вызывающей стороне коды ответов HTTP и описания ошибок в HTTP BODY, согласно таблице ниже.

Код ответа HTTP	Значение параметра «code»	Описание
400	BPE-002001	Неверный Content-Type HTTP-запроса
400	BPE-002002	Неверный метод HTTP-запроса
400	BPE-002003	Не удалось прочитать биометрический образец
400	BPE-002004	Не удалось прочитать биометрический шаблон

Код ответа HTTP	Значение параметра «code»	Описание
400	BPE-002005	Неверный Content-Type части multiparted HTTP-запроса
400	BPE-003001	Не удалось извлечь биометрический шаблон
400	BPE-003002	Сообщение отличается в зависимости типа контента: • audio: На биометрическом образце отсутствует голос • image: На биометрическом образце отсутствует лицо
400	BPE-003003	Только для типа контента image. На биометрическом образце присутствует более, чем одно лицо * Для определения лица следует руководствоваться следующими требованиями: Расстояние между центрами глаз (IOD - inter-ocular distance): • IOD > 16px • IOD > 5% от ширины кадра Оба условия накладываются одновременно. Все, что не соответствует предъявленным требованиям, не должно считаться лицом.
500	BPE-001001	Внутренняя ошибка биометрического процессора

Метод «Проверка состояния БП» - health

Метод возвращает текущее состояние работоспособности БП. Реализация метода должна обеспечить проверку всех зависимостей БП, необходимых для работоспособности методов API БП путем вызова всех соответствующих внутренних функций на контрольных примерах. Метод должен возвращать положительный ответ только в случае успешного прохождения проверки всех внутренних функций.

Подсистема мониторинга ЕБС обрабатывает ответ метода следующим образом:

− БП работает корректно: в случае получения от метода успешного ответа (см. таблицу ниже), параметр «status» равен 0;

− БП неработоспособен: в случае получения от метода успешного ответа (см. таблицу ниже), параметр «status» равен 3;

− REST API БП неработоспособно (внутренняя ошибка): в случае получения от метода ошибки с HTTP-кодом 500;

− анализирует время отклика метода для сбора статистики производительности и утилизации, генерации событий мониторинга при превышении пороговых значений.

Вызов метода: GET /{версия}/{pathPrefix}/health

Входные параметры:

Отсутствуют.

Пример запроса:

```
GET /v1/{pathPrefix}/health HTTP/1.1
     Host: bioproc1.ebs.ru
```

Успешный ответ метода:

В случае успешного ответа, метод возвращает сообщение со следующими параметрами.

Тип контента HTTP-ответа: «application/json».

Выходные параметры:

Наименование параметра	Значение	Описание
status	Number	Обязательный параметр. Возможные значения параметра: 0: биометрический процессор работает корректно (данный статус обязателен для реализации); 1: Minor – в работе биометрического процессора возникло событие низкой критичности (данный статус необязателен для реализации); 2: Major – в работе биометрического процессора возникло событие высокой критичности (данный статус необязателен для реализации); 3: Critical – биометрический процессор неработоспособен (данный статус обязателен для реализации);
message	String	Необязательный параметр. Строка длиной не более 128 символов. Если значение параметра «status» не равно нулю (1-3), то рекомендуется передать данный параметр и включить в него текст с описанием возникшего события. Данное событие будет показано сотруднику дежурной смены в подсистеме мониторинга.

Пример ответа:

```
HTTP/1.1 200 OK
```

```
Content-Type: application/json; charset=utf-8


{
  "status": 3,
  "message": "Функция сравнения биометрических шаблонов не ответила на контрольный запрос"
}
```

Ошибки метода:

В случае возникновения ошибки при обработке запроса, Система возвращает вызывающей стороне коды ответов HTTP и описания ошибок в HTTP BODY, согласно таблице ниже.

Код ответа HTTP	Значение параметра «code»	Описание
400	BPE-002006	Неверный запрос
500	BPE-001002	Внутренняя ошибка REST API биометрического процессора