

Модуль индексирования LUNA

Руководство администратора

v.5.67.0

Содержание

Глоссарий	9
1 Обзор	10
2 Основные положения	12
2.1 Индекс	12
2.1.1 Связь индекса с версией биометрического шаблона	13
2.1.2 Структура индекса	13
2.1.3 Процесс создания задачи на построение индекса	13
2.1.4 Процесс создания индекса	14
2.2 Сравнение	15
2.2.1 Запросы на сравнение	15
2.2.2 Процесс сравнения	15
3 Взаимодействие сервисов	16
4 Сервисы индексирования	19
4.1 Сервис Index Manager	19
4.1.1 Фоновые процедуры	19
4.1.2 Настройка перестроения и обновления индекса	19
4.1.3 Хранилище Index Manager	20
4.1.3.1 Работа с несколькими экземплярами	20
4.1.4 Запросы к сервису	21
4.2 Сервис Indexer	23
4.3 Сервис Indexed Matcher	24
4.3.1 Синхронизация меток сравнения в памяти	24
4.3.2 Перезагрузка индекса	25
4.3.3 Обновление индекса в памяти	26
4.3.4 Кэширование индекса	27
5 Плагин сравнения для Python Matcher Proxy	28
5.1 Описание работы плагина сравнения	28
5.2 Сложность запроса на сравнение	28
5.3 Поля target, выступающие в качестве сравнения	29
6 Мониторинг	30
6.1 Отправка данных в InfluxDB	30
6.1.1 Отправляемые данные	30

7	Диаграммы последовательностей	34
7.1	Диаграмма построения индекса	34
7.2	Диаграмма первичной загрузки индекса в память	36
7.3	Диаграмма сравнения биометрических шаблонов	37
7.4	Диаграмма перезагрузки индекса	39
7.5	Диаграмма обновления индекса	41
8	Использование Redis Sentinel	44
9	Ошибки API	45
9.1	Ошибки сервиса Indexer	45
9.1.1	Вернулся код ошибки 26100	45
9.1.2	Вернулся код ошибки 26101	45
9.1.3	Вернулся код ошибки 26103	45
9.1.4	Вернулся код ошибки 26104	46
9.1.5	Вернулся код ошибки 26105	46
9.1.6	Вернулся код ошибки 26106	46
9.1.7	Вернулся код ошибки 26107	47
9.1.8	Вернулся код ошибки 26108	47
9.1.9	Вернулся код ошибки 26109	47
9.2	Ошибки сервиса Index Manager	47
9.2.1	Вернулся код ошибки 26201	47
9.2.2	Вернулся код ошибки 26202	48
9.2.3	Вернулся код ошибки 26203	48
9.2.4	Вернулся код ошибки 26204	48
9.3	Ошибки сервиса Indexed Matcher	49
9.3.1	Вернулся код ошибки 26301	49
9.3.2	Вернулся код ошибки 26302	49
9.3.3	Вернулся код ошибки 26303	49
9.3.4	Вернулся код ошибки 26304	50
9.3.5	Вернулся код ошибки 26305	50
9.3.6	Вернулся код ошибки 26306	50
9.3.7	Вернулся код ошибки 26307	51
9.3.8	Вернулся код ошибки 26308	51
10	Описание параметров сервисов	52
10.1	Настройки сервиса Index Manager	52
10.1.1	Группа параметров LIM_MANAGER_LOGGER	52
10.1.1.1	log_level	52
10.1.1.2	log_time	52

10.1.1.3	log_to_stdout	52
10.1.1.4	log_to_file	52
10.1.1.5	folder_with_logs	53
10.1.1.6	max_log_file_size	53
10.1.1.7	multiline_stack_trace	53
10.1.1.8	format	53
10.1.2	Группа параметров LIM_MANAGER_INDEXING	54
10.1.2.1	indexer_origins	54
10.1.2.2	planning_period	54
10.1.2.3	planning_schedule	54
10.1.2.4	lookup_period	55
10.1.2.5	face_lists > min_indexing_list_size	55
10.1.2.6	face_lists > indexing_lists	55
10.1.2.7	ef_construction	56
10.1.2.8	rebuild_rules > default	56
10.1.2.9	rebuild_rules > max_removal_for_rebuild	56
10.1.3	Группа параметров LIM_MANAGER_HTTP_SETTINGS	56
10.1.3.1	request_timeout	56
10.1.3.2	response_timeout	57
10.1.3.3	request_max_size	57
10.1.3.4	keep_alive_timeout	57
10.1.4	Группа параметров LIM_MANAGER_DB	57
10.1.4.1	db_user	57
10.1.4.2	db_password	57
10.1.4.3	db_host	58
10.1.4.4	db_port	58
10.1.4.5	db_settings > connection_pool_size	58
10.1.4.6	db_number	58
10.1.4.7	sentinel > master_name	58
10.1.4.8	sentinel > sentinels	58
10.1.4.9	sentinel > user	59
10.1.4.10	sentinel > password	59
10.1.5	Группа параметров INFLUX_MONITORING	59
10.1.5.1	send_data_for_monitoring	59
10.1.5.2	use_ssl	59
10.1.5.3	organization	59
10.1.5.4	token	59
10.1.5.5	bucket	60
10.1.5.6	host	60

10.1.5.7	port	60
10.1.5.8	flushing_period	60
10.1.6	Группа параметров LUNA_FACES_ADDRESS	60
10.1.6.1	origin	60
10.1.6.2	api_version	61
10.1.7	Группа параметров LUNA_FACES_TIMEOUTS	61
10.1.7.1	connect	61
10.1.7.2	request	61
10.1.7.3	sock_connect	61
10.1.7.4	sock_read	61
10.1.8	Группа параметров LUNA_SERVICE_METRICS	62
10.1.8.1	enabled	62
10.1.8.2	metrics_format	62
10.1.8.3	extra_labels	62
10.1.9	Прочие	62
10.1.9.1	index_storage_type	62
10.1.9.2	index_storage_local	63
10.1.9.3	storage_time	63
10.1.9.4	lim_manager_active_plugins	63
10.1.9.5	default_face_descriptor_version	63
10.2	Настройки сервиса Indexed Matcher	64
10.2.1	Группа параметров LIM_MATCHING	64
10.2.1.1	ef_search	64
10.2.2	Группа параметров LIM_MATCHER_REFRESH	64
10.2.2.1	enabled	64
10.2.3	Группа параметров LIM_MATCHER_LOGGER	64
10.2.3.1	log_level	64
10.2.3.2	log_time	65
10.2.3.3	log_to_stdout	65
10.2.3.4	log_to_file	65
10.2.3.5	folder_with_logs	65
10.2.3.6	max_log_file_size	65
10.2.3.7	multiline_stack_trace	66
10.2.3.8	format	66
10.2.4	Группа параметров LIM_MATCHER_HTTP_SETTINGS	66
10.2.4.1	request_timeout	67
10.2.4.2	response_timeout	67
10.2.4.3	request_max_size	67
10.2.4.4	keep_alive_timeout	67

10.2.5	Группа параметров LIM_MATCHER_DB	67
10.2.5.1	db_user	67
10.2.5.2	db_password	68
10.2.5.3	db_host	68
10.2.5.4	db_port	68
10.2.5.5	db_settings > connection_pool_size	68
10.2.5.6	db_number	68
10.2.5.7	sentinel > master_name	68
10.2.5.8	sentinel > sentinels	69
10.2.5.9	sentinel > user	69
10.2.5.10	sentinel > password	69
10.2.6	Группа параметров INFLUX_MONITORING	69
10.2.6.1	send_data_for_monitoring	69
10.2.6.2	use_ssl	69
10.2.6.3	organization	70
10.2.6.4	token	70
10.2.6.5	bucket	70
10.2.6.6	host	70
10.2.6.7	port	70
10.2.6.8	flushing_period	70
10.2.7	Группа параметров LUNA_FACES_ADDRESS	70
10.2.7.1	origin	71
10.2.7.2	api_version	71
10.2.8	Группа параметров LUNA_FACES_TIMEOUTS	71
10.2.8.1	connect	71
10.2.8.2	request	71
10.2.8.3	sock_connect	71
10.2.8.4	sock_read	72
10.2.9	Группа параметров LUNA_LICENSES_ADDRESS	72
10.2.9.1	origin	72
10.2.9.2	api_version	72
10.2.10	Группа параметров LUNA_SERVICE_METRICS	72
10.2.10.1	enabled	72
10.2.10.2	metrics_format	73
10.2.10.3	extra_labels	73
10.2.11	Прочие	73
10.2.11.1	lim_matcher_cache	73
10.2.11.2	index_storage_type	73
10.2.11.3	index_storage_local	74

10.2.11.4	lim_matcher_active_plugins	74
10.2.11.5	default_face_descriptor_version	74
10.3	Настройки сервиса Indexer	75
10.3.1	Группа параметров LIM_INDEXER_LOGGER	75
10.3.1.1	log_level	75
10.3.1.2	log_time	75
10.3.1.3	log_to_stdout	75
10.3.1.4	log_to_file	75
10.3.1.5	folder_with_logs	76
10.3.1.6	max_log_file_size	76
10.3.1.7	multiline_stack_trace	76
10.3.1.8	format	76
10.3.2	Группа параметров INFLUX_MONITORING	77
10.3.2.1	send_data_for_monitoring	77
10.3.2.2	use_ssl	77
10.3.2.3	organization	77
10.3.2.4	token	77
10.3.2.5	bucket	78
10.3.2.6	host	78
10.3.2.7	port	78
10.3.2.8	flushing_period	78
10.3.3	Группа параметров LUNA_FACES_ADDRESS	78
10.3.3.1	origin	78
10.3.3.2	api_version	79
10.3.4	Группа параметров LUNA_FACES_TIMEOUTS	79
10.3.4.1	connect	79
10.3.4.2	request	79
10.3.4.3	sock_connect	79
10.3.4.4	sock_read	79
10.3.5	Группа параметров LIM_INDEXER_HTTP_SETTINGS	80
10.3.5.1	request_timeout	80
10.3.5.2	response_timeout	80
10.3.5.3	request_max_size	80
10.3.5.4	keep_alive_timeout	80
10.3.6	Группа параметров LUNA_SERVICE_METRICS	80
10.3.6.1	enabled	81
10.3.6.2	metrics_format	81
10.3.6.3	extra_labels	81

10.3.7	Прочие	81
10.3.7.1	index_storage_type	81
10.3.7.2	index_storage_local	81
10.3.7.3	lim_indexer_active_plugins	82
10.3.7.4	default_face_descriptor_version	82
10.4	Настройки плагина сравнения	83
10.4.1	Группа параметров LUNA_INDEXED_LIST_PLUGIN	83
10.4.1.1	redis_url	83
10.4.1.2	request_timeout	83

Глоссарий

Термин	Определение
Биометрический образец	Изображение, на котором присутствует лицо или тело и которые соответствуют стандартам VisionLabs и другим стандартам.
Биометрический шаблон	Набор уникальных признаков, полученных от биометрического образца. БШ не считается персональными данными. С его помощью невозможно восстановить исходное лицо.
Индекс	Сущность LUNA Index Module (LIM), содержащая набор БШ, полученных из предоставленных пользователем изображений и развернутых вместе для приблизительного сравнения. Индекс генерируется сервисом Indexer после получения задачи от сервиса Index Manager.
Метка для сравнения	Сущность LIM, содержащая идентификатор списка с лицами.
Релевантный индекс	Последний построенный индекс для списка, поскольку индекс может перестраиваться если задача создается в автоматическом режиме.
Сравнение	Процесс сравнения БШ с некоторой партией БШ, который приводит к получению оценок сходства. Цель сравнения — поиск БШ, наиболее похожих на сравниваемый БШ, по некоторому предоставленному пользователем набору БШ.
Аббревиатура	Расшифровка
БШ	Биометрический шаблон
БД, DB	База данных
LP	LUNA PLATFORM
LIM	VisionLabs LP5 Index, LUNA Index Module

1 Обзор

LUNA Index Module — модуль, состоящий из сервисов Index Manager, Indexer и Indexed Matcher, предназначенный для ускорения сравнения большого количества биометрических шаблонов. При классическом сравнении методом последовательного перебора большого количества биометрических шаблонов с помощью сервиса Python Matcher, невозможно получить малую задержку при высоком количестве запросов в секунду. Поэтому требуется использовать методы аппроксимации, реализованные в LIM, которые обменивают некоторую точность на высокую скорость. Эти методы ускоряют сравнение за счет построения индекса, содержащего данные предварительной обработки.

Основной принцип работы модуля заключается в следующем:

Индекс создается с помощью задачи, содержащей «list_id» с прикрепленными лицами, по которому будет происходить сравнение. Также можно не указывать «list_id», а задать настройки для автоматической индексации всех списков, у которых количество лиц превышает значение, заданное в определенной настройке. После того, как индекс создан, пользователь отправляет запрос в сервис API, который перенаправляет его в сервис Python Matcher Proxy. Сервис Python Matcher Proxy определяет где будет выполняться сравнение — в сервисе Python Matcher или в сервисе Indexed Matcher. После сравнения ответ возвращается пользователю.

Возможность выполнения сравнения с помощью сервиса Indexed Matcher регулируется с помощью отдельного параметра в лицензионном ключе LUNA PLATFORM 5. Таким образом, без лицензии можно использовать сервисы Indexer и Indexed Matcher, однако построенные индексы невозможно будет обработать.

Данный документ содержит следующие основные разделы:

- [Основные положения](#). В разделе приведено:
 - описание индекса
 - описание принципа работы LIM
 - описание процесса создания задач на построение индексов
 - описание процесса построения индекса
 - описание процесса сравнения

Рекомендуется начать знакомство с LIM с данного раздела.

- [Взаимодействие сервисов](#). В разделе приведена схема взаимодействия сервисов LIM, отражающая необходимую последовательность действий для выполнения сравнения с помощью сервисов LIM.
- [Сервисы индексирования](#). В разделе приведена основная информация о сервисах LIM и о нюансах работы с ними.
- [Плагин сравнения для Python Matcher Proxy](#). В разделе приведено описание работы плагина

сравнения, встроенного в сервис Python Matcher Proxy, необходимого для выполнения сравнения.

- [Мониторинг](#). В разделе приведено описание процесса мониторинга для сервисов LIM.
- [Диаграммы последовательности](#). В разделе приведены диаграммы последовательности для основных операций LIM.
- [Ошибки API](#). В разделе приведено расширенное описание возвращаемых ошибок сервисами LIM.
- [Настройки сервисов](#). В разделе приведено описание настроек всех сервисов LIM.

2 Основные положения

Дополнительный модуль индексирования LUNA Index Module:

- отправляет запросы на индексацию списка с БШ;
- выполняет построение индекса;
- загружает индекс в память и выполняет сравнение.

LIM содержит следующие сервисы:

- [Index Manager](#) — управляет задачами на построение индекса и координирует сервис Indexer;
- [Indexer](#) — строит индексы на основе списка БШ;
- [Indexed Matcher](#) — выполняет аппроксимированное сравнение ближайших соседей (БШ) с помощью индекса.

См. подробную информацию про задачу поиска ближайшего соседа в [Wiki](#).

Для работы с модулем требуется сервис Python Matcher Proxy с встроенным [плагином сравнения](#), позволяющим определять к какому сервису будут направляться запросы из сервиса LUNA API — к сервису Python Matcher или к сервису Indexed Matcher. Также для работы сервисов Index Manager и Indexed Matcher требуется наличие БД Redis.

Все сервисы LIM масштабируются, что означает возможность использования нескольких экземпляров.

При необходимости можно сменить формат логирования всех сервисов на json (см. настройку FORMAT для каждого сервиса).

2.1 Индекс

Индекс — коллекция предоставленного пользователем набора БШ, развернутых вместе для выполнения аппроксимированного сравнения. Он строится как граф зависимостей, вершинами которого являются БШ. Поиск зависимостей (биометрических шаблонов) в этом графе выполняется при перемещении по его вершинам (см. раздел «[Процесс сравнения](#)» ниже).

Построение индекса требует много ресурсов в течение длительного времени и является достаточно медленным процессом, поэтому нужно грамотно выставить настройку периода автоматического перестроения индекса при появлении изменений кл списке (см. ниже).

Размер списка с БШ определяет соотношение скорости и точности при построении индекса и поиске. Более высокие значения приводят к более точному, но более медленному поиску. Для настройки этих параметров следует использовать настройки «[ef_construction](#)» и «[ef_search](#)».

2.1.1 Связь индекса с версией биометрического шаблона

LUNA Index Module учитывает изменение версии биометрических шаблонов лиц. Сервис Indexer выполняет построение индекса из биометрических шаблонов версии, указанной в настройке «DEFAULT_FACE_DESCRIPTOR_VERSION» сервиса Index Manager. Сервис Index Manager автоматически перестраивает индекс, если в нем не содержится информация о версиях биометрических шаблонов. Сервис Indexed Matcher загружает только те индексы, которые содержат биометрические шаблоны версии, указанной в настройке «DEFAULT_FACE_DESCRIPTOR_VERSION» сервиса Index Manager.

2.1.2 Структура индекса

Индекс состоит из следующих файлов:

- файла meta.json, который содержит метаданные об индексе, в том числе о версии биометрического шаблона и о том, какие объекты индексируются;
- файла index.dat, который содержит двоичные данные индекса;
- файла ids.dat, который содержит упорядоченный список идентификаторов объектов в индексе.

Каждый индекс имеет уникальное имя, которое используется в качестве имени ключа/папки.

Стандартная директория для сохранения индекса указывается для каждого сервиса LIM в настройке «index_storage_local» секции «OTHER» сервиса Configurator. Обратите внимание, что для всех трех сервисов директория должна быть одинаковой.

2.1.3 Процесс создания задачи на построение индекса

Индексация набора БШ осуществляется посредством постановки задач на индексирование в очередь. Такие задачи создаются в сервисе Index Manager. Существует два типа задач построения индекса — **разовый** и **фоновый**.

Разовый тип позволяет единоразово создать задачу на построение индекса с помощью HTTP-запроса «[create task](#)» к сервису Index Manager. В теле запроса необходимо указать требуемый «list_id».

Фоновый тип позволяет создавать задачи на построение индекса в фоновом режиме, где:

- набор списков явно задается в настройке «[indexing_lists](#)»;
- динамически индексируются все существующие в LP списки, у которых количество лиц превышает количество, заданное в настройке «[min_indexing_list_size](#)». При этом значение настройки «[indexing_lists](#)» должно принимать значение «dynamic». Значение по умолчанию — 50000 лиц.

При использовании фонового типа, сервис Index Manager отслеживает изменение количества лиц в списках, взаимодействуя с сервисом Faces. Если количество лиц изменилось, то во внутреннюю очередь будет отправлена новая задача.

Одна задача обрабатывает только один список.

Для отключения построения задач в фоновом режиме необходимо установить значение настройки «indexing_lists» равным [].

2.1.4 Процесс создания индекса

Ниже описан процесс создания индекса:

1. Для начала индексации сервис Index Manager отправляет запрос сервису Indexer с необходимыми параметрами — «list_id» и «task_id». Сервис Indexer преобразует данные параметры в «label» (далее — метка для сравнения) и «index_id» соответственно.
2. При получении запроса на индексацию сервис Indexer запускает отдельный процесс для индексации. На данном этапе Indexer устанавливает свой статус на «indexing».
3. Когда процесс индексации запущен, сервис Indexer извлекает БШ из сервиса Faces. Выборка выполняется партиями по 1000 элементов.
4. После того, как все БШ были извлечены и загружены в память, сервис Indexer начинает построение индекса. Строится ориентированный граф зависимостей БШ (см. раздел «Индекс»).
5. Затем, когда индексация завершена, сам индекс сохраняется в хранилище (файловую систему). В хранилище индекс представляет собой каталог, содержащий некоторые файлы (см. раздел «Структура индекса»).
6. После успешного сохранения индекса процесс индексации останавливается. На данном этапе Indexer устанавливает свой статус на «success». Если же процесс индексации закончился ошибкой, то Indexer установит свой статус на «error».

Информацию о сохраненных индексах можно получить с помощью запросов «get indexes» или «get most relevant indexes» к сервису Index Manager.

Посмотреть статус сервиса Indexer можно с помощью запроса «get tasks» к сервису Index Manager.

Через некоторое время после сохранения индексов все запущенные экземпляры Indexed Matcher автоматически (повторно) загружают эти индексы в память. После загрузки индексов в память можно отправлять запросы на сравнение индексированных наборов БШ с указанной меткой для сравнения.

2.2 Сравнение

Сервис Indexed Matcher загружает более релевантные индексы из хранилища и обрабатывает запросы на сравнение. Поскольку хранилище индексов может содержать несколько версий индексов с определенной меткой для сравнения, сервис Indexed Matcher всегда стремится выполнить сравнение с более новой (т.е. более релевантной) версией.

Индекс устаревает, как только в LUNA PLATFORM 5 создаются или удаляются БШ.

Для синхронизации индексов в памяти сервиса Indexed Matcher с хранилищем индексов предназначен периодический фоновый процесс, называемый перезагрузкой индекса (см. подробную информацию в разделе «[Перезагрузка индекса](#)»).

Если в исходный список были внесены какие-то изменения, то сервис Indexed Matcher обновляет соответствующие индексы у себя в памяти, путем постепенного добавления небольшого количества новых биометрических шаблонов в загруженный в память индекс (см. подробную информацию в разделе «[Обновление индекса в памяти](#)»).

2.2.1 Запросы на сравнение

Запросы на сравнение поступают из сервиса API в сервис Python Matcher Proxy, который использует [плагин сравнения](#) для перенаправления запроса в сервис Indexed Matcher. Сервис Indexed Matcher принимает запросы на сравнение через [поток Redis](#), выполняет сравнение и отправляет результат сравнения в [канал Redis](#), откуда результат перенаправляется в сервис Python Matcher Proxy, а затем в сервис API.

Для запросов для каждой соответствующей метки для сравнения существует поток с именем метки. Несколько запущенных экземпляров Indexed Matcher с загруженным индексом являются [группой потребителей](#) для этого потока.

2.2.2 Процесс сравнения

Сервис Indexed Matcher перемещается по вершинам графа зависимостей (индекса).

После перехода к первой вершине входящий БШ сравнивается со всеми вершинами, связанными с текущей вершиной. Когда найдена наиболее похожая вершина, выполняется следующее сравнение с вершинами, связанными с ней. После нескольких итераций находится наиболее похожая вершина (т.е. БШ с наибольшим показателем сходства). Количество операций при таком поиске значительно сокращается, что увеличивает производительность поиска в сотни раз.

3 Взаимодействие сервисов

Ниже приведена схема взаимодействия сервисов модуля индексирования.

Перед началом работы с LIM пользователь должен создать список и прикрепить к нему необходимое количество лиц.

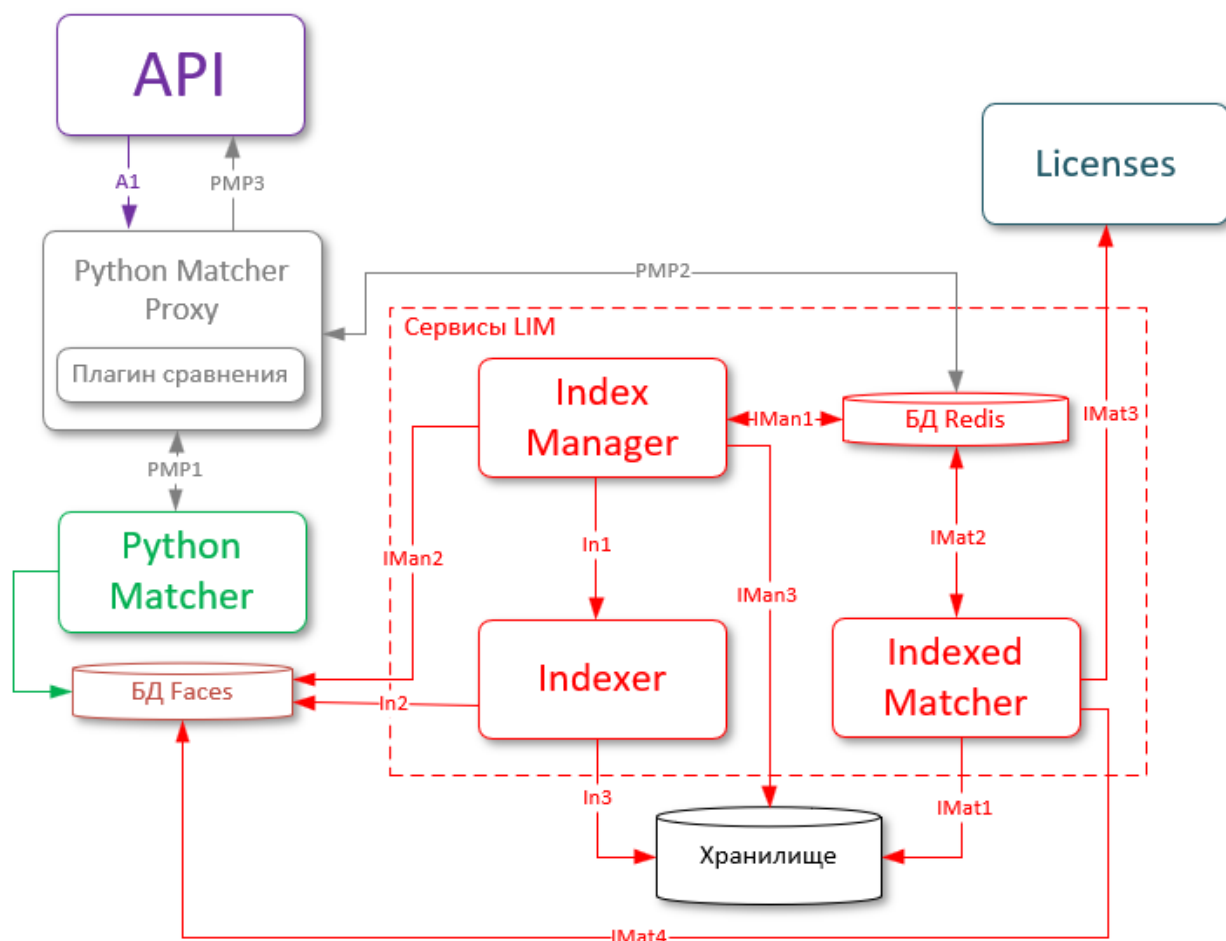


Рис. 1: Взаимодействие сервисов

Перед тем как отправить запрос на сравнение из сервиса API, пользователь должен создать индекс, который создается с помощью создания задачи на его построение. Задача создается в сервисе Index Manager и может быть **разового типа или фонового типа**.

После создания, задача отправляется в очередь в БД Redis (**IMan1**).

Кроме очереди, БД Redis выступает хранилищем для всех задач на построение индексов. Там же используется механизм RedLock, обеспечивающий работу нескольких экземпляров сервиса Index Manager (см. раздел «Работа с несколькими экземплярами»).

Сервис Index Manager взаимодействует с БД Faces для отслеживания изменений в списках (**IMan2**) если была создана задача фонового типа.

Далее Index Manager отправляет запрос в сервис Indexer на построение индекса (**In1**). После получения запроса, сервис Indexer извлекает БШ из БД Faces (**In2**) и начинает построение индекса. Построенный индекс сохраняется в хранилище (**In3**).

После успешного создания индекса, статус задачи изменяется на «success». Перед отправкой запроса на сравнение нужно проверить статус индекса с помощью запроса [«get tasks»](#) к сервису Index Manager. При отправке пользовательских запросов к сервису Index Manager на получение информации об индексах, сервис будет взаимодействовать с хранилищем индексов (**IMan3**).

Indexed Matcher загружает индекс в память (**IMat1**), загружает метку для сравнения (содержит «list_id» индекса в памяти) в БД Redis (**IMat2**) и слушает [поток Redis](#) пока там не появится запрос на сравнение.

Сервис Indexed Matcher постоянно следит за изменениями в списках с лицами, взаимодействуя с базой данных Faces (**IMat4**). В случае внесения новых изменений в список, сервис Indexed Matcher постепенно добавляет новые биометрические шаблоны в соответствующий индекс в памяти. При этом индекс, находящийся в хранилище, остается неизменным до его перестроения. При необходимости данный функционал можно отключить (см. раздел [«Обновление индекса в памяти»](#)).

После того, как индекс создан и загружен в память сервиса Indexed Matcher, пользователь выполняет запрос на сравнение в сервисе API, прикрепляя изображение эталона. Данный запрос перенаправляется в сервис Python Matcher Proxy (**API**), где плагин сравнения генерирует запрос определенного формата, содержащий метку для сравнения и биометрический шаблон эталона, и определяет загружена ли метка для сравнения в БД Redis сервисом Indexed Matcher. Далее выполняется выбор сервиса для выполнения сравнения: - если метка для сравнения не загружена в БД Redis, то запрос направляется в Python Matcher (**PMP1**). В таком случае сравнение будет выполнено с помощью классического метода перебора биометрических шаблонов. - если метка для сравнения загружена в БД Redis, то запрос направляется в виде сообщения в поток Redis. Сервис Indexed Matcher считывает сообщение из потока Redis (**IMat2**). После этого сервис Indexed Matcher проверяет наличие лицензии LUNA PLATFORM 5 на возможность выполнения сравнения, взаимодействуя с сервисом Licenses (**IMat3**).

См. подробную информацию о выборе сервиса для выполнения сравнения в разделе [«Плагин сравнения для Python Matcher Proxy»](#).

После окончания сравнения сервис Indexed Matcher записывает результаты сравнения в [канал Redis](#) (**IMat2**). Плагин сравнения сервиса Python Matcher Proxy считывает результаты сравнения и возвращает их пользователю в сервис API (**PMP2**).

См. раздел [«Диаграммы последовательности»](#) для более подробного описания процессов LUNA Index Module.

4 Сервисы индексирования

4.1 Сервис Index Manager

Сервис управляет процессом создания индексов для списков, содержащих БШ лиц, и выполняет следующие задачи:

- формирует задачи для построения индекса;
- отправляет задачи во внутреннюю очередь;
- извлекает задачи из внутренней очереди и координирует процесс доставки задач для индексирования в сервис Indexer.

Рекомендуется запускать как минимум два экземпляра сервиса Index Manager в целях резервирования. Так как управление задачами осуществляется через Redis, то если один сервис Index Manager не работает, второй сможет продолжить свою работу с шага, на котором остановился предыдущий.

4.1.1 Фоновые процедуры

Сервис Index Manager параллельно выполняет два типа фоновых процедур:

- процедура планирования;
- процедура поиска.

В процедуре планирования Index Manager проверяет, какие наборы списков должны быть проиндексированы, затем создает задачи под уникальным идентификатором «task_id» и помещает их во внутреннюю очередь. Процедура планирования выполняется с периодом (сек), указанным в настройке «[planning_period](#)», или с помощью Cron-подобной строки, указываемой в настройке «[planning_schedule](#)».

В процедуре поиска Index Manager проверяет статус всех запущенных экземпляров Indexer. Если какой-либо экземпляр Indexer завершил построение индекса, Index Manager обновляет информацию о задаче и отправляет данные для [мониторинга](#). Если какой-либо экземпляр Indexer готов принять задачу, сервис Index Manager извлекает следующую задачу из внутренней очереди, отправляет задачу свободному экземпляру Indexer и обновляет информацию о задаче. Процедура поиска выполняется с периодом (сек), указанным в настройке «[lookup_period](#)».

4.1.2 Настройка перестроения и обновления индекса

Сервис Index Manager отправляет задачи на создание, перестроение и обновление индекса согласно вышеописанным процедурам.

По умолчанию выполняется полное перестроение индекса. При необходимости можно обновить индексы вместо их перестроения. В целях экономии времени и ресурсов сервис Index Manager мо-

жет учитывать только вновь добавленные или удаленные биометрические шаблоны. В этом случае сервис Indexer не будет перестраивать индексы с нуля. Чтобы установить приоритет обновления вместо перестроения, нужно задать для настройки `«rebuild_rules»` > `«default»` значение `false`, что означает «не перестраивать, а обновлять». По умолчанию выполняется именно перестроение (значение `true`).

Обновление подразумевает под собой удаление старых и/или добавление новых биометрических шаблонов из/в собранный индекс. Из-за алгоритма индексирования удаление биометрического шаблона из собранного индекса приводит к деградации индекса. Чем больше биометрических шаблонов было удалено, тем хуже качество индекса, что приводит к снижению точности поиска по этому индексу. Настоятельно рекомендуется перестраивать индексы в случае удаления большого количества лиц из исходного набора данных.

Чтобы указать допустимое количество удалений биометрических шаблонов с момента создания индекса, при котором индекс будет перестраиваться с нуля, можно установить соответствующее значение для параметра `«rebuild_rules»` > `«max_removal_for_rebuild»`. Например, если этот параметр установлен на 10, это означает, что можно удалить до 10 биометрических шаблонов, прежде чем индекс будет перестроен с нуля для обеспечения его эффективности. По умолчанию установлено значение «0», что означает «никогда не перестраивать с нуля». Рекомендуемый ориентир для перестройки индексов — удаление не более 10% от общего объема биометрических шаблонов.

4.1.3 Хранилище Index Manager

Вся информация о созданных задачах хранится в БД Redis. Также с помощью механизма Redis Redlock регулируется работа с несколькими экземплярами.

4.1.3.1 Работа с несколькими экземплярами

Режим работы с несколькими экземплярами поддерживается автоматическим выбором экземпляра-мастера (основного экземпляра) на основе Redis Redlock.

См. подробную информацию о распределенных блокировках, выполняющихся с помощью Redis: <https://redis.io/docs/reference/patterns/distributed-locks/>.

Только экземпляр-мастер может выполнять фоновые процедуры планирования и поиска. Остальные экземпляры могут только принимать запросы на единовременное создание индекса, а также выдавать ответы на GET-запросы.

При необходимости можно передать указать следующие переменные окружения при старте соответствующих контейнеров:

- `LIM_MANAGER_MASTER_LOCK` — имя блокировки Redis для экземпляра-мастера сервиса Index Manager. Эта блокировка используется для обеспечения того, чтобы только один экземпляр

Index Manager мог выполнять процедуры планирования и поиска. Значение по умолчанию — `lim_manager_master`.

- `LIM_MATCHER_CONSUMER` — название группы потребителей Redis для запросов на сравнение. Значение по умолчанию — `lim_matcher`.
- `LIM_MATCHER_LOCK_PREFIX` — префикс для имени блокировки сервиса Indexed Matcher в Redis. Это позволяет избежать возможных конфликтов имен с другими пользователями Redis. Значение по умолчанию — `lim_matcher`.

4.1.4 Запросы к сервису

Взаимодействие с сервисом Index Manager осуществляется с помощью HTTP-запросов. Ниже перечислены основные запросы:

- `«get queue»` — позволяет получить список задач и их количество из очереди
- `«get tasks»` — позволяет получить информацию по задачам:
 - `task_id` — идентификатор задачи
 - `status` — статусы: `«pending»`, `«indexing»`, `«success»`, `«error»`
 - `create_time` — время создания задачи на построение индекса в формате RFC 3339
 - `start_time` — время начала построения индекса в формате RFC 3339
 - `end_time` — время окончания построения индекса в формате RFC 3339
 - `indexer` — адрес сервера, где запущен экземпляр Indexer, который обрабатывает указанную задачу
 - `error` — ошибка, полученная во время построения индекса
 - `content` — обрабатываемый `«list_id»`

При необходимости можно отфильтровать получаемые задачи.

- `«create task»` — позволяет единоразово создать задачу на построение индекса
- `«remove tasks»` — позволяет удалить задачи. При необходимости можно отфильтровать удаляемые задачи.
- `«get indexes»` — позволяет получить количество индексов, а также следующую информацию по каждому индексу:
 - `index_id` (равен `task_id`)
 - `index_type` — тип индекса (только список)
 - `label` — обрабатываемый `«list_id»`
- `«remove indexes»` — позволяет удалить индекс из хранилища по идентификатору

- [«get most relevant indexes»](#) — позволяет получить информацию по наиболее релевантному индексу, т.е. по последнему построенному индексу для списка.

Более подробную информацию о запросах, выполняемых к сервису Index Manager, и другие запросы см. [в спецификации OpenAPI](#).

4.2 Сервис Indexer

Сервис Indexer предназначен для обработки задач, полученных сервисом Index Manager, и выполнения [процесса создания индексов](#).

Запросы к сервису Indexer не предназначены для пользователя. Все запросы, связанные с LUNA Index Module должны выполняться к сервису Index Manager (см. раздел [«Запросы к сервису Index Manager»](#)).

Развертывание сервиса Indexer должно выполняться на отдельном сервере, поскольку создание индекса занимает много ресурсов в течение длительного времени. Один экземпляр сервиса Indexer может одновременно создавать только один индекс, поэтому рекомендуется запускать несколько экземпляров сервиса. Сервис также должен быть настроен с хранилищем, которое должно быть достаточно большим.

4.3 Сервис Indexed Matcher

Сервис Indexed Matcher загружает наиболее релевантные индексы из хранилища индексов (файловая система) и обрабатывает запросы на сравнение.

При запуске, сервис Indexed Matcher загружает все индексы последней версии из хранилища индексов в память и настраивает потоки Redis на прием сообщений для сравнения для всех меток сравнения, загруженных в хранилище индексов.

Сервис Indexed Matcher всегда проверяет существование списка при запуске, загрузке нового индекса в память и обновлении индекса в памяти. Индекс без существующего списка будет удален из памяти сервиса.

Для ускорения доступа к индексу, можно настроить кэширование индекса в специальную папку в контейнере сервиса Indexed Matcher (по умолчанию кэширование отключено). Кэширование включается с помощью настройки «[LIM_MATCHER_CACHE](#)».

Запросы к сервису Indexed Matcher не предназначены для пользователя. Все запросы, связанные с LUNA Index Module должны выполняться к сервису Index Manager (см. раздел «[Запросы к сервису Index Manager](#)»).

Сервис Indexed Matcher не взаимодействует с другими сервисами LIM. Он только следит за хранилищем и при появлении индексов загружает их в память. Поскольку обработка запросов на сравнение выполняется через потоки Redis, любое количество экземпляров сервиса Indexed Matcher может быть запущено без каких-либо обновлений конфигурации системы. Количество экземпляров сервиса Indexed Matcher должно определяться требованиями к производительности.

4.3.1 Синхронизация меток сравнения в памяти

Сервис Indexed Matcher синхронизирует метки для сравнения индексов с ключами Redis у себя в памяти. Для всех меток в памяти, сервис устанавливает ключи в следующем формате:

```
matching_label__<label>__<matcher_id>
```

Например, `matching_label__17cdbe41-c7f1-440b-b9ad-aad93c7176ee__127.0.0.1:5200`.

Поле `<matcher_id>` в ключе метки — это хост и порт экземпляра Indexed Matcher. Хост считывается из переменной окружения `VL_LIM_MATCHER_HOST` или, если переменная не задана, угадывается с помощью API сокетов операционной системы. Считывание этих ключей из Redis позволяет плагину сравнения получить информацию о том, для каких экземпляров Indexed Matcher были загружены в память конкретные метки индекса.

Устанавливаемый ключ метки для сравнения имеет срок действия (TTL) и истекает, если не будет обновлен снова. Наличие такого ключа в Redis означает, что некоторые из запущенных экземпляров Indexed Matcher могут обрабатывать запросы на сравнение по метке для сравнения.

4.3.2 Перезагрузка индекса

Для синхронизации индексов в памяти сервиса Indexed Matcher с хранилищем предназначен персодический фоновый процесс, называемый перезагрузкой индекса.

Если индекс исчезает из хранилища, индекс также удаляется из памяти сервиса Indexed Matcher.

Если в хранилище появится новый индекс с новой меткой сравнения, сервис Indexed Matcher попытается загрузить новый индекс в память.

Если в хранилище появляется новый индекс с более новой версией метки сравнения, чем индекс, загруженный в память сервиса Indexed Matcher, то тот попытается загрузить новый индекс в память вместо старого.

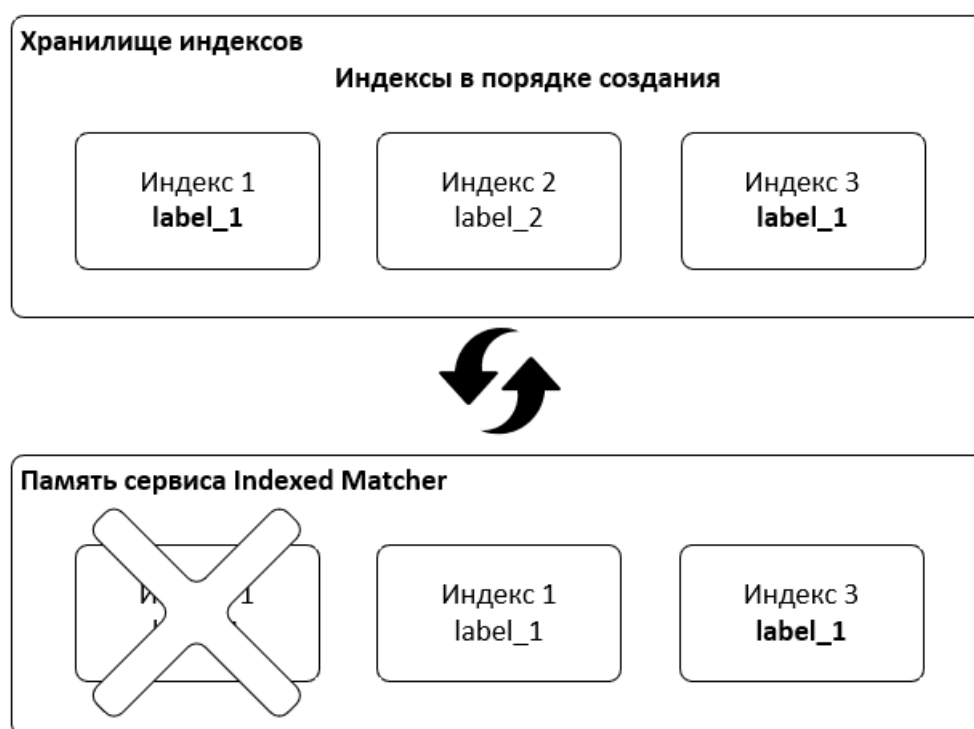


Рис. 2: Замена индекса с неактуальной версией списка (Индекс 1) на новый (Индекс 3)

Чтобы гарантировать, что определенный индекс может быть перезагружен только одним сервисом Indexed Matcher одновременно, используется механизм Redis Redlock. Если блокировка установлена, более старая версия индекса удаляется из памяти сервиса Indexed Matcher, а более новая загружается.

Если при загрузке индекса возникает проблема, например, нехватка памяти, в логи и мониторинг отправляется соответствующее сообщение.

При перезагрузке индекса сервис Indexed Matcher не принимает запросы на сравнение по соответствующей метке. Однако только один Indexed Matcher может одновременно перезагружать индекс для конкретной метки. Поэтому рекомендуется запускать несколько экземпляров Indexed Matcher, чтобы иметь возможность сравнивать все метки в любое время.

См. диаграмму последовательности для перезагрузки индекса в разделе «[Диаграмма перезагрузки индекса](#)».

4.3.3 Обновление индекса в памяти

По умолчанию сервис Indexed Matcher следит за изменениями в списках с лицами 1 раз в секунду. В случае внесения новых изменений в список, сервис Indexed Matcher обновляет соответствующие индексы у себя в памяти, путем постепенного добавления небольшого количества биометрических шаблонов.

Использование данного функционала регулируется настройкой «[enabled](#)».

Данная информация описывается для индекса, который уже загружен в память сервиса Indexed Matcher. Индекс в памяти сервиса и индекс в хранилище могут отличаться.

При обновлении индекса в памяти, сервис Indexed Matcher останавливает сравнение по этому индексу, но продолжает принимать новые запросы на сравнение для этого индекса. Благодаря тому, что в индекс в памяти добавляется небольшое количество биометрических шаблонов (не более 10 БШ за 1 раз), процесс сравнения выполняется с минимальным прерыванием. Однако следует учитывать то, что если в список слишком часто вставляются элементы (десятки и сотни добавлений), то это отразится на существенной деградации в скорости работы, вплоть до почти полной остановки процесса сравнения.

Во время обновления индекса, сервис Index Matcher выводит следующую информацию в логи:

```
Refresh index for: 2d5832ad-8c8f-415f-a0b4-d12d69fabd60
Sync: 5->6, 0->0
Refresh index for: 2d5832ad-8c8f-415f-a0b4-d12d69fabd60 has finished
successfully
```

где:

- 2d5832ad-8c8f-415f-a0b4-d12d69fabd60 — идентификатор списка;
- 5->6 — информация о выкачивании пакетов с БШ (1 пакет равен 10 БШ) из базы данных Faces. Здесь 6 — общее количество пакетов, которые необходимо выкачать из БД, а 5 — текущее количество выкачанных пакетов. Таким образом, сообщение 5->6 означает, что синхронизация будет продолжена и будет выкачан еще один пакет.
- 0->0 — информация об удалении пакетов с БШ (1 пакет равен 10 БШ) из индекса в памяти. Принцип работы аналогичен выкачиванию пакетов из базы данных Faces.

Скорость обновления индекса в памяти зависит от размера текущего индекса.

Если используется данный функционал, то нет необходимости и не рекомендуется выполнять частые перестроения индекса. Соответственно, рекомендуется увеличить [период процедуры планирования](#) (настройка «[planning_period](#)»). Однако добавление новых лиц в индекс в памяти происходит медленнее, чем перестроение индекса, поэтому нет смысла использовать данную функцию если в список было добавлено очень большое количество лиц. В таком случае проще заново перестроить индекс.

Открепление лиц от списка не приводит к удалению этих лиц из индекса в памяти. В таком случае биометрические шаблоны помечаются как недоступные для поиска, поэтому индекс сохраняет выделенное для них место для хранения.

См. диаграмму последовательности для обновления индекса в разделе «[Диаграмма обновления индекса](#)».

4.3.4 Кэширование индекса

Можно включить кэширование индекса для ускорения процесса загрузки данных в память сервиса Indexed Matcher. Использование кэширования позволяет не загружать индекс в память из Хранилища, а загружать его из кэшированного каталога в случае непредвиденной перезагрузки сервиса Indexed Matcher.

Кэширование включается при задании промежуточного каталога для хранения и загрузки индексов в настройке «[lim_matcher_cache](#)» сервиса Configurator. По умолчанию каталог не указывается, т.е. кэширование отключено.

Промежуточный каталог должен быть расположен в локальной файловой системе (использование таких вещей, как GlusterFS или NFS, может вызвать ошибки). Каждый раз, когда сервис Indexed Matcher перезагружает свои индексы, он пытается очистить каталог кэша, удаляя старое поколение индексов списка. Кэш-система имеет механизм блокировки. В случае нескольких экземпляров сервиса Indexed Matcher, работающих на одном хосте и совместно использующих один и тот же каталог для кэша, блокировка предотвратит загрузку одних и тех же индексов несколько раз. Это означает, что хранилище индексов будет задействовано ровно один раз, когда данные отправляются между хостом сервисов Indexed Matcher и хранилищем.

5 Плагин сравнения для Python Matcher Proxy

С помощью плагина сравнения сервис Python Matcher Proxy может перенаправлять запросы на сравнение, поступающие из сервиса API, либо в сервис Python Matcher, либо в сервис Indexed Matcher. Принцип работы плагина и описание выбора сервиса, в котором будет выполняться сравнение, описаны ниже.

Плагин сравнения уже встроен в Docker-контейнер сервиса Python Matcher Proxy, необходимо только включить его использование (см. руководство по установке).

5.1 Описание работы плагина сравнения

Каждый запрос на сравнение представлен в виде всех возможных комбинаций кандидатов и эталонов, затем каждая такая комбинация обрабатывается как отдельный подзапрос следующим образом (дополнительный подзапрос означает комбинацию эталонов и кандидатов):

- Получение сложности подзапроса (см. [«Сложность запроса на сравнение»](#)).
- Выбор способа обработки подзапроса: с помощью плагина сравнения или сервиса Python Matcher.
 - Если на предыдущем шаге был выбран сервис Python Matcher, он обработает подзапрос и вернет ответ сервису Python Matcher Proxy.
 - Если на предыдущем шаге был выбран плагин сравнения, то он обработает подзапрос. Если подзапрос успешно обработан, ответ возвращается в сервис Python Matcher Proxy. Если подзапрос не был успешно обработан, будет совершена попытка обработки в сервисе Python Matcher.
- Если запрос был успешно обработан плагином сравнения, но у него нет доступа ко всем полям объекта, указанным в подзапросе в поле [«target»](#) для сравнения, то сервис Python Matcher Proxy получит эти данные перед следующим шагом.
- Сервис Python Matcher Proxy собирает результаты от всех подзапросов, сортирует их в правильном порядке, и выдает ответ пользователю.

5.2 Сложность запроса на сравнение

Matching cost — это число с плавающей запятой, определяющее сложность запроса на сравнение с использованием плагина.

Определение сложности сравнения необходимо для выбора наилучшего способа обработки запроса на сравнение: с помощью сервиса Python Matcher или с помощью плагина сравнения.

Значение сложности запроса на сравнение для сервиса Python Matcher равно бесконечности. Если в БД Redis загружена метка для сравнения, то будет рассчитана определенная сложность за-

проса и будет использован плагин сравнения. Если же метка не загружена, то будет использован сервис Python Matcher.

5.3 Поля target, выступающие в качестве сравнения

Сервис Python Matcher имеет доступ ко всем данным сущностей сравнения, поэтому он может обрабатывать запросы на сравнение со всеми полями target. В свою очередь, плагин сравнения может не иметь доступ к данным, указанным в поле target запроса. В этом случае сервис Python Matcher Proxy дополнит ответ плагина сравнения отсутствующими данными о полях target, например:

- ответ на сравнение содержит следующие поля target: face_id , user_data и similarity, а плагин сравнения не имеет доступа к полю user_data, тогда:
 - плагин сравнения сравнивает эталон с указанными face_id и возвращает результат сравнения сервису Python Matcher Proxy, который содержит только пары face_id и similarity.
 - для каждого кандидата на сравнение в результате сервис Python Matcher Proxy получит user_data из основной базы данных по face_id и объединит face_id и similarity с user_data.
 - плагин сравнения вернет пользователю расширенный ответ с указанными полями target.
- ответ на сравнение содержит следующие поля target: age и gender, а выбранный плагин сравнения имеет доступ только к полям event_id , descriptor и age.
 - плагин сравнения сравнивает эталон и возвращает соответствующий ответ в сервис Python Matcher Proxy, который содержит только пары event_id , age и similarity.
 - для каждого кандидата на сравнение в результате сервис Matcher Proxy получит gender из основной базы данных по event_id и объединит event_id с age, а также после этого удалит ненужные event_id и similarity из ответа.
 - плагин сравнения вернет пользователю подготовленный ответ с указанными полями target.

В LUNA PLATFORM может использоваться несколько плагинов сравнения. См. подробную информацию в разделе «Плагины сравнения» руководства администратора LUNA PLATFORM.

6 Мониторинг

В LIM есть несколько методов мониторинга:

- Отправка данных в InfluxDB (включен по умолчанию)
- Экспорт метрик в формате Prometheus через ресурс `/metrics` (отключен по умолчанию)

Глобально мониторинг LIM работает аналогично мониторингу LUNA PLATFORM. Исключением являются только отправляемые данные в InfluxDB (см. ниже).

Метрики в формате Prometheus содержат точно такие же данные, как и в LUNA PLATFORM.

См. подробную информацию в разделе «Мониторинг» руководства администратора LUNA PLATFORM.

6.1 Отправка данных в InfluxDB

6.1.1 Отправляемые данные

Для каждого сервиса типы события мониторинга отличаются. Ниже приведена таблица, отражающая все типы событий для каждого сервиса:

Сервис	Типы событий
Index Manager	все HTTP-запросы, все неудачные HTTP-запросы, построение индекса
Indexer	все HTTP-запросы, все неудачные HTTP-запросы
Indexed Matcher	все HTTP-запросы, все неудачные HTTP-запросы, перезагрузка индексов , запросы на сравнение (проходят через Redis)

Каждое событие — это точка временного ряда. Точка представлена с использованием следующих данных:

- тип события (запросы или ошибки)
- отметка времени начала запроса
- теги
- поля

Тег — это индексированные данные в хранилище. Он представлен в виде словаря, где

- `keys` — имена тегов (string),
- `values` — значения с типами данных string, integer, float

Поле представляет собой неиндексированные данные в хранилище. Оно представлено в виде словаря, где

- `keys` — имена полей (string),

- values — значения с типами данных string, integer, float

Сохранение данных для **серии запросов** запускается при каждом HTTP-запросе. Каждая точка содержит данные о соответствующем запросе (время выполнения и т.д.).

- теги

Имя тега	Описание
service	сервис, «lim-indexer», «lim-manager» или «lim-matcher»
route	объединение метода запроса и ресурса запроса (GET:/version)
status_code	HTTP статус код ответа

- поля

Имя поля	Описание
request_id	идентификатор запроса
execution_time	время выполнения запроса

Сохранение данных для **серии ошибок** запускается при сбое HTTP-запроса. Каждая точка содержит *error_code* ошибки.

- теги

Имя тега	Описание
service	сервис, «lim-indexer», «lim-manager» или «lim-matcher»
route	объединение метода запроса и ресурса запроса (GET:/version)
status_code	HTTP статус код ответа
error_code	код ошибки LIM

- поля

Имя поля	Описание
request_id	идентификатор запроса

Сохранение данных для **серии построения индекса** запускается при ошибке во время построения

индекса.

- теги

Имя тега	Описание
service	сервис, «lim-manager» или «lim-matcher»
socket_ address	адрес сервиса в формате <host> : <port>
stage	«build_index» (построить индекс), «load_index» (загрузить индекс) или «drop_index» (удалить индекс)
label	метка для сравнения индекса (идентификатор списка)
error_code	код ошибки LIM (0 — запрос выполнен успешно)

- поля

Имя поля	Описание
task_id	идентификатор задачи на индексирование
pending	время, проведенное во внутренней очереди (сек)
duration	время построения индекса (сек)
generation	unix-время генерации индекса

Сохранение данных для **серии сравнения** запускается при выполнении сравнения.

- теги

Имя тега	Описание
service	сервис, всегда «lim-matcher»
socket_ address	адрес сервиса в формате <host> : <port>
label	метка для сравнения индекса (идентификатор списка)
error_code	код ошибки LIM (0 — запрос выполнен успешно)

- поля

Имя поля	Описание
request_id	идентификатор запроса
index_id	идентификатор индекса
execution_ time	время выполнения запроса на сравнение (сек)

7 Диаграммы последовательностей

В данном разделе приведены диаграммы последовательности для основных операций LIM.

7.1 Диаграмма построения индекса

Индекс строится после создания задачи на его построение. Есть два типа создания задач на построение индекса — **разовый** и **фоновый**.

Ниже приведена диаграмма последовательности для обоих типов создания задач.

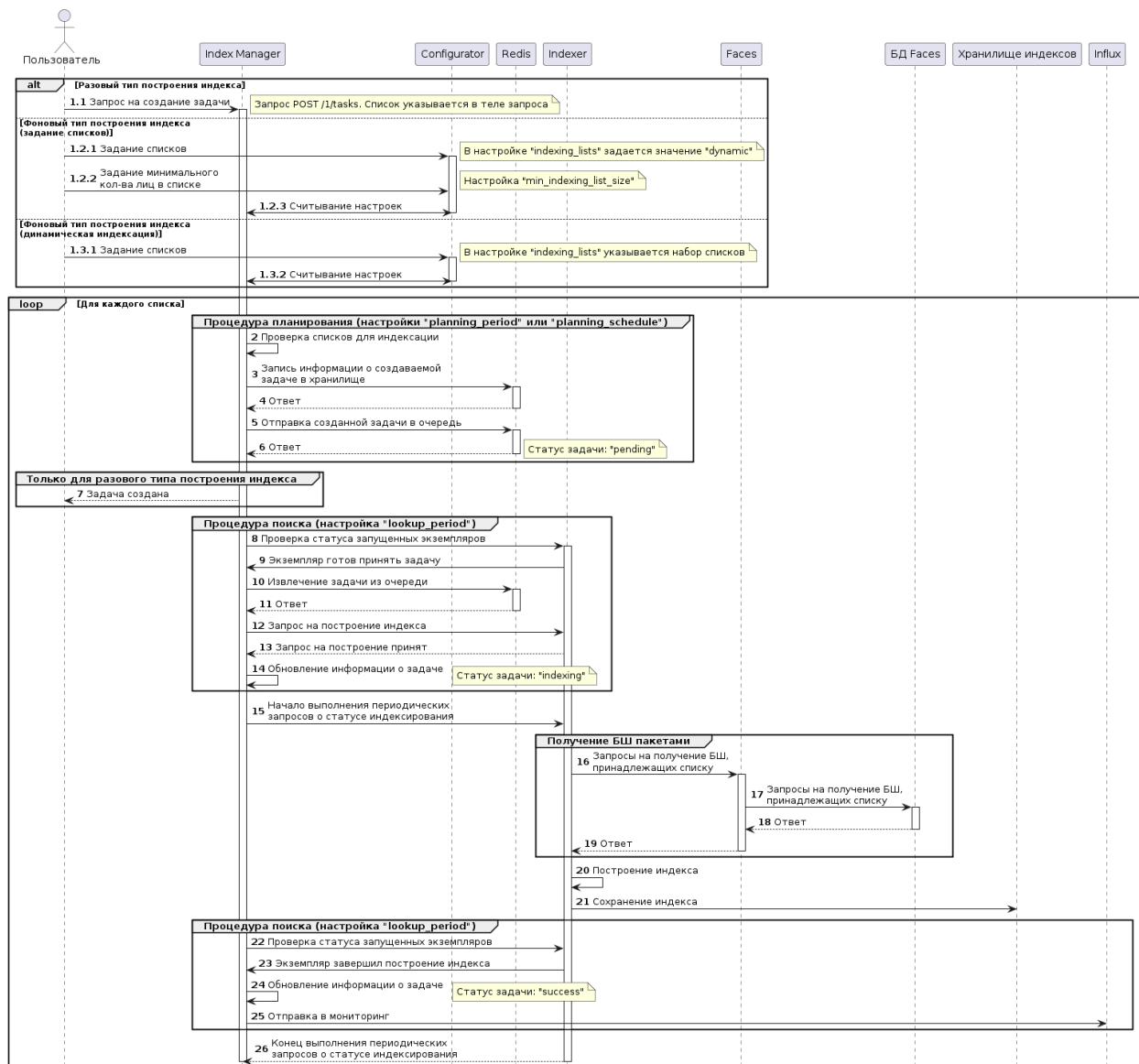


Рис. 3: Диаграмма построения индекса

- **(1.1)** Отправка запроса «create task» на [разовое](#) построение индекса
- **(1.2.1)** Включение [динамического индексирования](#) списков с помощью задания значения «dynamic» для настройки «indexing_lists»
- (1.2.2) Задание минимального количества лиц в списке по которому будет строиться индекс (по умолчанию 50 000) в настройке «min_indexing_list_size»
- (1.2.3) Считывание значений настроек из сервиса Configurator
- **(1.3.1)** Задание набора списков в настройке «indexing_lists»
- (1.3.2) Считывание значения настройки из сервиса Configurator
- (2) Начало [процедуры планирования](#) (регулируется настройками «[planning_period](#)» и «[planning_schedule](#)»). Проверка, какие наборы списков должны быть проиндексированы
- (3) Запись информации о создаваемой задаче в хранилище Redis
- (4) Ответ, что задача помещена в хранилище
- (5) Отправка задачи во внутреннюю очередь
- (6) Ответ, что задача помещена в очередь. На данном этапе статус задачи устанавливается как «pending»
- (7) Ответ о создании задачи. Только для [разового](#) построения индекса (см. (1.1))
- (8) Начало [процедуры поиска](#) (регулируется настройкой «[lookup_period](#)»). Проверка статуса запущенных экземпляров Indexer
- (9) Ответ, что экземпляр Indexer готов начать работу
- (10) Извлечение задачи из внутренней очереди
- (11) Ответ, что задача извлечена
- (12) Запрос на построение индекса по созданной задаче
- (13) Ответ, что запрос на построение индекса по созданной задаче принят
- (14) Обновление информации о задаче. На данном этапе статус задачи меняется на «indexing»
- (15) Начало выполнения периодических запросов о статусе индексирования
- (16) Выполнение запросов к сервису Faces на получение биометрических шаблонов, принадлежащих списку
- (17) Перенаправление запросов в БД Faces
- (18) Выгрузка данных из базы данных Faces

- (19) Перенаправление выгруженных данных в сервис Faces
- (20) Процесс построения индекса
- (21) Сохранение индекса в Хранилище индексов
- (22) Продолжение [процедуры поиска](#). Проверка статуса запущенных экземпляров Indexer
- (23) Ответ, что экземпляр завершил построение индекса
- (24) Обновление информации о задаче. На данном этапе статус задачи меняется на «success»
- (25) Отправка в [базу данных Influx](#)
- (26) Окончание выполнения периодических запросов о статусе индексирования

Узнать актуальный статус задачи можно с помощью запроса [«get tasks»](#) после её создания.

7.2 Диаграмма первичной загрузки индекса в память

Перед началом сравнения индекс загружается в память сервиса Indexed Matcher.



Рис. 4: Диаграмма загрузки индекса в память

- (1) Загрузка самого актуального индекса из Хранилища в память сервиса Indexed Matcher

- (2) Кэширование индекса в промежуточный каталог. Каталог указывается в настройке «lim_matcher_cache» сервиса Indexed Matcher. По умолчанию кэширование отключено. См. раздел [«Кэширование индекса»](#) для подробной информации.
- (3) Загрузка метки для сравнения в Redis. Имя метки совпадает с идентификатором списка
- (4) Ответ, что метка загружена
- (5) Регистрация [потока Redis](#) в качестве обработчика сообщений для соответствующей метки для сравнения. На данном этапе потоку присваивается имя метки для сравнения (т.е. имя идентификатора списка)
- (6) Ответ, что поток зарегистрирован
- (7) Ожидание появления запроса на сравнение, который содержит метку для сравнения с точно таким же названием как та, которая загружена в Redis (3) и для которой зарегистрирован поток (5)

7.3 Диаграмма сравнения биометрических шаблонов

Данная диаграмма подразумевает, что индекс уже загружен в память сервиса Indexed Matcher (см. [«Диаграмма первичной загрузки индекса в память»](#)).

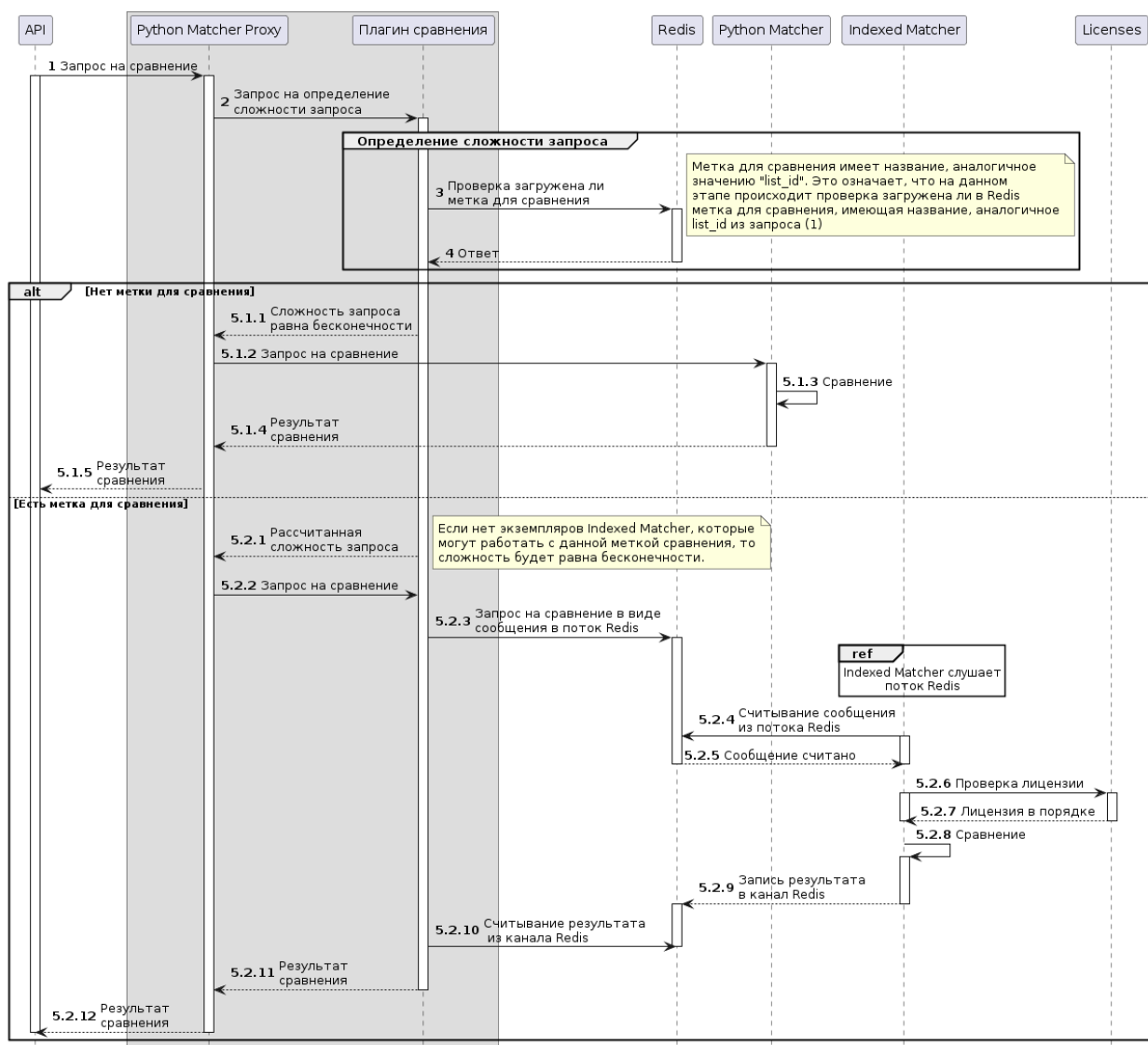


Рис. 5: Диаграмма сравнения биометрических шаблонов

- (1) Отправка запроса на сравнение
- (2) Запрос на определение сложности запроса
- (3) Проверка наличия метки для сравнения
- (4) Ответ
 - (5.1.1) Если нет метки для сравнения, то сложность запроса считается бесконечной и плагин сравнения возвращает соответствующую информацию сервису Python Matcher Proxy
 - (5.1.2) Запрос на сравнение в сервис Python Matcher
 - (5.1.3) Процесс сравнения в сервисе Python Matcher

- (5.1.4) Ответ о результате сравнения в Python Matcher Proxy
- (5.1.5) Ответ о результате сравнения в API
- **(5.2.1)** Если метка для сравнения существует, то высчитывается определенная сложность запроса и возвращается ответ в сервис Python Matcher Proxy
- (5.2.2) Отправка запроса на сравнение в плагин сравнения
- (5.2.3) Конвертация запроса и отправка его в виде сообщения в [поток Redis](#)
- (5.2.4) Сервис Indexed Matcher слушает поток Redis (см. [«Диаграмма первичной загрузки индекса в память»](#)). Как только появляется запрос на сравнение, который содержит метку для сравнения с точно таким же названием как та, которая ранее загружена в Redis и для которой зарегистрирован был поток, то Indexed Matcher считывает сообщение на сравнение
- (5.2.5) Ответ, что сообщение считано
- (5.2.6) Выполнение проверки лицензии
- (5.2.7) Ответ о состоянии лицензии (данная диаграмма не отражает случай невалидной лицензии)
- (5.2.8) Выполнения процесса сравнения
- (5.2.9) Запись результата сравнения в [канал Redis](#)
- (5.2.10) Считывание плагином сравнения результата сравнения из канала Redis
- (5.2.11) Отправка результата сравнения в Python Matcher Proxy
- (5.2.12) Отправка результата сравнения в API

7.4 Диаграмма перезагрузки индекса

Данная диаграмма отражает только последовательность работы в случае, когда в Хранилище индексов появляется индекс с более новой версией метки сравнения. Если же индекс исчезает из Хранилища, то после проверки состояния индекса, сервис Indexed Matcher удаляет его из памяти. Данный функционал включается с помощью настройки «enabled = 1» (по умолчанию включена) секции «LIM_MATCHER_REFRESH» настроек сервиса Indexed Matcher.

Рекомендуется ознакомиться с разделом [«Перезагрузка индекса»](#).



Рис. 6: Диаграмма перезагрузки индекса

- (1) Проверка изменился ли текущий индекс в Хранилище индексов. Если в Хранилище есть более свежий индекс по времени сохранения для той же метки для сравнения и с нужной версией биометрического шаблона, то это будет учитываться как то, что текущий индекс не актуален.
- (2) Ответ, что индекс не актуален
- (3) Запрос на включение блокировки Redis
- (4) Ответ об успешной блокировке Redis. Если в данный момент невозможно выполнить блокировку, то Indexed Matcher будет выполнять попытки блокировки пока очередная попытка не закончится успехом.
- (5) Если блокировка успешна, то остановка сравнения по текущему индексу

- (6) Удаление старого индекса из памяти Indexed Matcher
- (7) Загрузка нового индекса из Хранилища индексов в память сервиса Indexed Matcher
- (8) Ответ, что индекс загружен в память
- (9) Запрос на отключение блокировки Redis
- (10) Ответ, что блокировка отключена
- (11) Продолжение сравнения

7.5 Диаграмма обновления индекса

Ниже описана диаграмма последовательности для процесса достроения индекса в памяти сервиса Indexed Matcher.

Данная информация описывается для индекса, который уже загружен в память сервиса Indexed Matcher. Индекс в памяти сервиса и индекс в хранилище могут отличаться. Рекомендуется ознакомиться с разделом [«Обновление индекса в памяти»](#).

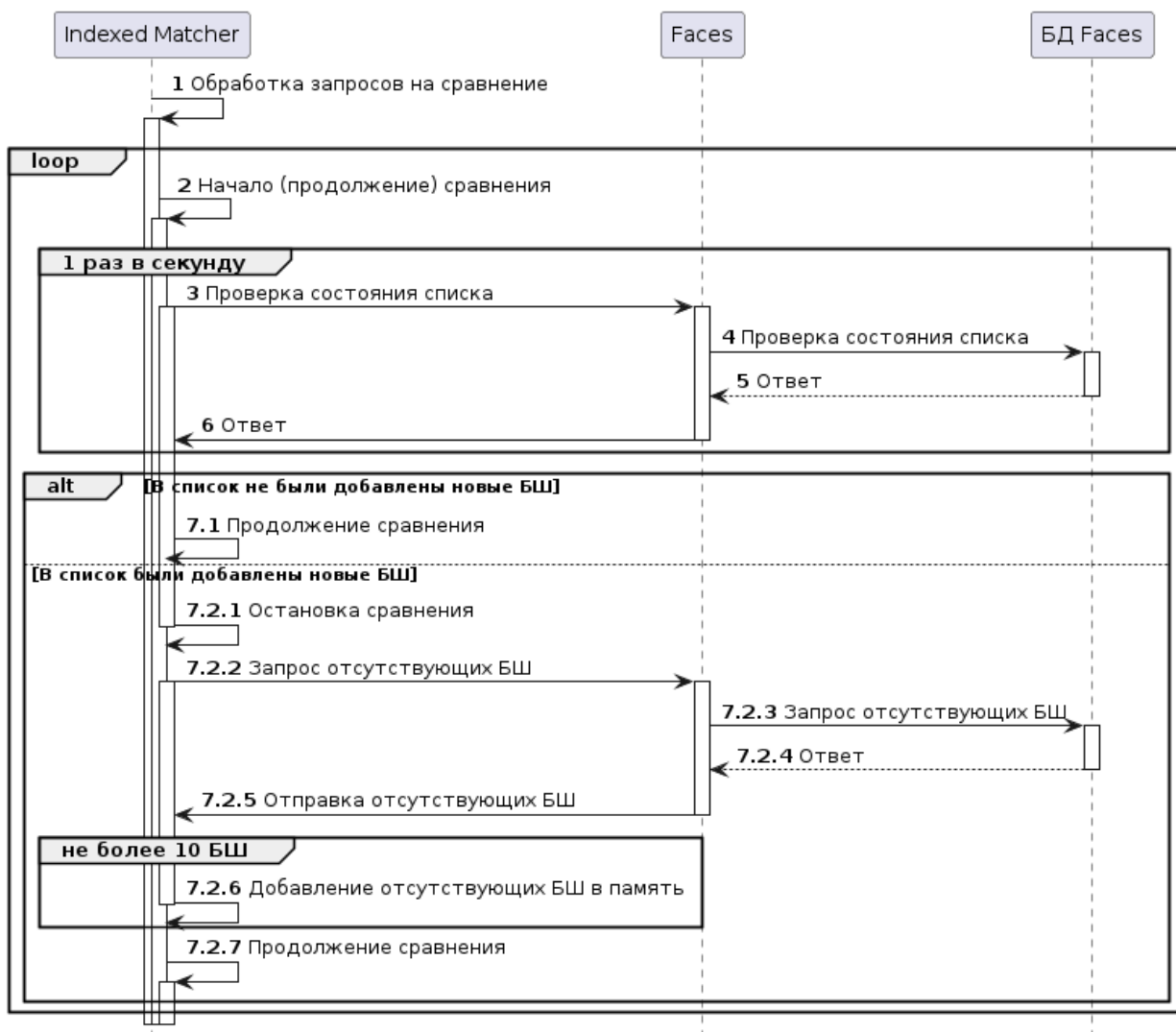


Рис. 7: Диаграмма обновления индекса в памяти

- (1) Прием обработки запросов на сравнение
- (2) Начало сравнения
- (3) Запрос на проверку состояния списка к Faces (запрос выполняется каждую секунду)
- (4) Перенаправление запроса на проверку состояния списка к БД Faces
- (5) Ответ
- (6) Ответ
 - (7.1) Если в список не были добавлены новые биометрические шаблоны, то сравнение продолжится до очередной проверки состояния списка (3)
 - (7.2.1) Если в список были добавлены новые биометрические шаблоны, то сравнение останавливается

- (7.2.2) Запрос отсутствующих биометрических шаблонов к сервису Faces
- (7.2.3) Перенаправление запроса к БД Faces
- (7.2.4) Ответ, содержащий отсутствующие биометрические шаблоны
- (7.2.5) Ответ, содержащий отсутствующие биометрические шаблоны
- (7.2.6) Добавление отсутствующих биометрических шаблонов в индекс, расположенный в памяти сервиса Indexed Matcher (не более 10 БШ за раз)
- (7.2.7) Продолжение сравнения до очередной проверки состояния списка (3)

8 Использование Redis Sentinel

Redis Sentinel — это компонент в системе управления высокой доступностью (HA) для базы данных Redis. Он используется для обнаружения сбоев в мастер-нодах Redis и автоматической переконфигурации системы для обеспечения непрерывной работы.

Для использования Redis Sentinel необходимо заполнить параметры группы «sentinel» из групп настроек «LIM_MANAGER_DB» и «LIM_MATCHER_DB».

Также использование Redis Sentinel можно указать в группе настроек «LUNA_INDEXED_LIST_PLUGIN», которая отвечает за соединение [плагина сравнения](#) с Redis при [высчитывании сложности запроса](#).

Пример заполнения параметров из группы настроек «LUNA_INDEXED_LIST_PLUGIN» для использования Redis Sentinel:

```
LUNA_INDEXED_LIST_PLUGIN = {"REDIS_URL": "redis+sentinel://localhost:26379,localhost:26378/indexed_matcher?username=master_username&password=master_password", "REQUEST_TIMEOUT": 60}
```

Здесь:

- «redis+sentinel://» — префикс, указывающий на использование протокола Redis и Redis Sentinel для подключения;
- «localhost:26379,localhost:26378» — список адресов и портов Sentinel'ов, которые сервис будет использовать для обнаружения состояния мастер-нод Redis и координации действий при сбоях;
- «/indexed_matcher» — путь к конкретной базе данных Redis.
- «?username=master_username&password=master_password» — данные авторизации.

9 Ошибки API

Данный раздел описывает ошибки, которые возвращают сервисы индексирования. Каждая из ошибок API имеет свой уникальный номер. Его удобно использовать, чтобы найти ошибку.

Некоторые из этих ошибок могут иметь несколько различных причин возникновения.

В случае возникновения «Internal server error» или иной непредвиденной ошибки следует обратиться к логам сервиса для получения дополнительной информации о проблеме.

9.1 Ошибки сервиса Indexer

9.1.1 Вернулся код ошибки 26100

Текст ошибки:

Internal server error

Источник ошибки:

Ошибки сервиса Indexer

Описание ошибки:

Произошла неизвестная ошибка.

9.1.2 Вернулся код ошибки 26101

Текст ошибки:

Indexer busy

Источник ошибки:

Ошибки сервиса Indexer

Описание ошибки:

Сервис Indexer сейчас не может обработать запрос.

Дождитесь окончания обработки индекса или запустите еще один экземпляр сервиса Indexer.

9.1.3 Вернулся код ошибки 26103

Текст ошибки:

Build failed

Источник ошибки:

Ошибки сервиса Indexer

Описание ошибки:

Произошла ошибка при создании индекса. Повторите попытку.

9.1.4 Вернулся код ошибки 26104**Текст ошибки:**

Build process failed. Build process died

Источник ошибки:

Ошибки сервиса Indexer

Описание ошибки:

Построение индекса завершилось ошибкой.

Данная ошибка могла возникнуть из-за завершения процесса (OOM killer).

9.1.5 Вернулся код ошибки 26105**Текст ошибки:**

Build process failed. List is empty

Источник ошибки:

Ошибки сервиса Indexer

Описание ошибки:

Построение индекса завершилось ошибкой из-за отсутствия привязанных лиц к списку.

Убедитесь, что лица привязаны к списку.

9.1.6 Вернулся код ошибки 26106**Текст ошибки:**

Build process cancelled

Источник ошибки:

Ошибки сервиса Indexer

Описание ошибки:

Построение индекса было отменено с помощью запроса «/stop» к сервису Indexer.

9.1.7 Вернулся код ошибки 26107

Текст ошибки:

Build process failed. List not found

Источник ошибки:

Ошибки сервиса Indexer

Описание ошибки:

Построение индекса завершилось ошибкой из-за отсутствия списка.

Убедитесь, что список существует.

9.1.8 Вернулся код ошибки 26108

Текст ошибки:

Build process failed. Index with specified ID already exists

Источник ошибки:

Ошибки сервиса Indexer

Описание ошибки:

Построение индекса завершилось ошибкой из-за того, что индекс с указанным идентификатором уже существует.

9.1.9 Вернулся код ошибки 26109

Текст ошибки:

Build process failed. It's probably due to running out of memory and the OS was triggering the OOM killer.

Источник ошибки:

Ошибки сервиса Indexer

Описание ошибки:

Построение индекса завершилось ошибкой. Вероятно, это связано с нехваткой памяти, и ОС запускала OOM killer.

9.2 Ошибки сервиса Index Manager

9.2.1 Вернулся код ошибки 26201

Текст ошибки:

Task duplicate. Indexing task already exists

Источник ошибки:

Ошибки сервиса Index Manager

Описание ошибки:

Ошибка при попытке создать задачу. Задача с таким идентификатором уже существует (идентификатор задачи равен идентификатору списка).

9.2.2 Вернулся код ошибки 26202

Текст ошибки:

Index duplicate. Index for the most recent content version already exists

Источник ошибки:

Ошибки сервиса Index Manager

Описание ошибки:

Ошибка при попытке создать индекс. Наиболее релевантный индекс уже существует.

9.2.3 Вернулся код ошибки 26203

Текст ошибки:

Internal server error. Indexer was restarted for internal reasons

Источник ошибки:

Ошибки сервиса Index Manager

Описание ошибки:

Сервис Indexer перезапустился по неизвестным причинам.

9.2.4 Вернулся код ошибки 26204

Текст ошибки:

Object not found. Index with id «{ }» not found in the storage

Источник ошибки:

Ошибки сервиса Index Manager

Описание ошибки:

Индекс с указанным идентификатором не найден в хранилище. Убедитесь, что указанный индекс существует.

Проверить список существующих индексов можно с помощью запроса [«get indexes»](#).

Ссылка ведет на последнюю версию документации. Убедитесь, что вы читаете документацию для вашей текущей версии LIM.

9.3 Ошибки сервиса Indexed Matcher

9.3.1 Вернулся код ошибки 26301

Текст ошибки:

Bad/incomplete input data. Failed to validate matching request: {value}

Источник ошибки:

Ошибки сервиса Indexed Matcher

Описание ошибки:

Обработка запроса на сравнение завершилась ошибкой из-за некорректных указанных данных.

9.3.2 Вернулся код ошибки 26302

Текст ошибки:

Bad/incomplete input data. Failed to load descriptor bytes {value}

Источник ошибки:

Ошибки сервиса Indexed Matcher

Описание ошибки:

Ошибка при загрузке БШ.

Ошибка может возникнуть из-за неправильного формата БШ.

9.3.3 Вернулся код ошибки 26303

Текст ошибки:

Internal server error. Failed to search descriptor {value}

Источник ошибки:

Ошибки сервиса Indexed Matcher

Описание ошибки:

Внутренняя ошибка при поиске БШ.

9.3.4 Вернулся код ошибки 26304

Текст ошибки:

Index not found. Index for label {value} not found

Источник ошибки:

Ошибки сервиса Indexed Matcher

Описание ошибки:

Не найден индекс для указанной метки для сравнения.

Убедитесь, что индекс с идентификатором, равным метке для сравнения, создан.

Проверить список существующих индексов можно с помощью запроса [«get indexes»](#) к сервису Index Manager.

Ссылка ведет на последнюю версию документации. Убедитесь, что вы читаете документацию для вашей текущей версии LIM.

9.3.5 Вернулся код ошибки 26305

Текст ошибки:

Descriptor version mismatch. Descriptor of version {value} cannot be searched in index of version {value}

Источник ошибки:

Ошибки сервиса Indexed Matcher

Описание ошибки:

Обнаружено несовпадение версий сравниваемых биометрических шаблонов.

Убедитесь, что версии БШ совпадают. Обновить версии БШ можно с помощью задачи Additional extraction (см. раздел «Запуск задачи Additional extraction» руководства администратора LUNA PLATFORM 5).

Версия БШ по умолчанию содержится в настройке «DEFAULT_FACE_DESCRIPTOR_VERSION» в сервисе Configurator.

9.3.6 Вернулся код ошибки 26306

Текст ошибки:

Index processing internal error. Skip load index for label {value} due to index storage damage

Источник ошибки:

Ошибки сервиса Indexed Matcher

Описание ошибки:

Внутренняя ошибка обработки индекса. Загрузка индекса для указанной метки для сравнения пропущена из-за повреждения хранилища индексов.

9.3.7 Вернулся код ошибки 26307**Текст ошибки:**

Index processing internal error. Skip load index for label {value} due to insufficient memory

Источник ошибки:

Ошибки сервиса Indexed Matcher

Описание ошибки:

Внутренняя ошибка обработки индекса. Загрузка индекса для указанной метки для сравнения пропущена из-за недостаточного количества свободного места.

Убедитесь, что имеется достаточное количество свободного места.

Систему можно очистить с помощью команды `docker system prune`. По умолчанию будут удалены остановленные контейнеры, слои, не связанные с используемыми образами, тома и сети, не относящиеся к работающим контейнерам.

9.3.8 Вернулся код ошибки 26308**Текст ошибки:**

Index processing internal error. Skip load index for label {value} due to index corruption

Источник ошибки:

Ошибки сервиса Indexed Matcher

Описание ошибки:

Внутренняя ошибка обработки индекса. Загрузка индекса для указанной метки для сравнения пропущена из-за поврежденного индекса.

10 Описание параметров сервисов

10.1 Настройки сервиса Index Manager

Данный раздел описывает параметры сервиса Index Manager.

Настройку сервиса можно выполнить с помощью сервиса Configurator.

10.1.1 Группа параметров LIM_MANAGER_LOGGER

Данная группа параметров задает настройки логирования.

10.1.1.1 log_level

Параметр задает уровень отладочной печати, по приоритету: «ERROR», «WARNING», «INFO», «DEBUG».

Формат задания настройки — string.

Значение по умолчанию — INFO.

10.1.1.2 log_time

Параметр задает формат времени, используемый в записях лога. Доступны следующие значения:

- «LOCAL» — отображает местное время системы, на которой выполняется запись логов;
- «UTC» — отображает координированное всемирное время, которое является стандартом времени и не зависит от местной временной зоны или сезонных изменений времени.

Формат задания настройки — string.

Значение по умолчанию — LOCAL.

10.1.1.3 log_to_stdout

Параметр позволяет отправлять логи в стандартный вывод (stdout).

Формат задания настройки — boolean.

Значение по умолчанию — true

10.1.1.4 log_to_file

Параметр позволяет сохранять логи в файл. Директория с файлами логов указывается в параметре «folder_with_logs».

Формат задания настройки — boolean.

Значение по умолчанию — false.

10.1.1.5 folder_with_logs

Параметр задает директорию, в которой хранятся логи. Относительный путь начинается с каталога с приложением.

Для использования данного параметра требуется включить параметр «log_to_file».

Формат задания настройки — string.

Значение по умолчанию — ./

Пример:

```
"folder_with_logs": "/srv/logs"
```

10.1.1.6 max_log_file_size

Параметр задает максимальный размер файла лога в МБ перед выполнением его ротации (0 — не использовать ротацию).

Для использования данного параметра требуется включить параметр «log_to_file».

При необходимости можно настроить ротацию логов Docker. См. раздел «Настройка ротации логов Docker» в руководстве по установке LUNA PLATFORM.

Формат задания настройки — integer.

Значение по умолчанию — 1024

10.1.1.7 multiline_stack_trace

Параметр включает многострочную трассировку стека в логах. Когда параметр включен, информация о стеке вызовов записывается в логах так, что каждый фрейм стека помещается на отдельной строке, что улучшает читаемость. Если параметр выключен, информация о стеке вызовов записывается на одной строке, что может сделать логи менее удобными для анализа.

Формат задания настройки — boolean.

Значение по умолчанию — true.

10.1.1.8 format

Параметр определяет формат выводимых логов. Доступны следующие значения:

- «default» — стандартный формат вывода логов LUNA PLATFORM
- «json» — вывод логов в формате json
- «ecs» — вывод логов в формате ECS (Elastic Common Schema)

При использовании значения «ecs» будут использоваться следующие поля:

- «http.response.status_code» — содержит код состояния ответа HTTP (200, 404, 500 и т.д.);
- «http.response.execution_time» — содержит информацию о времени, затраченном на выполнение запроса и получение ответа;
- «http.request.method» — содержит метод HTTP-запроса (GET, POST, PUT и т.д.);
- «url.path» — содержит путь в URL-адресе запроса;
- «error.code» — содержит код ошибки, если запрос завершается с ошибкой.

Формат задания настройки — `string`.

Значение по умолчанию — `default`.

10.1.2 Группа параметров LIM_MANAGER_INDEXING

В данном разделе задаются настройки индексирования, задаваемые сервисом Index Manager.

10.1.2.1 indexer_origins

Параметр задает список адресов запущенных экземпляров сервиса Indexer.

Формат задания настройки — `array > string`.

Значение по умолчанию — `http://127.0.0.1:5180`.

10.1.2.2 planning_period

Параметр задает период [процедуры планирования](#), которая осуществляет проверку наборов списков, которые требуется проиндексировать.

Если задан параметр «[planning_schedule](#)», то параметр «`planning_period`» будет игнорироваться.

Если заданы оба параметра, то в приоритете будет параметр «`planning_schedule`».

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — `600`.

10.1.2.3 planning_schedule

Параметр задает расписание выполнения [процедуры планирования](#) в Cron-формате, которая осуществляет проверку наборов списков, которые требуется проиндексировать.

Использование расписания дает точное знание о времени запуска индексирования по сравнению с периодическим запуском каждые несколько часов или минут. Это обеспечивает более предсказуемое и контролируемое выполнение индексации, а также позволяет избежать сдвигов во времени, которые могут возникать при периодическом запуске. Например, если сервис недоступен или возникли другие проблемы, периодический запуск может быть пропущен, что может нарушить

планы обновления индекса. Однако, стоит учитывать, что периодический запуск имеет свои преимущества. Например, если требуется обновлять индекс каждый час без необходимости настройки сложного расписания.

Номер недели в Cron-формате начинается с воскресенья. Например, значение «0 0 * * 0» означает, что индексирование будет выполняться в 00:00 каждое воскресенье.

Если задан параметр «`planning_schedule`», то параметр «`planning_period`» будет игнорироваться.

Если заданы оба параметра, то в приоритете будет параметр «`planning_schedule`».

Формат задания настройки — `string` (Cron-подобная строка).

Значение по умолчанию не задано.

10.1.2.4 `lookup_period`

Параметр задает период [процедуры поиска](#), которая осуществляет проверку статусов всех запущенных экземпляров Indexer.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 5.

10.1.2.5 `face_lists > min_indexing_list_size`

Параметр задает минимальное количество лиц в списках для того, чтобы список был проиндексирован.

Параметр используется только при использовании параметра «`indexing_lists`» со значением «dynamic».

Формат задания настройки — `integer`.

Значение по умолчанию — 50000.

10.1.2.6 `face_lists > indexing_lists`

Параметр задает набор списков для индексирования.

Можно либо явно указать списки, либо указать значение «dynamic». В последнем случае будут обработаны все списки, количество биометрических шаблонов лиц в которых превышает количество, заданное в параметре «`min_indexing_list_size`».

Формат задания настройки — `array > string`.

Значение по умолчанию — `dynamic`.

10.1.2.7 ef_construction

Параметр задает ограничение на рассматриваемое число ближайших соседей при построении индекса.

Более высокие значения приводят к более точному графу, но требующему больше времени для построения.

Рекомендуется изменять параметр совместно с параметром «ef_search» сервиса Indexed Matcher.

Формат задания настройки — integer.

Значение по умолчанию — 1600.

10.1.2.8 rebuild_rules > default

Параметр задает включает полное перестроение индекса вместо его обновления.

См. подробную информацию в разделе «Настройка перестроения и обновления индекса».

Формат задания настройки — boolean.

Значение по умолчанию — true.

10.1.2.9 rebuild_rules > max_removal_for_rebuild

Параметр задает допустимое количество удалений биометрических шаблонов с момента создания индекса.

См. подробную информацию в разделе «Настройка перестроения и обновления индекса».

Формат задания настройки — integer.

Значение по умолчанию — 0.

10.1.3 Группа параметров LIM_MANAGER_HTTP_SETTINGS

В данной группе параметров содержатся настройки, отвечающие за обработку HTTP-подключений.

См. подробную информацию по следующей ссылке: <https://sanic.dev/en/guide/deployment/configuration.html#builtin-values>

10.1.3.1 request_timeout

Параметр задает продолжительность времени между моментом, когда новое открытое TCP-соединение передается на сервер, и моментом, когда получен весь HTTP-запрос.

Формат задания настройки — integer (секунды).

Значение по умолчанию — 60.

10.1.3.2 `response_timeout`

Параметр задает продолжительность времени между моментом, когда сервер передает HTTP-запрос приложению, и моментом, когда HTTP-ответ отправляется клиенту.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 600.

10.1.3.3 `request_max_size`

Параметр задает максимальный размер запроса.

Формат задания настройки — `integer` (байты).

Значение по умолчанию — 1073741824.

10.1.3.4 `keep_alive_timeout`

Параметр задает тайм-аут поддержания активности HTTP.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 15.

10.1.4 Группа параметров `LIM_MANAGER_DB`

В данной группе параметров задаются настройки подключения к базе данных сервиса Index Manager.

10.1.4.1 `db_user`

Параметр задает имя пользователя базы данных Redis.

Формат задания настройки — `string`.

Значение по умолчанию не задано.

10.1.4.2 `db_password`

Параметр задает пароль пользователя базы данных Redis.

Формат задания настройки — `string`.

Значение по умолчанию не задано.

10.1.4.3 db_host

Параметр задает имя сервера (хост) базы данных Redis.

Формат задания настройки — `string`.

Значение по умолчанию — `127.0.0.1`.

10.1.4.4 db_port

Параметр задает порт базы данных Redis.

Формат задания настройки — `string`.

Значение по умолчанию — `6379`.

10.1.4.5 db_settings > connection_pool_size

Параметр задает размер пула соединений к БД Redis.

Формат задания настройки — `string`.

Значение по умолчанию — `100`.

10.1.4.6 db_number

Параметр задает номер базы данных Redis. Каждый номер соответствует отдельной базе данных, что позволяет разделить данные.

Формат задания настройки — `integer`.

Значение по умолчанию — `0`.

10.1.4.7 sentinel > master_name

Параметр задает имя мастера базы данных Redis, который контролируется и управляется системой Sentinel.

Формат задания настройки — `string`.

Значение по умолчанию — `index_manager`.

10.1.4.8 sentinel > sentinels

Параметр задает список адресов и портов серверов Sentinel, которые будут использоваться клиентами для обнаружения и мониторинга БД Redis.

Формат задания настройки — `list > string`.

Значение по умолчанию — `[]`.

10.1.4.9 sentinel > user

Параметр задает имя пользователя сервера Sentinel.

Формат задания настройки — string.

Значение по умолчанию не задано.

10.1.4.10 sentinel > password

Параметр задает пароль пользователя сервера Sentinel.

Формат задания настройки — string.

Значение по умолчанию не задано.

10.1.5 Группа параметров INFLUX_MONITORING

Данная группа параметров задает настройки мониторинга.

См. подробную информацию о мониторинге в разделе [«Мониторинг»](#).

10.1.5.1 send_data_for_monitoring

Параметр позволяет включить или отключить отправку данных для мониторинга в InfluxDB.

Формат задания настройки — integer.

Значение по умолчанию — 1.

10.1.5.2 use_ssl

Параметр позволяет использовать HTTPS для подключения к InfluxDB.

Формат задания настройки — integer.

Значение по умолчанию — 0.

10.1.5.3 organization

Параметр задает рабочую область InfluxDB 2.x.

Формат задания настройки — string.

Значение по умолчанию — luna.

10.1.5.4 token

Параметр задает аутентификации InfluxDB 2.x.

Формат задания настройки — string.

10.1.5.5 bucket

Параметр задаёт имя бакета InfluxDB 2.x.

Формат задания настройки — `string`.

Значение по умолчанию — `luna_monitoring`.

10.1.5.6 host

Параметр задаёт IP-адрес InfluxDB.

Формат задания настройки — `string`.

Значение по умолчанию — `127.0.0.1`.

10.1.5.7 port

Параметр задает порт InfluxDB.

Формат задания настройки — `string`.

Значение по умолчанию — `8086`.

10.1.5.8 flushing_period

Параметр задает частоту отправки данных мониторинга в InfluxDB.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — `1`.

10.1.6 Группа параметров LUNA_FACES_ADDRESS

Данная группа параметров задает настройки подключения к сервису Faces.

10.1.6.1 origin

Параметр задает протокол, IP адрес и порт сервиса Faces.

IP адрес «127.0.0.1» означает, что будет использоваться сервис Faces, расположенный на сервере с Configurator. Если сервис находится на ином сервере, то в данном параметре нужно указать корректный IP адрес сервера с запущенным сервисом Faces.

Формат задания настройки — `string`.

Значение по умолчанию — `http://127.0.0.1:5030`.

10.1.6.2 `api_version`

Параметр задает версию API сервиса Faces. Доступная версия API — «3».

Формат задания настройки — `integer`.

Значение по умолчанию — 3.

10.1.7 Группа параметров `LUNA_FACES_TIMEOUTS`

Данная группа параметров определяет временные интервалы для управления таймаутами HTTP-запросов, которые направляются к сервису Faces.

10.1.7.1 `connect`

Параметр задает таймаут для установки соединения при отправке HTTP-запроса к сервису Faces. Это время ожидания, в течение которого клиент пытается установить соединение с сервисом.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 20.

10.1.7.2 `request`

Параметр задает общий таймаут для выполнения всего HTTP-запроса. Он включает в себя время установки соединения, отправки запроса, получения ответа и закрытия соединения. Если весь процесс занимает больше времени, чем указано в этом параметре, то запрос будет прерван.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 60.

10.1.7.3 `sock_connect`

Параметр задает таймаут для установки соединения на уровне сокетов. Если соединение на уровне сокетов не устанавливается в установленное время, операция будет прервана.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 10.

10.1.7.4 `sock_read`

Параметр задает таймаут на чтение данных с сокета после успешного соединения. Если данные не поступают в установленное время, операция чтения будет прервана.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 60.

10.1.8 Группа параметров LUNA_SERVICE_METRICS

Данная группа параметров включает и настраивает сбор метрик в формате Prometheus.

См. подробную информацию в разделе [«Мониторинг»](#).

10.1.8.1 enabled

Параметр включает сбор метрик.

Если сбор метрик отключен, то запрос к ресурсу `/metrics` вернет соответствующее сообщение.

Формат задания настройки — `boolean`.

Значение по умолчанию — `false`.

10.1.8.2 metrics_format

Параметр задает формат метрик.

На данный момент поддерживается только формат Prometheus.

См. [официальную документацию Prometheus](#) для более подробной информации.

Формат задания настройки — `string`.

Значение по умолчанию — `prometheus`.

10.1.8.3 extra_labels

Параметр задает пользовательские типы лейблов.

Формат задания настройки — `label_name=label_value`.

Значение по умолчанию не задано.

10.1.9 Прочие

10.1.9.1 index_storage_type

Параметр задает тип хранилища индексов. В настоящее время доступен только вариант «LOCAL».

Значение «LOCAL» означает, что индексы будут храниться в директории, указанной в параметре [«index_storage_local»](#).

Формат задания настройки — `string`.

Значение по умолчанию — `LOCAL`.

10.1.9.2 index_storage_local

Параметр задает директорию для хранения индексов при [типе хранилища](#) «LOCAL».

Формат задания настройки — string.

Значение по умолчанию — ./local_storage.

10.1.9.3 storage_time

Параметр задает формат времени, используемый для записей в базе данных. Доступны следующие значения:

- «LOCAL» — отображает местное время системы, на которой выполняется запись логов;
- «UTC» — отображает координированное всемирное время, которое является стандартом времени и не зависит от местной временной зоны или сезонных изменений времени.

Формат задания настройки — string.

Значение по умолчанию — LOCAL.

10.1.9.4 lim_manager_active_plugins

Параметр задает список плагинов, которые должен использовать сервис.

Имена задаются в следующем формате:

```
[  
    "plugin_1",  
    "plugin_2",  
    "plugin_3"  
]
```

Список должен содержать имена файлов без расширения (.py).

Формат задания настройки — integer.

Значение по умолчанию — 1.

10.1.9.5 default_face_descriptor_version

Параметр задает используемую версию биометрического шаблона лица.

Формат задания настройки — string.

Значение по умолчанию — 59.

10.2 Настройки сервиса Indexed Matcher

Данный раздел описывает параметры сервиса Indexed Matcher.

Настройку сервиса можно выполнить с помощью сервиса Configurator.

10.2.1 Группа параметров LIM_MATCHING

В данной группе параметров задаются настройки индексированного сравнения.

10.2.1.1 ef_search

Параметр задает ограничение на рассматриваемое число ближайших соседей при поиске индекса.

Более высокие значения приводят к более точному, но медленному поиску.

Рекомендуется изменять параметр совместно с параметром «ef_construction» сервиса Indexed Manager.

Формат задания настройки — integer.

Значение по умолчанию — 1600.

10.2.2 Группа параметров LIM_MATCHER_REFRESH

В данной группе параметров задаются настройки [обновления индекса в памяти](#) сервиса Indexed Matcher.

10.2.2.1 enabled

Параметр позволяет включить [обновление индекса в памяти](#).

Формат задания настройки — integer.

Значение по умолчанию — 1.

10.2.3 Группа параметров LIM_MATCHER_LOGGER

Данная группа параметров задает настройки логирования.

10.2.3.1 log_level

Параметр задает уровень отладочной печати, по приоритету: «ERROR», «WARNING», «INFO», «DEBUG».

Формат задания настройки — string.

Значение по умолчанию — INFO.

10.2.3.2 log_time

Параметр задает формат времени, используемый в записях лога. Доступны следующие значения:

- «LOCAL» — отображает местное время системы, на которой выполняется запись логов;
- «UTC» — отображает координированное всемирное время, которое является стандартом времени и не зависит от местной временной зоны или сезонных изменений времени.

Формат задания настройки — `string`.

Значение по умолчанию — `LOCAL`.

10.2.3.3 log_to_stdout

Параметр позволяет отправлять логи в стандартный вывод (`stdout`).

Формат задания настройки — `boolean`.

Значение по умолчанию — `true`

10.2.3.4 log_to_file

Параметр позволяет сохранять логи в файл. Директория с файлами логов указывается в параметре «`folder_with_logs`».

Формат задания настройки — `boolean`.

Значение по умолчанию — `false`.

10.2.3.5 folder_with_logs

Параметр задает директорию, в которой хранятся логи. Относительный путь начинается с каталога с приложением.

Для использования данного параметра требуется включить параметр «`log_to_file`».

Формат задания настройки — `string`.

Значение по умолчанию — `./`

Пример:

```
"folder_with_logs": "/srv/logs"
```

10.2.3.6 max_log_file_size

Параметр задает максимальный размер файла лога в МБ перед выполнением его ротации (0 — не использовать ротацию).

Для использования данного параметра требуется включить параметр «`log_to_file`».

При необходимости можно настроить ротацию логов Docker. См. раздел «Настройка ротации логов Docker» в руководстве по установке LUNA PLATFORM.

Формат задания настройки — `integer`.

Значение по умолчанию — `1024`

10.2.3.7 `multiline_stack_trace`

Параметр включает многострочную трассировку стека в логах. Когда параметр включен, информация о стеке вызовов записывается в логах так, что каждый фрейм стека помещается на отдельной строке, что улучшает читаемость. Если параметр выключен, информация о стеке вызовов записывается на одной строке, что может сделать логи менее удобными для анализа.

Формат задания настройки — `boolean`.

Значение по умолчанию — `true`.

10.2.3.8 `format`

Параметр определяет формат выводимых логов. Доступны следующие значения:

- «default» — стандартный формат вывода логов LUNA PLATFORM
- «json» — вывод логов в формате json
- «ecs» — вывод логов в формате ECS (Elastic Common Schema)

При использовании значения «ecs» будут использоваться следующие поля:

- «`http.response.status_code`» — содержит код состояния ответа HTTP (200, 404, 500 и т.д.);
- «`http.response.execution_time`» — содержит информацию о времени, затраченном на выполнение запроса и получение ответа;
- «`http.request.method`» — содержит метод HTTP-запроса (GET, POST, PUT и т.д.);
- «`url.path`» — содержит путь в URL-адресе запроса;
- «`error.code`» — содержит код ошибки, если запрос завершается с ошибкой.

Формат задания настройки — `string`.

Значение по умолчанию — `default`.

10.2.4 Группа параметров `LIM_MATCHER_HTTP_SETTINGS`

В данной группе параметров содержатся настройки, отвечающие за обработку HTTP-подключений. См. подробную информацию по следующей ссылке: <https://sanic.dev/en/guide/deployment/configuration.html#builtin-values>

10.2.4.1 request_timeout

Параметр задает продолжительность времени между моментом, когда новое открытое TCP-соединение передается на сервер, и моментом, когда получен весь HTTP-запрос.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 60.

10.2.4.2 response_timeout

Параметр задает продолжительность времени между моментом, когда сервер передает HTTP-запрос приложению, и моментом, когда HTTP-ответ отправляется клиенту.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 600.

10.2.4.3 request_max_size

Параметр задает максимальный размер запроса.

Формат задания настройки — `integer` (байты).

Значение по умолчанию — 1073741824.

10.2.4.4 keep_alive_timeout

Параметр задает тайм-аут поддержания активности HTTP.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 15.

10.2.5 Группа параметров LIM_MATCHER_DB

В данной группе параметров задаются настройки подключения к базе данных сервиса Index Matcher.

10.2.5.1 db_user

Параметр задает имя пользователя базы данных Redis.

Формат задания настройки — `string`.

Значение по умолчанию не задано.

10.2.5.2 db_password

Параметр задает пароль пользователя базы данных Redis.

Формат задания настройки — `string`.

Значение по умолчанию не задано.

10.2.5.3 db_host

Параметр задает имя сервера (хост) базы данных Redis.

Формат задания настройки — `string`.

Значение по умолчанию — `127.0.0.1`.

10.2.5.4 db_port

Параметр задает порт базы данных Redis.

Формат задания настройки — `string`.

Значение по умолчанию — `6379`.

10.2.5.5 db_settings > connection_pool_size

Параметр задает размер пула соединений к БД Redis.

Формат задания настройки — `string`.

Значение по умолчанию — `100`.

10.2.5.6 db_number

Параметр задает номер базы данных Redis. Каждый номер соответствует отдельной базе данных, что позволяет разделить данные.

Формат задания настройки — `integer`.

Значение по умолчанию — `0`.

10.2.5.7 sentinel > master_name

Параметр задает имя мастера базы данных Redis, который контролируется и управляется системой Sentinel.

Формат задания настройки — `string`.

Значение по умолчанию — `indexed_matcher`.

10.2.5.8 sentinel > sentinels

Параметр задает список адресов и портов серверов Sentinel, которые будут использоваться клиентами для обнаружения и мониторинга БД Redis.

Формат задания настройки — `list > string`.

Значение по умолчанию — `[]`.

10.2.5.9 sentinel > user

Параметр задает имя пользователя сервера Sentinel.

Формат задания настройки — `string`.

Значение по умолчанию не задано.

10.2.5.10 sentinel > password

Параметр задает пароль пользователя сервера Sentinel.

Формат задания настройки — `string`.

Значение по умолчанию не задано.

10.2.6 Группа параметров INFLUX_MONITORING

Данная группа параметров задает настройки мониторинга.

См. подробную информацию о мониторинге в разделе [«Мониторинг»](#).

10.2.6.1 send_data_for_monitoring

Параметр позволяет включить или отключить отправку данных для мониторинга в InfluxDB.

Формат задания настройки — `integer`.

Значение по умолчанию — `1`.

10.2.6.2 use_ssl

Параметр позволяет использовать HTTPS для подключения к InfluxDB.

Формат задания настройки — `integer`.

Значение по умолчанию — `0`.

10.2.6.3 organization

Параметр задает рабочую область InfluxDB 2.x.

Формат задания настройки — `string`.

Значение по умолчанию — `luna`.

10.2.6.4 token

Параметр задает аутентификации InfluxDB 2.x.

Формат задания настройки — `string`.

10.2.6.5 bucket

Параметр задаёт имя бакета InfluxDB 2.x.

Формат задания настройки — `string`.

Значение по умолчанию — `luna_monitoring`.

10.2.6.6 host

Параметр задаёт IP-адрес InfluxDB.

Формат задания настройки — `string`.

Значение по умолчанию — `127.0.0.1`.

10.2.6.7 port

Параметр задает порт InfluxDB.

Формат задания настройки — `string`.

Значение по умолчанию — `8086`.

10.2.6.8 flushing_period

Параметр задает частоту отправки данных мониторинга в InfluxDB.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — `1`.

10.2.7 Группа параметров LUNA_FACES_ADDRESS

Данная группа параметров задает настройки подключения к сервису Faces.

10.2.7.1 origin

Параметр задает протокол, IP адрес и порт сервиса Faces.

IP адрес «127.0.0.1» означает, что будет использоваться сервис Faces, расположенный на сервере с Configurator. Если сервис находится на ином сервере, то в данном параметре нужно указать корректный IP адрес сервера с запущенным сервисом Faces.

Формат задания настройки — `string`.

Значение по умолчанию — `http://127.0.0.1:5030`.

10.2.7.2 api_version

Параметр задает версию API сервиса Faces. Доступная версия API — «3».

Формат задания настройки — `integer`.

Значение по умолчанию — 3.

10.2.8 Группа параметров LUNA_FACES_TIMEOUTS

Данная группа параметров определяет временные интервалы для управления таймаутами HTTP-запросов, которые направляются к сервису Faces.

10.2.8.1 connect

Параметр задает таймаут для установки соединения при отправке HTTP-запроса к сервису Faces. Это время ожидания, в течение которого клиент пытается установить соединение с сервисом.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 20.

10.2.8.2 request

Параметр задает общий таймаут для выполнения всего HTTP-запроса. Он включает в себя время установки соединения, отправки запроса, получения ответа и закрытия соединения. Если весь процесс занимает больше времени, чем указано в этом параметре, то запрос будет прерван.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 60.

10.2.8.3 sock_connect

Параметр задает таймаут для установки соединения на уровне сокетов. Если соединение на уровне сокетов не устанавливается в установленное время, операция будет прервана.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 10.

10.2.8.4 `sock_read`

Параметр задает таймаут на чтение данных с сокета после успешного соединения. Если данные не поступают в установленное время, операция чтения будет прервана.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 60.

10.2.9 Группа параметров `LUNA_LICENSES_ADDRESS`

Данная группа параметров задает настройки подключения к сервису Licenses.

10.2.9.1 `origin`

Параметр задает протокол, IP адрес и порт сервиса Licenses.

IP адрес «127.0.0.1» означает, что будет использоваться сервис Licenses, расположенный на сервере с Configurator. Если сервис находится на ином сервере, то в данном параметре нужно указать корректный IP адрес сервера с запущенным сервисом Licenses.

Формат задания настройки — `string`.

Значение по умолчанию — `http://127.0.0.1:5120`.

10.2.9.2 `api_version`

Параметр задает версию API сервиса Licenses. Доступная версия API — «1».

Формат задания настройки — `integer`.

Значение по умолчанию — 1.

10.2.10 Группа параметров `LUNA_SERVICE_METRICS`

Данная группа параметров включает и настраивает сбор метрик в формате Prometheus.

См. подробную информацию в разделе [«Мониторинг»](#).

10.2.10.1 `enabled`

Параметр включает сбор метрик.

Если сбор метрик отключен, то запрос к ресурсу `/metrics` вернет соответствующее сообщение.

Формат задания настройки — `boolean`.

Значение по умолчанию — `false`.

10.2.10.2 `metrics_format`

Параметр задает формат метрик.

На данный момент поддерживается только формат Prometheus.

См. [официальную документацию Prometheus](#) для более подробной информации.

Формат задания настройки — `string`.

Значение по умолчанию — `prometheus`.

10.2.10.3 `extra_labels`

Параметр задает пользовательские типы лейблов.

Формат задания настройки — `label_name=label_value`.

Значение по умолчанию не задано.

10.2.11 Прочие

10.2.11.1 `lim_matcher_cache`

Параметр задает путь до директории с кэшем.

См. подробную информацию о кэшировании в разделе [«Кэширование индекса»](#).

Для отключения кэширования необходимо оставить поле пустым.

Формат задания настройки — `string`.

Значение по умолчанию не задано.

10.2.11.2 `index_storage_type`

Параметр задает тип хранилища индексов. В настоящее время доступен только вариант «LOCAL».

Значение «LOCAL» означает, что индексы будут храниться в директории, указанной в параметре [«`index_storage_local`»](#).

Формат задания настройки — `string`.

Значение по умолчанию — `LOCAL`.

10.2.11.3 index_storage_local

Параметр задает директорию для хранения индексов при [типе хранилища](#) «LOCAL».

Формат задания настройки — string.

Значение по умолчанию — ./local_storage.

10.2.11.4 lim_matcher_active_plugins

Параметр задает список плагинов, которые должен использовать сервис.

Имена задаются в следующем формате:

```
[  
    "plugin_1",  
    "plugin_2",  
    "plugin_3"  
]
```

Список должен содержать имена файлов без расширения (.py).

Формат задания настройки — integer.

Значение по умолчанию — 1.

10.2.11.5 default_face_descriptor_version

Параметр задает используемую версию биометрического шаблона лица.

Формат задания настройки — string.

Значение по умолчанию — 59.

10.3 Настройки сервиса Indexer

Данный раздел описывает параметры сервиса Indexer.

Настройку сервиса можно выполнить с помощью сервиса Configurator.

10.3.1 Группа параметров LIM_INDEXER_LOGGER

Данная группа параметров задает настройки логирования.

10.3.1.1 log_level

Параметр задает уровень отладочной печати, по приоритету: «ERROR», «WARNING», «INFO», «DEBUG».

Формат задания настройки — string.

Значение по умолчанию — INFO.

10.3.1.2 log_time

Параметр задает формат времени, используемый в записях лога. Доступны следующие значения:

- «LOCAL» — отображает местное время системы, на которой выполняется запись логов;
- «UTC» — отображает координированное всемирное время, которое является стандартом времени и не зависит от местной временной зоны или сезонных изменений времени.

Формат задания настройки — string.

Значение по умолчанию — LOCAL.

10.3.1.3 log_to_stdout

Параметр позволяет отправлять логи в стандартный вывод (stdout).

Формат задания настройки — boolean.

Значение по умолчанию — true

10.3.1.4 log_to_file

Параметр позволяет сохранять логи в файл. Директория с файлами логов указывается в параметре «folder_with_logs».

Формат задания настройки — boolean.

Значение по умолчанию — false.

10.3.1.5 folder_with_logs

Параметр задает директорию, в которой хранятся логи. Относительный путь начинается с каталога с приложением.

Для использования данного параметра требуется включить параметр «log_to_file».

Формат задания настройки — string.

Значение по умолчанию — ./

Пример:

```
"folder_with_logs": "/srv/logs"
```

10.3.1.6 max_log_file_size

Параметр задает максимальный размер файла лога в МБ перед выполнением его ротации (0 — не использовать ротацию).

Для использования данного параметра требуется включить параметр «log_to_file».

При необходимости можно настроить ротацию логов Docker. См. раздел «Настройка ротации логов Docker» в руководстве по установке LUNA PLATFORM.

Формат задания настройки — integer.

Значение по умолчанию — 1024

10.3.1.7 multiline_stack_trace

Параметр включает многострочную трассировку стека в логах. Когда параметр включен, информация о стеке вызовов записывается в логах так, что каждый фрейм стека помещается на отдельной строке, что улучшает читаемость. Если параметр выключен, информация о стеке вызовов записывается на одной строке, что может сделать логи менее удобными для анализа.

Формат задания настройки — boolean.

Значение по умолчанию — true.

10.3.1.8 format

Параметр определяет формат выводимых логов. Доступны следующие значения:

- «default» — стандартный формат вывода логов LUNA PLATFORM
- «json» — вывод логов в формате json
- «ecs» — вывод логов в формате ECS (Elastic Common Schema)

При использовании значения «ecs» будут использоваться следующие поля:

- «`http.response.status_code`» — содержит код состояния ответа HTTP (200, 404, 500 и т.д.);
- «`http.response.execution_time`» — содержит информацию о времени, затраченном на выполнение запроса и получение ответа;
- «`http.request.method`» — содержит метод HTTP-запроса (GET, POST, PUT и т.д.);
- «`url.path`» — содержит путь в URL-адресе запроса;
- «`error.code`» — содержит код ошибки, если запрос завершается с ошибкой.

Формат задания настройки — `string`.

Значение по умолчанию — `default`.

10.3.2 Группа параметров `INFLUX_MONITORING`

Данная группа параметров задает настройки мониторинга.

См. подробную информацию о мониторинге в разделе [«Мониторинг»](#).

10.3.2.1 `send_data_for_monitoring`

Параметр позволяет включить или отключить отправку данных для мониторинга в InfluxDB.

Формат задания настройки — `integer`.

Значение по умолчанию — `1`.

10.3.2.2 `use_ssl`

Параметр позволяет использовать HTTPS для подключения к InfluxDB.

Формат задания настройки — `integer`.

Значение по умолчанию — `0`.

10.3.2.3 `organization`

Параметр задает рабочую область InfluxDB 2.x.

Формат задания настройки — `string`.

Значение по умолчанию — `luna`.

10.3.2.4 `token`

Параметр задает аутентификации InfluxDB 2.x.

Формат задания настройки — `string`.

10.3.2.5 bucket

Параметр задаёт имя бакета InfluxDB 2.x.

Формат задания настройки — `string`.

Значение по умолчанию — `luna_monitoring`.

10.3.2.6 host

Параметр задаёт IP-адрес InfluxDB.

Формат задания настройки — `string`.

Значение по умолчанию — `127.0.0.1`.

10.3.2.7 port

Параметр задает порт InfluxDB.

Формат задания настройки — `string`.

Значение по умолчанию — `8086`.

10.3.2.8 flushing_period

Параметр задает частоту отправки данных мониторинга в InfluxDB.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — `1`.

10.3.3 Группа параметров LUNA_FACES_ADDRESS

Данная группа параметров задает настройки подключения к сервису Faces.

10.3.3.1 origin

Параметр задает протокол, IP адрес и порт сервиса Faces.

IP адрес «127.0.0.1» означает, что будет использоваться сервис Faces, расположенный на сервере с Configurator. Если сервис находится на ином сервере, то в данном параметре нужно указать корректный IP адрес сервера с запущенным сервисом Faces.

Формат задания настройки — `string`.

Значение по умолчанию — `http://127.0.0.1:5030`.

10.3.3.2 `api_version`

Параметр задает версию API сервиса Faces. Доступная версия API — «3».

Формат задания настройки — `integer`.

Значение по умолчанию — 3.

10.3.4 Группа параметров `LUNA_FACES_TIMEOUTS`

Данная группа параметров определяет временные интервалы для управления таймаутами HTTP-запросов, которые направляются к сервису Faces.

10.3.4.1 `connect`

Параметр задает таймаут для установки соединения при отправке HTTP-запроса к сервису Faces. Это время ожидания, в течение которого клиент пытается установить соединение с сервисом.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 20.

10.3.4.2 `request`

Параметр задает общий таймаут для выполнения всего HTTP-запроса. Он включает в себя время установки соединения, отправки запроса, получения ответа и закрытия соединения. Если весь процесс занимает больше времени, чем указано в этом параметре, то запрос будет прерван.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 60.

10.3.4.3 `sock_connect`

Параметр задает таймаут для установки соединения на уровне сокетов. Если соединение на уровне сокетов не устанавливается в установленное время, операция будет прервана.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 10.

10.3.4.4 `sock_read`

Параметр задает таймаут на чтение данных с сокета после успешного соединения. Если данные не поступают в установленное время, операция чтения будет прервана.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 60.

10.3.5 Группа параметров LIM_INDEXER_HTTP_SETTINGS

В данной группе параметров содержатся настройки, отвечающие за обработку HTTP-подключений. См. подробную информацию по следующей ссылке: <https://sanic.dev/en/guide/deployment/configuration.html#builtin-values>

10.3.5.1 request_timeout

Параметр задает продолжительность времени между моментом, когда новое открытое TCP-соединение передается на сервер, и моментом, когда получен весь HTTP-запрос.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 60.

10.3.5.2 response_timeout

Параметр задает продолжительность времени между моментом, когда сервер передает HTTP-запрос приложению, и моментом, когда HTTP-ответ отправляется клиенту.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 600.

10.3.5.3 request_max_size

Параметр задает максимальный размер запроса.

Формат задания настройки — `integer` (байты).

Значение по умолчанию — 1073741824.

10.3.5.4 keep_alive_timeout

Параметр задает тайм-аут поддержания активности HTTP.

Формат задания настройки — `integer` (секунды).

Значение по умолчанию — 15.

10.3.6 Группа параметров LUNA_SERVICE_METRICS

Данная группа параметров включает и настраивает сбор метрик в формате Prometheus.

См. подробную информацию в разделе «Мониторинг».

10.3.6.1 `enabled`

Параметр включает сбор метрик.

Если сбор метрик отключен, то запрос к ресурсу `/metrics` вернет соответствующее сообщение.

Формат задания настройки — `boolean`.

Значение по умолчанию — `false`.

10.3.6.2 `metrics_format`

Параметр задает формат метрик.

На данный момент поддерживается только формат Prometheus.

См. [официальную документацию Prometheus](#) для более подробной информации.

Формат задания настройки — `string`.

Значение по умолчанию — `prometheus`.

10.3.6.3 `extra_labels`

Параметр задает пользовательские типы лейблов.

Формат задания настройки — `label_name=label_value`.

Значение по умолчанию не задано.

10.3.7 Прочие

10.3.7.1 `index_storage_type`

Параметр задает тип хранилища индексов. В настоящее время доступен только вариант «LOCAL».

Значение «LOCAL» означает, что индексы будут храниться в директории, указанной в параметре [«index_storage_local»](#).

Формат задания настройки — `string`.

Значение по умолчанию — `LOCAL`.

10.3.7.2 `index_storage_local`

Параметр задает директорию для хранения индексов при [типе хранилища](#) «LOCAL».

Формат задания настройки — `string`.

Значение по умолчанию — `./local_storage`.

10.3.7.3 `lim_indexer_active_plugins`

Параметр задает список плагинов, которые должен использовать сервис.

Имена задаются в следующем формате:

```
[  
    "plugin_1",  
    "plugin_2",  
    "plugin_3"  
]
```

Список должен содержать имена файлов без расширения (.py).

Формат задания настройки — `integer`.

Значение по умолчанию — 1.

10.3.7.4 `default_face_descriptor_version`

Параметр задает используемую версию биометрического шаблона лица.

Формат задания настройки — `string`.

Значение по умолчанию — 59.

10.4 Настройки плагина сравнения

10.4.1 Группа параметров LUNA_INDEXED_LIST_PLUGIN

Данная группа параметров отвечает за соединение [плагина сравнения](#) с Redis при [высчитывании сложности запроса](#).

Доступна возможность указать адрес Redis Sentinel. См. раздел [«Использование Redis Sentinel»](#).

10.4.1.1 `redis_url`

Параметр задает адрес Redis.

Формат задания настройки — `string`.

Значение по умолчанию — `redis://localhost:6379`.

10.4.1.2 `request_timeout`

Параметр задает таймаут подключения к Redis.

Формат задания настройки — `integer`.

Значение по умолчанию — `60`.