Process Mining от Сбера

version 2.10.24

Руководство по установке продуктов "Process Mining от Сбера"

July 27, 2026

Содержание

Установка

1. Системные требования

1.1. Минимальные требования к конфигурации серверов

Название сервера	CPU	RAM	HDD
Сервер приложений (СП)	8	16	60
Сервер СУБД PostgreSQL	2	4	100
Сервер СУБД ClickHouse	8	32	100
Управляющий узел Ansible (CI / CD)	2	4	60

Примечание:

1. Компоненты `pm-ai` и `pm-agent` могут быть размещены на выделенных серверах либо на существующем сервере приложений при наличии достаточного объема вычислительных ресурсов.
2. Итоговые требования к CPU, RAM и HDD для компонентов `pm-ai` и `pm-agent` определяются выбранным сценарием установки, профилем нагрузки и используемой моделью/инференс-движком.

1.2. Требования к программному обеспечению

	Требуемое СПО	Версия СПО	download
ОС	CentOS, RHEL	7.X и выше	https://www.centos.org/download/ и https://access.redhat.com/downloads
	Debian	10.X и выше	https://www.debian.org/distrib/
	Ubuntu	20.X и выше	https://ubuntu.com/#download-ubuntu
	Astra Linux	1.8 и выше	https://astralinux.ru/os/server-astra/
	РЕД ОС	7.X и выше	https://redos.red-soft.ru/product/downloads/
Сервер приложений (СП)	Docker	27.1.1 и выше	https://docs.docker.com/engine/install/debian/
Серверы СУБД	PostgreSQL server	13.X	Поставляется в дистрибутиве
	ClickHouse server	23.X	Поставляется в дистрибутиве
Сервис кеширования Redis	Redis	7.0.12	Поставляется в дистрибутиве

	Требуемое СПО	Версия СПО	download
Управляющий узел Ansible (CI / CD)	Ansible	2.9 и выше	https://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html

1.3. Требования к сетевым доступам

Источник	Назначение	Порты
Рабочая станция пользователя	Сервер приложений	tcp/80, tcp/443
Сервер приложений	Сервер СУБД PostgreSQL	tcp/5432
Сервер приложений	Сервер СУБД ClickHouse	tcp/8123, tcp/9000
Управляющий сервер	Сервер приложений, Сервер СУБД	tcp/22
Рабочая станция администратора	Сервер приложений, Сервер СУБД, Управляющий сервер	tcp/22
Управляющий сервер	Сервер приложений	tcp/80, tcp/443, tcp/5432, tcp/8123, tcp/6379, tcp/8010, tcp/8020, tcp/8030, tcp/8040, tcp/8060, tcp/8070, tcp/8090, tcp/8000, tcp/9030
Сервер приложений / компонента продукта	Сервер pm-ai	tcp/9020, tcp/9040
Сервер приложений / компонента продукта	Подключение к pm-agent	tcp/9070

Примечание: Доступы к внешним AI-интеграциям, TLS-сертификатам и иным внешним контурам предоставляются в соответствии с целевой архитектурой решения.

Сетевая схема описана в Приложении 1 к данной инструкции.

1.4. Требования к DNS-записи

DNS
Запись типа «A», указывающая на сервер приложения (пример: spm.company.ru)

1.5. Прикладное ПО состоит из сервисов

Имя сервиса	Описание
pm-ui	Компонент FrontEnd (IAM)
pm-bi	Компонент для исследования и визуализации данных (BI)
pm-ml	Компонент Machine Learning для анализа пользовательских путей (ML)
pm-etl	Компонент ETL для извлечения, анализа и загрузки данных (ETL), Airflow webserver
pm-flow	Планировщик, Airflow scheduler
pm-auth	Компонент авторизации пользователей (Auth)

Имя сервиса	Описание
pm-tm	Компонент сервиса логирования task mining (TM)
redis	Компонент кеширования данных
pm-docs	Компонент документации (DOCS)
pm-agent	Компонент AI/LLM-обработки, обеспечивающий выполнение запросов ассистента, обработку контекста, генерацию ответов и интеграцию с моделями/AI-сервисами
pm-ai	Компонент прикладного ассистента, обеспечивающий взаимодействие пользовательского интерфейса и внутренних сервисов с AI-функциями системы

2. Установка программного обеспечения

2.1. Состав поставки

Поставка представляет собой набор архивов для каждого сервиса и СУБД, а также скрипты развертывания.
 \ Имена архивов формируются по следующим правилам:

- Приложения: `<Имя_сервиса>-<Версия_поставки>.tgz`
- СУБД: `postgres-server-<Версия_postgres>.tgz`, `clickhouse-server-<Версия_clickhouse>.tgz`
- Сервис кеширования данных: `redis-<Версия_redis>.tgz`
- Скрипты развертывания: `deploy-<Версия_поставки>.tar`

В состав поставки прикладных сервисов входят архивы компонентов продукта, включая:

- `pm-ui-<Версия_поставки>.tgz`
- `pm-bi-<Версия_поставки>.tgz`
- `pm-ml-<Версия_поставки>.tgz`
- `pm-etl-<Версия_поставки>.tgz`
- `pm-flow-<Версия_поставки>.tgz`
- `pm-auth-<Версия_поставки>.tgz`
- `pm-tm-<Версия_поставки>.tgz`
- `pm-docs-<Версия_поставки>.tgz`
- `pm-agent-<Версия_поставки>.tgz`
- `pm-ai-<Версия_поставки>.tgz`

Примечание: фактический состав поставки зависит от выбранного сценария установки, версии поставки и состава лицензируемых компонентов продукта.

2.2. Подготовка окружения

2.2.1. Настройка серверов СУБД и сервера приложений

На серверах СУБД и сервере приложений должен быть установлен и настроен Docker версии 27.x и выше.

Если Docker требуемой версии уже установлен, необходимо проверить права пользователя на работу с Docker в соответствии с п. 2.2.1.3.

2.2.1.1. Добавить репозиторий Docker

Необходимо добавить официальный репозиторий Docker для используемой операционной системы. Подробное описание приведено в официальной документации Docker.

2.2.1.2. Выполнить установку пакетов Docker

Для Debian/Ubuntu установка может быть выполнена командой:

```
apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-com
pose-plugin
```

Для RPM-based дистрибутивов допускается установка эквивалентных пакетов через `yum` или `dnf` в соответствии с официальной документацией Docker.

2.2.1.3. Проверить права пользователя на работу с Docker

Если установка производится не под пользователем `root`, то перед началом необходимо добавить этого пользователя в группу `docker`, выполнив команду:

```
usermod -aG docker <user_name>
```

где `<user_name>` — имя пользователя, под которым будет выполнена установка.

Критерием успешного выполнения шага является возможность пользователя `<user_name>` выполнять команды Docker без `sudo`, например, команда:

```
docker ps
```

должна успешно выполняться от имени пользователя `<user_name>` без использования `sudo`.

2.2.1.4. Подготовить размещение компонентов `pm-ai` и `pm-agent`

Компоненты `pm-ai` и `pm-agent` могут быть развернуты:

- на сервере приложений;
- на выделенных серверах;
- в смешанном варианте, при котором один из компонентов размещается на сервере приложений, а второй — на выделенном сервере.

Для корректного развертывания компонентов `pm-ai` и `pm-agent` должны быть обеспечены:

- установленный и настроенный Docker на целевых серверах размещения компонентов;
- сетевые доступы в соответствии с разделом 1.3;
- доступ пользователя установки к каталогам конфигурации и хранения образов;
- доступ к внутренним сервисам продукта и внешним интеграциям, если такие интеграции используются.

2.2.1.5. Подготовить каталоги конфигурации и хранения образов

На целевых серверах должны существовать каталоги, используемые для хранения конфигурационных файлов сервисов и файлов образов.

Значения каталогов задаются параметрами `app_config_folder` и `source_images_storage_folder` в файле `config.yml`.

Для пользователя, от имени которого выполняется автоматизированная установка, должны быть предоставлены права на создание каталогов и файлов в указанных директориях.

2.2.2. Настройка управляющего сервера (CI/CD)

2.2.2.1. Установить Ansible

На управляющем сервере должен быть установлен Ansible версии 2.9 или выше.

Ниже приведён пример установки Ansible с помощью Python `venv` и `pip`; также допускается выполнение установки с использованием пакетных менеджеров ОС.

- Создать виртуальное окружение Python, выполнив команду:

```
python3 -m venv venv
```

- Активировать виртуальное окружение:

```
source venv/bin/activate
```

- Установить пакет Ansible через `pip`:

```
pip install ansible
```

При необходимости допускается установка `ansible-core`, если состав поставки и используемые playbook не требуют дополнительных коллекций.

2.2.2.2. Настроить SSH-доступ с управляющего сервера

С управляющего сервера должен быть настроен доступ по протоколу SSH к серверу приложений, серверам СУБД, а также к серверам `pm-ai` и `pm-agent` при их размещении на отдельных узлах.

Настройка подключения возможна как по паролю, так и по SSH-ключу. Рекомендуемый способ — подключение по ключу.

Критерием корректной настройки является возможность с управляющего сервера подключиться к серверу приложений, серверам СУБД и серверам `pm-ai`/`pm-agent`, если они размещаются на отдельных узлах, выполнив команду:

```
ssh <user_name>@<server_ip>
```

где `<user_name>` — пользователь, от имени которого выполняется подключение, `<server_ip>` — IP-адрес сервера назначения.

Описание процесса настройки подключения к серверу по протоколу SSH:

- Сгенерировать пару SSH-ключей, выполнив команду:

```
ssh-keygen
```

- Далее необходимо скопировать публичный ключ на сервер приложений, серверы СУБД и, при необходимости, на серверы `pm-ai` и `pm-agent`. Это можно сделать, вручную скопировав публичный ключ в файл `~/.ssh/authorized_keys`, либо выполнив команду:

```
ssh-copy-id -i ~/.ssh/id_rsa.pub <user_name>@<server_ip>
```

где `<user_name>` — пользователь, от имени которого выполняется подключение, `<server_ip>` — IP-адрес сервера назначения.

- На управляющем сервере должен быть установлен пакет `sshpass` в случае использования подключения по паролю. Установка выполняется командой:

```
yum install sshpass
```

или

```
apt install sshpass
```

Выбор команды зависит от пакетного менеджера используемой ОС.

2.2.2.3. Получить файлы лицензии

Необходимо получить файлы лицензий `license.json` и `license.json.sig`.

2.2.2.4. Получить и распаковать дистрибутив

Необходимо получить архив с дистрибутивом, после чего скопировать его и распаковать на управляющем сервере.

2.2.2.5. Проверить доступность целевых узлов для автоматизированного развертывания

До начала установки необходимо убедиться, что:

- с управляющего сервера доступен сервер приложений;

- с управляющего сервера доступны серверы СУБД;
- с управляющего сервера доступны серверы `pm-ai` и `pm-agent`, если они развертываются на отдельных узлах;
- пользователь установки имеет права на выполнение команд Docker на всех целевых серверах;
- каталоги, указанные в `config.yml`, доступны для записи.

2.3. Установка приложения

2.3.1. На управляющем сервере скопировать файлы образов с расширением `.tgz` в удобный каталог, по умолчанию предполагается использование каталога `/opt/images`. Каталог задаётся параметром `source_images_storage_folder` в файле `config.yml` (описание параметров представлено в таблице 2 «Описание параметров файла `config.yml`»).

2.3.2. На управляющем сервере среди файлов из распакованного ранее дистрибутива найти файл `deploy-<version>.tar`, где `<version>` — версия поставки, например, `2.9.6`, после чего распаковать данный файл, выполнив команду:

```
tar -xvf deploy-<version>.tar
```

После распаковки файла `deploy-<version>.tar` появится каталог `deploy-<version>` с файлами деплоя приложения.

2.3.3. В каталоге `deploy-<version>` найти файл `config.yml` и скопировать его в каталог `/opt`, выполнив команду:

```
cp config.yml /opt/config.yml
```

Данный файл является шаблоном постоянного конфигурационного файла, заполняется единожды при первой установке и в дальнейшем используется для установки обновлений, поэтому важно не потерять его.

2.3.4. Заполнить постоянный конфигурационный файл `/opt/config.yml`. Описание конфигурационных параметров представлено в таблице 2 «Описание параметров файла `config.yml`», а также дополнительно приведено в самом файле `config.yml` в виде комментариев.

При заполнении паролей и логинов **важно не использовать спецсимволы**, которые могут некорректно интерпретироваться оболочкой bash и приводить к ошибкам при установке.

Не рекомендуется использовать следующие символы:

~] (>
 \ ` {) /
 \ # } \ ?
 \ $; | @
 \ & ' ["
 \ * < ^ +

В качестве спецсимвола рекомендуется использовать восклицательный знак `!`, с ним проблем не возникает.

Описание параметров в файле config.yml:

Таблица 2. Описание параметров файла config.yml

№	Параметр	Значение	Описание
1	app_host	“<IP>”	ip-адрес или hostname хоста компонентов Приложения
2	db_host	“<IP>”	ip-адрес или hostname хоста компонентов СУБД Postgresql, используется для хранения метаданных
3	dwh_host	“<IP>”	ip-адрес хоста компонентов СУБД ClickHouse, используется для хранения аналитических данных
4	ui_url	“<FQDN>”	точка входа для пользователей, указать FQDN

№	Параметр	Значение	Описание
5	ssh_user	“<username>”	Имя пользователя под которым происходит подключение к серверам для автоматизированной установки компонент SPM
6	ssh_password	“<password>”	Пароль для подключение по ssh к серверам для автоматизированной установки компонент SPM (не требуется заполнять при подключении через ssh ключ)
7	use_local_deploy	“<no>”	Значение по умолчанию “no”. В случае использования сервера приложений в качестве управляющего узла ansible, т.е. когда все компоненты находятся на одном сервере и отсутствует управляющий сервер CI/CD устанавливаем значение параметра “yes”
8	source_images_on_app_server	“<no>”	Значение по умолчанию “no”. В случае расположения файлов образов *.tgz на сервере приложений устанавливаем значение параметра “yes”. По умолчанию предполагается расположения файлов образов на управляющем сервере
9	source_images_storage_folder	“/opt/images”	Абсолютный путь до каталога с файлами образов *.tgz. По умолчанию предполагается каталог /opt/images. На указанную директорию должны быть предоставлены права на создание папок и файлов для пользователя, подключающегося с управляющего сервера
10	app_config_folder	“/opt/spm”	Абсолютный путь до каталога в котором будут хранится конфигурационные файлы сервисов. Каталог должен быть как на сервере приложений, так и на серверах баз данных. По умолчанию предполагается каталог /opt/spm. На указанную директорию должны быть предоставлены права на создание папок и файлов для пользователя, подключающегося с управляющего сервера
11	features	“pm”	Параметр задает сценарий установки. Допустимые значения: pm, tm, ent. Значение по умолчанию “pm”. Указываем значение ent если устанавливаем продукты process mining и task mining по сценарию №1; pm — только process mining сценарий №2
12	spm_admin_username	“<username>”	Имя бизнес пользователя, который будет автоматически создан в системе. Под ним будет возможность войти в систему сразу после установки.
13	spm_admin_password	“<password>”	Пароль бизнес пользователя для входа в систему. Требования: минимум 1 заглавная буква, 1 цифра, 1 спецсимвол (без символов из Таблицы 1), длина ≥8. Пример: Admin123!
14	auth_admin_username	“<username>”	Имя пользователя, который будет администратором Keycloak. Под ним можно создавать пользователей, управлять доступом, сбрасывать пароли.

№	Параметр	Значение	Описание
15	auth_admin_password	“<password>”	Пароль администратора Keycloak. Требования: минимум 1 заглавная буква, 1 цифра, 1 спецсимвол, длина ≥12. Пример: Admin123!spm!
16	infra_admin_username	“<username>”	Имя пользователя под которым будет доступ к мониторингу
17	infra_admin_password	“<password>”	Пароль пользователя для входа в систему мониторинга. Требования: как у spm_admin_password. Пример: Admin123!
18	postgres_db_password	“<password>”	Пароль для подключения к БД PostgreSQL. При использовании внешних СУБД — указать в конфиге и предварительно создать БД (см. п.1)
19	clickhouse_db_password	“<password>”	Пароль для подключения к БД ClickHouse. При использовании внешних СУБД — указать в конфиге и предварительно создать БД
20	analytics_integration_password	“<password>”	Пароль технического пользователя интеграции компонентов.
21	redis_integration_password	“<password>”	Данные для подключения к сервису кеширования Redis.
22	ssl_certificate_password	“<password>”	Пароль генерируемого сертификата для Auth
23	use_external_db	“no”	Параметр регулирует использование внешних баз данных. По умолчанию “no”. Если используются собственные СУБД — установить “yes”
24	ml_host	“<IP>”	IP-адрес хоста компонента pm-ml. По умолчанию совпадает с app_host. При необходимости вынести ML на отдельный сервер — указать его IP
25	pg_port	“5432”	Порт для подключения к PostgreSQL. По умолчанию 5432, может быть изменён
26	ch_port_tcp	“9000”	Порт для подключения к ClickHouse. По умолчанию 9000, может быть изменён
27	title	“”	Сообщение на странице входа. По умолчанию “On-Premise”
28	timezone	"Europe/Moscow"	Временная зона, по умолчанию «Europe/Moscow»
29	calendar	“Ru”	Календарь: “Ru”, “By”, “Kg”. По умолчанию “Ru”
30	use_preview	“no”	Включает/выключает preview дашбордов. Возможные значения: “no”, “yes”. По умолчанию “no”
31	create_pg_db	“yes”	Автоматическое создание БД и схем в PostgreSQL. По умолчанию “yes”. Не менять при использовании СУБД из дистрибутива
32	processmining_db	“”	Заполняется только при use_external_db: “yes”. База данных для схемы processmining
33	db_processmining_username	“”	Имя пользователя PostgreSQL для схемы processmining (только при use_external_db: “yes”)

№	Параметр	Значение	Описание
34	processmining_db_passwd	“”	Пароль пользователя для схемы processmining (только при use_external_db: “yes”)
35	analytics_db	“”	База данных для схемы analytics (при use_external_db: “yes”)
36	db_analytics_username	“”	Имя пользователя для схемы analytics (при use_external_db: “yes”)
37	analytics_db_passwd	“”	Пароль для схемы analytics (при use_external_db: “yes”)
38	etl_db	“”	База данных для схемы etl (при use_external_db: “yes”)
39	db_etl_username	“”	Имя пользователя для схемы etl (при use_external_db: “yes”)
40	etl_db_passwd	“”	Пароль для схемы etl (при use_external_db: “yes”)
41	ml_db	“”	База данных для схемы ml (при use_external_db: “yes”)
42	db_ml_username	“”	Имя пользователя для схемы ml (при use_external_db: “yes”)
43	ml_db_passwd	“”	Пароль для схемы ml (при use_external_db: “yes”)
44	ai_db	“”	База данных для схемы ai (при use_external_db: “yes”)
45	db_ai_username	“”	Имя пользователя для схемы ai (при use_external_db: “yes”)
46	ai_db_passwd	“”	Пароль для схемы ai (при use_external_db: “yes”)
47	auth_db	“”	База данных для схемы auth (при use_external_db: “yes”)
48	db_auth_username	“”	Имя пользователя для схемы auth (при use_external_db: “yes”)
49	auth_db_passwd	“”	Пароль для схемы auth (при use_external_db: “yes”)
50	taskmining_db	“”	База данных для схемы taskmining (при use_external_db: “yes”)
51	db_taskmining_username	“”	Имя пользователя для схемы taskmining (при use_external_db: “yes”)
52	taskmining_db_passwd	“”	Пароль для схемы taskmining (при use_external_db: “yes”)
53	pre_existing_db_name	“”	Имя существующей БД в PostgreSQL, используемой приложением (при use_external_db: “yes”)
54	external_db_username	“”	Пользователь PostgreSQL с правами на инициализацию БД (при use_external_db: “yes”)
55	use_vault	“no”	Использование Vault для хранения секретов. Поддерживается только AppRole. Возможные значения: “yes”, “no”
56	vault_url	«https://example.com:8200»	Актуальная ссылка на Vault

№	Параметр	Значение	Описание
57	vault_secrets_path	"secret_name"	Название секрета в Vault, где хранятся пароли и чувствительные данные
58	vault_role_id	“”	Требуется указать актуальный role_id
59	vault_secret_id	“”	Требуется указать актуальный secret_id
60	vault_storage_path	"storage/path"	Путь до нужного секрета в Vault
61	ansible_hashi_vault_namespace	“”	Vault namespace. Если используется стандартный — оставить пустым

При использовании ассистента необходимо дополнительно задать параметры, приведённые в таблице 3.

Важно

Параметр use_assistant_chat используется для включения и отключения чат-интерфейса ассистента в BI / Process Mining.

Необходимо учитывать различие между параметрами:

- use_assistant — включает или отключает использование функциональности ассистента в составе поставки;
- use_assistant_chat — отдельно управляет отображением и работой чата ассистента в интерфейсе BI / Process Mining.

Если ассистент не используется, параметр use_assistant_chat, а также связанные с ним параметры настройки чата, можно не применять.

Параметры assistant_polling_interval, assistant_users и assistant_limited_tokens применяются только при значении use_assistant_chat="yes".

Таблица 3. Дополнительные параметры файла config.yml для ассистента

№	Параметр	Значение	Описание
62	use_assistant_chat	"no"	Управляет включением и отключением чат-интерфейса ассистента pm-bi.
63	assistant_polling_interval	"10000"	Интервал опроса чата ассистента в pm-bi. Используется только при use_assistant_chat="yes".
64	assistant_users	"*"	Список пользователей или маска пользователей, которым доступен чат ассистента в pm-bi. Используется только при use_assistant_chat="yes".
65	assistant_limited_tokens	"no"	Управляет отображением информации о доступных токенах в чате ассистента. Используется только при use_assistant_chat="yes".
66	global_token_quota	"unlimited"	Глобальная квота токенов для ассистента. Допустимое значение: число токенов либо unlimited.
67	global_token_start_date	""	Дата и время начала действия глобальной квоты токенов ассистента. Рекомендуемый формат: YYYY-MM-DD HH:MM:SS.

№	Параметр	Значение	Описание
68	global_token_end_date	""	Дата и время окончания действия глобальной квоты токенов ассистента. Рекомендуемый формат: YYYY-MM-DD HH:MM:SS.
69	etl_dag_research_enabled	"no"	Включает использование DAG, связанного с AI research, в компоненте ETL.
70	llm_provider	"sber_gigachat"	Тип LLM-провайдера для компонента pm-agent.
71	llm_host_link	"https://localhost/api/"	Адрес endpoint LLM-провайдера для компонента pm-agent.
72	llm_model_name	"llm-name"	Имя модели, используемой компонентом pm-agent.
73	llm_access_token	""	Токен доступа к LLM-провайдеру, если используется токенная аутентификация.
74	llm_scope	"GIGACHAT_API_CORP"	Область действия токена для провайдера GigaChat, если она требуется.
75	llm_verify_ssl	"false"	Проверка TLS-сертификата LLM-провайдера.
76	llm_certificate_path	"/app/certs/cert-agent.crt"	Путь к клиентскому сертификату для подключения к LLM-провайдеру, если контур интеграции требует клиентскую аутентификацию.
77	llm_key_path	"/app/certs/cert-agent.key"	Путь к приватному ключу клиентского сертификата для подключения к LLM-провайдеру.
78	llm_timeout	"300"	Таймаут запросов к LLM-провайдеру в секундах.
79	llm_temperature	"0.005"	Параметр температуры для LLM-модели.
80	llm_repetition_penalty	"1.0"	Параметр штрафа за повторения для LLM-модели.
81	llm_profanity_check	"true"	Включение проверки нежелательной лексики, если она поддерживается используемым провайдером.

2.3.4.1. Параметры квотирования токенов

При использовании ассистента можно задать параметры квотирования токенов.

> **Важно**
>
> Если ограничение токенов не требуется, в параметре global_token_quota указывается значение unlimited.
>
> Если используется ограничение токенов, рекомендуется явно задавать также параметры global_token_start_date и global_token_end_date.

Назначение параметров:

- global_token_quota — общая квота токенов для ассистента;
- global_token_start_date — дата и время начала действия квоты;
- global_token_end_date — дата и время окончания действия квоты;
- assistant_limited_tokens — управление отображением информации о доступных токенах в интерфейсе чата ассистента.

Пример задания параметров с ограничением токенов:

```
global_token_quota=1000000
global_token_start_date="2026-06-01 00:00:00"
global_token_end_date="2026-07-01 00:00:00"
assistant_limited_tokens="yes"
```

Если ассистент не используется, параметры квотирования токенов и отображения информации о токенах можно не задавать.

2.3.5. На сервере приложений создать каталог <app_config_folder>/license и скопировать в него файлы лицензии: license.json и license.json.sig.

2.3.6. При использовании ассистента необходимо убедиться, что в составе поставки присутствуют соответствующие архивы образов и что в config.yml указаны все обязательные пользовательские параметры его настройки.

В случае размещения компонентов ассистента на отдельных узлах необходимо также обеспечить наличие образов, конфигурационных каталогов и требуемых прав доступа на соответствующих целевых серверах.

2.3.7. При использовании TLS или сертификатной аутентификации при подключении компонента pm-agent к LLM-провайдеру необходимо подготовить клиентские сертификаты, приватные ключи и цепочки доверенных сертификатов, после чего скопировать их в целевое расположение, предусмотренное поставкой и конфигурацией. Значения путей должны соответствовать параметрам, указанным в config.yml.

Если компоненты ассистента размещаются на отдельных серверах, сертификаты должны быть размещены на целевых серверах соответствующих компонентов.

2.3.8. Примеры настройки подключения компонента pm-agent к LLM-провайдеру

Ниже приведены примеры пользовательских параметров для настройки подключения компонента pm-agent к различным LLM-провайдерам. В примерах используются плейсхолдеры; перед установкой необходимо заменить их фактическими значениями.

> **Важно**
>
> В файле config.yml следует указывать только актуальные пользовательские параметры, предусмотренные текущей поставкой.
>
> Перед использованием примеров ниже необходимо сверить поддерживаемые параметры с комментариями в config.yml и с шаблонами текущей поставки.

Пример 1. Подключение к Internet GigaChat

```
llm_provider=sber_gigachat
llm_host_link="https://<gigachat-host>/api/v1"
llm_model_name=GigaChat-2-Max

llm_access_token="<token>"
llm_scope="GIGACHAT_API_CORP"
llm_verify_ssl=true

llm_timeout=300
llm_temperature=0.005
llm_repetition_penalty=1.0
llm_profanity_check=true
```

Пример 2. Подключение к LLM через Ollama

```
llm_provider=ollama
llm_host_link="http://<ollama-host>:11434"
llm_model_name=qwen2.5:14b

llm_access_token=""
llm_verify_ssl=false

llm_timeout=300
llm_temperature=0.005
llm_repetition_penalty=1.0
```

Пример 3. Подключение к OpenAI-compatible API

```
llm_provider=openai_compatible
llm_host_link="http://<api-host>:8002/v1"
llm_model_name=qwen3-14b

llm_access_token="<token>"
llm_verify_ssl=false

llm_timeout=300
llm_temperature=0.005
llm_repetition_penalty=1.0
```

Примечание

Используемый способ аутентификации определяется целевым контуром интеграции. Если используется токенная аутентификация, клиентские сертификаты обычно не требуются. Если используется сертификатная аутентификация, токен может не использоваться.

2.3.9. При необходимости шифрования паролей использовать ansible-vault. Пример шифрования пароля с использованием ansible-vault:

ansible-vault encrypt_string '<password>' --name '<name>'

где <password> — пароль, который требуется зашифровать, <name> — имя параметра.

2.3.10. Файл deploy-<version>/inventories/local/group_vars/all/env.yml, где <version> — версия поставки, например, 2.9.6, содержит служебную информацию о портах, именах контейнеров и параметрах окружения, формируемых при развертывании.

В общем случае дополнительная настройка файла env.yml не требуется.

При использовании ассистента рекомендуется убедиться, что в config.yml заданы необходимые пользовательские параметры его настройки, в том числе:

- use_assistant;
- use_assistant_chat;
- assistant_polling_interval;
- assistant_users;
- assistant_limited_tokens;
- global_token_quota;
- global_token_start_date;
- global_token_end_date;
- etl_dag_research_enabled;
- llm_provider;

- `llm_host_link`;
- `llm_model_name`;
- параметры аутентификации и TLS для подключения к LLM-провайдеру.

2.3.11. В первую очередь запустить развертывание сервисов СУБД PostgreSQL и ClickHouse, выполнив playbook `setup.yml` командой (запуск команды предполагается из каталога `deploy-<version>`):

```
ansible-playbook -i inventories/local setup.yml -e "@/opt/config.yml"
```

2.3.12. После успешного выполнения развертывания сервисов СУБД запустить установку компонентов приложения, выполнив playbook `deploy.yml` командой (запуск команды предполагается из каталога `deploy-<version>`):

```
ansible-playbook -i inventories/local deploy.yml -e "@/opt/config.yml"
```

2.3.13. При включенном параметре `use_assistant_chat="yes"` в состав развертываемых компонентов приложения должны входить сервисы ассистента, предусмотренные текущим сценарием поставки.

2.3.14. Для компонентов, использующих миграции схемы БД, после развертывания необходимо применить миграции, если их выполнение не автоматизировано поставкой.

2.3.15. В случае использования шифрования паролей при помощи `ansible-vault` необходимо добавить к команде запуска развертывания параметр:

```
--ask-vault-pass
```

3. Обновление

3.1. Скачать дистрибутив по ссылке из поставки, скопировать его на управляющий сервер и распаковать.

3.2. Найти и распаковать архив deploy-<version>.tar на управляющем сервере (CI/CD) командой: `tar -xvf deploy-<version>.tar`, где <version> - это версия поставки, например, 2.9.6.

3.3. Запустить установку компонентов приложения, запустив playbook deploy.yml командой (запуск команды предполагается из каталога deploy-<version>): `ansible-playbook -i inventories/local deploy.yml -e "@/opt/config.yml"`

3.4. В случае обновления до версии HotFix, когда в состав поставки входит только 1 компонент, запустить playbook развертывания обновляемого компонента поставки (запуск команды предполагается из каталога deploy-<version>): `ansible-playbook -i inventories/local <имя_компонента>.yml -e "@/opt/config.yml"`, где <имя_компонента> это имя контейнера, например, pm-ui, pm-bi и пр.

4. Удаление

4.1. Удаление компонентов продукта с сервера приложений выполняется командой:

```
docker stop $(docker ps -a -q) || true && \
docker rm $(docker ps -a -q) || true && \
docker volume rm $(docker volume ls -q) || true && \
docker rmi $(docker images -aq) || true
```

Внимание!

Команда выше остановит и удалит все запущенные контейнеры, удалит все docker volumes и все образы на сервере приложений! Если на сервере приложений существуют контейнеры помимо компонентов продукта, то данную команду запускать не следует, в противном случае они также будут удалены. В этом случае следует:

4.1.1. заменить инструкцию $(docker ps -a -q) на список контейнеров, например, pm-ui pm-bi pm-etl и т.д. (список контейнеров можно получить, выполнив команду: `docker ps -a`);

4.1.2. инструкцию $(docker volume ls -q) заменить на список конкретных волюмов, например, etldata, authdata и т.д. (список волюмов можно получить, выполнив команду: `docker volume ls`);

4.1.3. инструкцию $(docker images -aq) на список конкретных образов, например, pm-ui:2.9.6 pm-bi:2.9.6 и т.д. (список образов можно получить, выполнив команду: `docker images`).

4.2. Удаление компонентов продукта с серверов баз данных производится аналогично удалению компонентов с сервера приложений, описанному в предыдущем пункте 4.1.

Внимание!

Очистка волюмов баз данных clickdata и pgdata приведет к безвозвратной потере данных!

5. Проверка работоспособности

5.1. Проверить, что все требуемые компоненты развернуты и запущены в Docker-контейнерах, выполнив команду:

```
docker ps -a
```

В выводе команды все компоненты, входящие в выбранный сценарий установки, должны находиться в статусе Up, за исключением компонентов ассистента в случае, если чат ассистента отключен параметром `use_assistant_chat="no"`.

5.2. В случае если контейнер с компонентом не запустился, необходимо проверить логи работы соответствующего сервиса. Для получения лога конкретного компонента выполнить команду:

```
docker logs --tail <количество_строк> <имя_контейнера>
```

Примеры:

- `docker logs --tail 1000 pm-ui`
- `docker logs --tail 1000 pm-agent`
- `docker logs --tail 1000 pm-ai`

5.3. При возникновении проблем с доступностью сервиса проверить доступность соответствующего порта командой `telnet` или аналогичной утилитой:

```
telnet <server_ip> <port>
```

где <server_ip> — IP-адрес сервера, на котором работает контейнер, <port> — порт сервиса.

Таблица соответствия портов и сервисов

№	Сервис (контейнер)	Порты
1	IAM (pm-ui)	80, 443
2	BI (pm-bi)	8030
3	ML (pm-ml)	8010
4	ETL (pm-etl)	8020
5	Auth (pm-auth)	8090, 9090
6	TM (pm-tm)	8000
7	PG (postgres-server)	5432
8	CH (clickhouse-server)	9000, 8123
9	DOCS (pm-docs)	9030
10	Agent (pm-agent)	9070

№	Сервис (контейнер)	Порты
11	AI Assistant API (pm-ai)	9020
12	AI Assistant MCP / Tools API (pm-ai)	9040

5.4. При проверке компонентов ассистента необходимо учитывать значение параметра use_assistant_chat в файле config.yml:

- если use_assistant_chat="yes", компоненты ассистента должны быть развернуты и находиться в статусе Up, а их API должны быть доступны по соответствующим портам;
- если use_assistant_chat="no", чат ассистента в bi должен быть отключен; компоненты ассистента могут быть остановлены либо не участвовать в текущем запуске, и такое состояние не считается ошибкой;
- параметры assistant_polling_interval, assistant_users и assistant_limited_tokens подлежат проверке только в случае, если use_assistant_chat="yes".

5.5. При использовании компонента pm-agent дополнительно необходимо проверить корректность конфигурации подключения к LLM-провайдеру.

Проверке подлежат:

- наличие обязательных параметров, определенных выбранным вариантом интеграции;
- корректность значения llm_provider;
- доступность значения параметра llm_host_link с хоста размещения компонента;
- корректность значения llm_model_name;
- наличие токена доступа либо клиентских сертификатов, если они требуются выбранным контуром интеграции;
- отсутствие в логах компонента pm-agent ошибок инициализации LLM-клиента, ошибок TLS и ошибок аутентификации.

5.6. При использовании квотирования токенов и параметров отображения ограничений дополнительно необходимо проверить согласованность пользовательской конфигурации ассистента.

Проверке подлежат:

- параметры global_token_quota, global_token_start_date и global_token_end_date;
- параметр assistant_limited_tokens.

Если используется сценарий с ограничениями по токенам, необходимо убедиться, что:

- значения периода квотирования заданы корректно;
- информация о доступных токенах отображается или скрывается в соответствии со значением assistant_limited_tokens;
- в логах компонентов отсутствуют ошибки, связанные с обработкой квот токенов и параметров конфигурации ассистента.

5.7. Для сценария №1 и сценария №2 установки продуктов выполнить проверку входа в UI-интерфейс по ссылке:

https://<ui_url>

Авторизация выполняется под пользователем spm_admin_username с паролем spm_admin_password.

Критерием успешности проверки является успешная авторизация пользователя в системе и открытие пользовательского интерфейса.

5.8. Для сценария №3 установки выполнить переход по ссылке:

https://<ui_url>/events

Критерием успешности проверки является открытие формы авторизации.

5.9. При включенном параметре use_assistant_chat="yes" дополнительно необходимо проверить корректность отображения чата ассистента в интерфейсе Process Mining.

Проверке подлежат:

- наличие элемента чата ассистента в пользовательском интерфейсе;
- доступность функций отправки запроса в чат;
- корректность отображения информации о доступных токенах в зависимости от значения `assistant_limited_tokens`;
- отсутствие ошибок в браузерной консоли и сетевых запросах, связанных с вызовами компонентов ассистента.

5.10. При значении `use_assistant_chat="no"` необходимо убедиться, что:

- чат ассистента в интерфейсе Process Mining не отображается;
- отсутствие запущенных компонентов ассистента не интерпретируется как ошибка;
- в логах развертывания отсутствуют критические ошибки, связанные с отключенным чат-интерфейсом ассистента.

5.11. Общим критерием успешной проверки работоспособности является:

- все требуемые контейнеры находятся в статусе Up;
- пользовательский интерфейс доступен;
- авторизация выполняется успешно;
- компоненты ассистента доступны по своим API при включенном `use_assistant_chat`;
- параметры конфигурации ассистента, квотирования и отображения токенов согласованы;
- в логах отсутствуют критические ошибки запуска, подключения к зависимым сервисам, инициализации LLM-провайдера и чтения конфигурации.

6. Установка TLS сертификата, выпущенного ЦС организации

6.1. Создание запроса на сертификат и выпуск сертификата

В случае необходимости выпуска собственного серверного сертификата, подписанного Центром сертификации организации, требуется выполнить установку TLS сертификата.

Установка выполняется после успешной установки компонентов Sber Process Mining.

6.1.1. Создание ключа и запроса на сертификат одним из двух способов:

1. Прописав параметры напрямую в команде

```
openssl req -newkey rsa:2048 -nodes -keyout servercert.key -out servercert.csr -subj "/CN=FQDN/OU=Organization Unit/O=Organization/C=RU"
```

Заполнить Параметры:

`CN=FQDN`

`OU=Organization Unit`

`O=Organization`

`C=RU`

В результате выполнения команды будут созданы файлы ключа и запроса на сертификат:

`servercert.key`

`servercert.csr`

2. С использованием файла конфигурации (предварительно создать файл конфигурации):

```
openssl req -out servercert1.csr -newkey rsa:2048 -nodes -keyout servercert.key -config certificate.conf
```

Пример содержимого файла конфигурации certificate.conf:

```
[req] prompt = no default_bits = 2048 distinguished_name = dn req_extensions = req_ex
t [dn] countryName = RU organizationName = <Имя организации> commonName = <CN сертифи
ката> organizationalUnitName = <Наименование организации> [req_ext] subjectAltName =@
alt_names [alt_names] DNS.1 = <FQDN1> DNS.2 = <FQDN2>
```

В результате выполнения команды будут созданы файлы ключа и запроса на сертификат:

`servercert.key`

`servercert.csr`

- Ключ –nodes не шифрует содержимое файла закрытого ключа.

6.2. Подписание, выпуск серверного сертификата

Выпуск сертификата выполняется Удостоверяющим Центром (далее - УЦ) с использованием ключа и сертификата УЦ.

Команда для выпуска сертификата с использованием ключа и сертификата УЦ:

```
openssl x509 -req -CA issuer.cer -CAkey issuer.key -extensions server_cert -in <hostn
ame>.csr -out <hostname>.cer -days 2048 -CAcreateserial
```

`issuer.cer` - сертификат-подпись УЦ, `issuer.key` - ключ УЦ, `<hostname>.csr` запрос на сертификат сервера (п.6.1).

6.3. Проверить выпуск сертификата

```
openssl x509 -noout -text -in <hostname>.cer
```

6.4. Конвертация приватного ключа и сертификата в требуемый формат base64

6.4.1. В случае если при выпуске исходный файл сертификата был сформирован в формате pfx (хранилище p12), то требуется преобразовать его в отдельный файл сертификата и отдельный файл закрытого ключа. И далее сконвертировать полученные ключ и сертификат в формат base64. Последовательность действий:

1. Посмотреть содержимое хранилища сертификата:

```
openssl pkcs12 -in <имя_файла_сертификата>.pfx -nodes -passin pass:<пароль>
```

2. Извлечение ключа:

```
openssl pkcs12 -in <имя_файла_хранилища>.pfx -nocerts -nodes -out <имя_файла_ключа1>.
key
```

Вводим пароль от хранилища и получаем файл ключа.

3. Конвертация ключа из PKCS#8 в base64:

```
openssl pkey -inform PEM -in <имя_файла_ключа1>.key -outform PEM -out <имя_файла_ключ
а2>.key
```

4. Извлечение сертификата:

```
openssl pkcs12 -in <имя_файла_хранилища>.pfx -clcerts -nokeys -out <имя_файла_сертифи
ката1>.pem
```

Вводим пароль от хранилища и получаем файл сертификата.

5. Конвертация сертификата из PKCS#8 в base64:

```
openssl x509 -inform PEM -in <имя_файла_сертификата1>.pem -outform PEM -out <имя_файл
а_сертификата2>.pem
```

6.4.2. В случае если ключ и сертификат выпущены в формате DER - конвертировать в base64:

1. Конвертация ключа:

```
openssl pkey -inform der -outform pem -in <имя_файла_ключа_в_формате_DER>.key -out <и
мя_файла_ключа_в_формате_BASE64>.key
```

2. Конвертация сертификата:

`openssl x509 -inform der -outform pem -in <имя_файла_сертификата_в_формате_DER>.cer -out <имя_файла_сертификата_в_формате_BASE64>.pem`

6.5. Переименование файлов приватного ключа и сертификата по шаблону именования

Файл серверного сертификата должен называться: `<URL-входа-в-систему>.crt`

Файл ключа должен называться: `<URL-входа-в-систему>.key`

Название должно быть таким же как значение переменной ui_url в файле конфигурации **config.yml**

Пример: `spm.company.crt и spm.company.key`

6.6. Копирование файлов приватного ключа и сертификата в целевое расположение

Скопировать файлы ключа и сертификата в папку: **/opt/spm/certs**

6.7. Копирование файлов корневых и промежуточных сертификатов в целевое расположение

1. Способ: скопировать корневой сертификат root ca в формате base64 в файл trusted_root_ca.crt в папку: `/opt/spm/certs`
2. Способ: скопировать промежуточный сертификат sub ca в формате base64 в файл trusted_sub_ca.crt в папку: `/opt/spm/certs`

6.8. Запуск установки/обновления сертификата

Сертификаты обновляются вне зависимости от установки релиза. Если работы по обновлению сертификатов выполняются вместе с установкой или обновлением релиза, то запуск обновления сертификатов необходимо провести после установки/обновления релиза и проверки работоспособности запуском playbook (запуск команды предполагается из каталога deploy-<version>):

`ansible-playbook -i inventories/local tasks.yml -e "@/opt/config.yml" -e "cert_renew=yes"`, где ../config.yml путь до файла конфигурации с которым выполняется деплой или обновление системы.

7. Настройка интеграции с Vault

Интеграция с HashiCorp Vault является опциональной. Поддерживается вариант интеграции только через AppRole.

7.1. Параметры, которые могут быть получены из Vault

Параметры, указанные ниже, могут быть получены из Vault и должны быть предварительно туда занесены:

analytics_integration_password
auth_admin_password
clickhouse_db_password
infra_admin_password
postgres_db_password
redis_integration_password

analytics_integration_password
spm_admin_password
ssl_certificate_password
pm_ai_pg_password

При использовании компонента `pm-ai` в Vault рекомендуется также хранить чувствительные параметры, относящиеся к его интеграциям с LLM-провайдерами и внешними AI-сервисами, включая токены доступа, пароли и иные секреты, если такие параметры используются в конкретной поставке.

В случае использования Vault перечисленные выше параметры в файле `config.yml` не заполняются.

7.2. Параметры Vault в файле config.yml

Для использования интеграции с Vault в файле `config.yml` необходимо заполнить следующие параметры:

use_vault
vault_url
vault_secrets_path
vault_role_id
vault_secret_id
vault_storage_path
ansible_hashi_vault_namespace

Параметры `vault_role_id` и `vault_secret_id` используются для аутентификации в Vault по схеме AppRole.

7.3. Подготовка сертификатов для подключения к Vault

На сервере приложения необходимо разместить актуальную цепочку сертификатов в каталог `<app_config_folder>/certs` (по умолчанию `/opt/spm/certs`) для коммуникации с сервером Vault.

Важно, чтобы расширение файлов сертификатов было `.crt`.

Без корректной цепочки доверенных сертификатов подключение к Vault по TLS может завершиться ошибкой проверки SSL/TLS-соединения.

Если компонент `pm-ai` размещается на отдельном сервере, сертификаты для подключения к Vault должны быть также размещены на целевом сервере данного компонента.

7.4. Запуск установки приложения с использованием Vault

После заполнения параметров Vault в `config.yml` и размещения необходимых сертификатов необходимо выполнить установку приложения той же командой, что используется для установки и обновления приложения согласно пункту 2.3.12.

При использовании компонентов `pm-ai` и `pm-agent` необходимо убедиться, что целевые сервисы имеют доступ к Vault в соответствии с сетевой схемой и архитектурой развертывания.

8. Откат

Откат до предыдущей версии не предусмотрен ввиду отсутствия обратной совместимости после обновления СУБД. Для исправления ошибок выпускается и устанавливается релиз HotFix.

9. Проблемы и пути их устранения (FAQ)

В случае возникновения ошибки развертывания через ansible необходимо проанализировать текст ошибки на предмет причины возникновения.

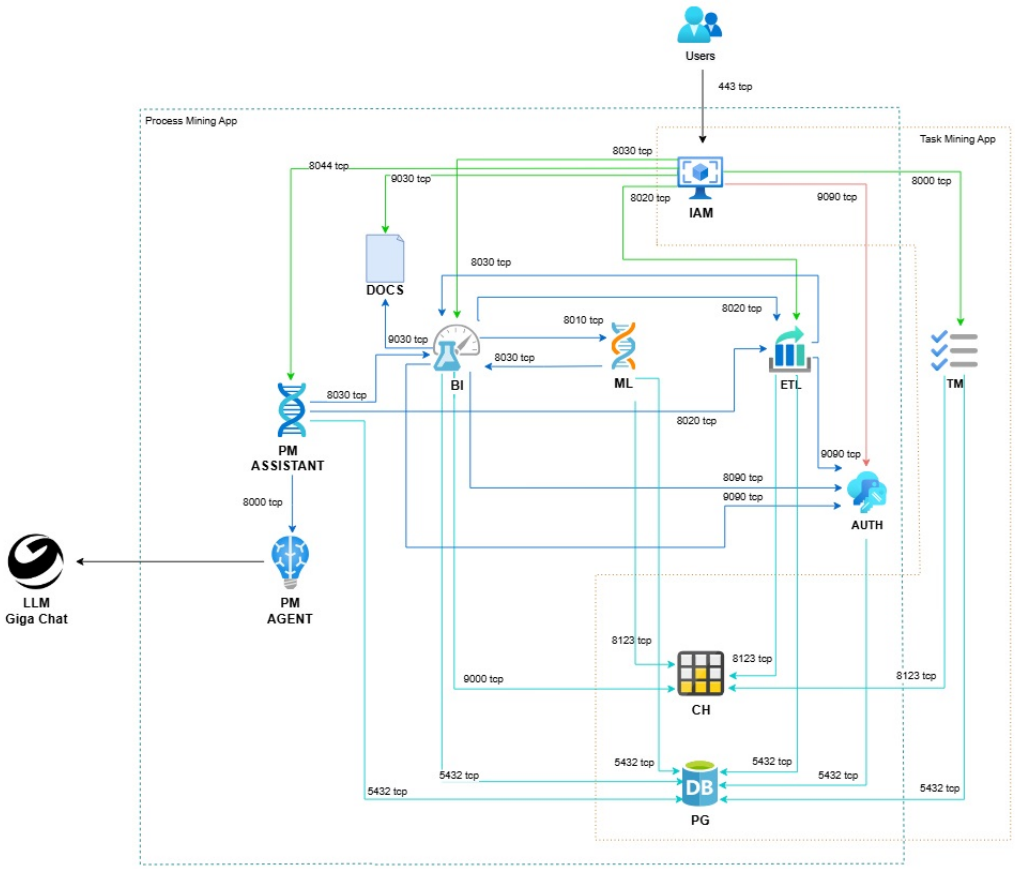
В случае инфраструктурных проблем выполнить поиск данной ошибки по общедоступным источникам, официальной документации на платформу ОС. Если решение найти не удалось – выписать название ansible TASK, текст ошибки и передать сопровождению.

10. Проверка установки. Чек-лист

Критерий проверки	Ожидаемый результат
Развертывание выполнено без ошибок	Запущенные playbook отработали без ошибок
Все требуемые сервисы развернуты и запущены	Результат команды `docker ps -a` показывает, что все сервисы, входящие в выбранный сценарий установки, находятся в статусе `Up`, за исключением компонентов ассистента при `use_assistant_chat="no"`
В случае установки по сценарию №1 и сценарию №2: вход на UI выполнен успешно	Вход в пользовательский интерфейс по ссылке `ui_url` пользователем `spm_admin_username` выполнен успешно
В случае установки по сценарию №3: переход по ссылке `<ui_url>/events`	Переход по ссылке `<ui_url>/events` выполнен успешно, отображается форма авторизации
В случае использования ассистента при `use_assistant_chat="yes"`: сервис `pm-agent` запущен	Контейнер `pm-agent` находится в статусе `Up`, API компонента доступно по порту `9070`
В случае использования ассистента при `use_assistant_chat="yes"`: подключение к LLM настроено корректно	В конфигурации присутствуют обязательные параметры выбранного LLM-провайдера, значения `llm_provider`, `llm_host_link` и `llm_model_name` заполнены корректно, в логах отсутствуют ошибки TLS, аутентификации и инициализации LLM-клиента
В случае использования ассистента при `use_assistant_chat="yes"`: сервис `pm-ai` запущен	Контейнер `pm-ai` находится в статусе `Up`, основной API доступен по порту `9020`, MCP / Tools API доступен по порту `9040`
В случае использования ассистента при `use_assistant_chat="yes"`: зависимости доступны	Компонент `pm-ai` имеет успешные подключения к PostgreSQL, bi, Airflow и сертификатам, указанным в конфигурации
В случае использования ассистента при `use_assistant_chat="yes"`: миграции БД применены	Миграции схемы БД выполнены успешно, в логах отсутствуют ошибки `alembic` и ошибки инициализации БД
В случае использования квотирования токенов: параметры заполнены корректно	Параметры `global_token_quota`, `global_token_start_date` и `global_token_end_date` заданы корректно в соответствии с выбранным сценарием эксплуатации
В случае использования чата ассистента: отображение информации о токенах настроено корректно	Информация о доступных токенах в интерфейсе чата отображается или скрывается в соответствии со значением `assistant_limited_tokens`

Критерий проверки	Ожидаемый результат
В случае `use_assistant_chat="yes"`: чат ассистента доступен в интерфейсе	В интерфейсе Process Mining отображается чат ассистента, запросы в чат отправляются успешно, ошибки frontend/backend интеграции отсутствуют
В случае `use_assistant_chat="no"`: чат ассистента отключен корректно	Чат ассистента не отображается, отсутствие запущенных контейнеров `pm-agent` и `pm-ai` не считается ошибкой
Логи сервисов не содержат критических ошибок	В логах контейнеров отсутствуют критические ошибки запуска, подключения к зависимым сервисам, инициализации LLM-провайдера, чтения конфигурации и обработки параметров квотирования

Приложение 1:



Легенда:

- цветом показаны не аутентифицированные запросы.
- цветом отмечены аутентифицированные запросы.
- цветом отражены взаимодействия между компонентами внутри системы.
- выделены обращения к базам данных.
- выделены внешнии взаимодействия.

Описание схемы сетевого взаимодействия:

1. Пользователь из браузера отправляет запрос. Запрос поступает в IAM, где проверяется наличие корректного JWT-токена.

В случае отсутствия JWT-токена пользователь перенаправляется на страницу аутентификации компонента Auth для ввода логина и пароля. Если JWT-токен присутствует, пользователь авторизован и имеет возможность выполнять запросы к системе.

В противном случае пользователь аутентифицируется в сервисе авторизации Auth и получает JWT-токен.

2. IAM отправляет и получает данные от BI и Auth.
3. BI обращается к БД `spm_analytics` в СУБД PostgreSQL.
4. Auth обращается к БД `spm_auth` в СУБД PostgreSQL.
5. BI и ML обращаются к базам данных в СУБД ClickHouse для построения аналитики на основе загруженных пользователем данных.
6. Данные от агентов логирования на рабочих станциях пользователей поступают в сервис логирования и в БД `TaskMining`.
7. При использовании чата ассистента компонент BI / Process Mining взаимодействует с компонентами `pm-ai` и `pm-agent` в соответствии с сетевой схемой и параметром `use_assistant_chat`.
8. При использовании внешнего или внутреннего LLM-провайдера компонент `pm-agent` выполняет вызовы к соответствующему AI-контуру в соответствии с конфигурацией подключения и требованиями безопасности.

Сокращения и определения

Сокращение	Описание
IAM	Компонент FrontEnd (pm-ui)
BI	Компонент для исследования и визуализации данных (pm-bi)
ML	Компонент Machine Learning для анализа пользовательских путей (pm-ml)
ETL	Компонент извлечения, преобразования и загрузки данных
Auth	Компонент авторизации пользователей (pm-auth)
DOCS	Компонент документации (pm-docs)
TM	Компонент логирования task mining
PG	СУБД PostgreSQL
CH	СУБД ClickHouse
СП	Сервер приложений
ОС	Операционная система
LLM	Большая языковая модель, используемая компонентами `pm-agent` и `pm-ai` через внешнего или внутреннего провайдера
TLS	Механизм криптографической защиты сетевого взаимодействия, используемый при подключении к внутренним и внешним сервисам
Ассистент	Ассистент для исследований данных, компонент (pm-ai)
Агент	Компонент AI / LLM-обработки, обеспечивающий выполнение запросов ассистента (pm-agent)

Продукт Process Mining включает в себя компоненты IAM, BI, ML, ETL, Auth, DOCS, PG и CH.

Продукт Task Mining включает в себя компоненты IAM, TM, PG и CH.

Возможны различные сценарии установки продуктов:

Сценарий №1: Установка продуктов Process Mining и Task Mining.

Устанавливаются компоненты: IAM, BI, ML, ETL, TM, Auth, DOCS, PG и CH.

Сценарий №2: Установка продукта Process Mining.

Устанавливаются компоненты: IAM, BI, ML, ETL, Auth, DOCS, PG и CH.

Сценарий №3: Установка продукта Task Mining.

Устанавливаются компоненты: IAM, TM, PG и CH.

Сценарий №4: Установка продукта Process Mining и агента.

Устанавливаются компоненты: IAM, BI, ML, ETL, Auth, DOCS, PG, CH, PM-AGENT и PM-AI.

Для развертывания компонентов продукта Sber Process Mining потребуется минимально четыре сервера, перечисленных в таблице раздела 1.2.

Для развертывания используется управляющий сервер (CI/CD), в роли которого выступает отдельный сервер либо сервер, совмещенный с сервером приложений (СП). При необходимости масштабирования предусмотрена возможность выделить под каждый компонент системы отдельный СП.

На сервере СУБД PostgreSQL будет расположен 1 экземпляр СУБД PostgreSQL (PG), на сервере СУБД ClickHouse будет расположен 1 экземпляр СУБД ClickHouse (CH).

PG — компонент СУБД PostgreSQL, содержит следующие базы данных и схемы (создаются автоматически):

- БД `sberprocessmining`
- Схема `processmining` (сервис `pm-core`)
- Схема `analytics` (сервис `pm-bi`)
- Схема `auth` (сервис `pm-auth`)
- Схема `etl` (сервис `pm-etl`)
- Схема `taskmining` (сервис `pm-tm`)

CH — компонент СУБД ClickHouse, содержит следующие БД с пользовательскими данными для работы сервисов BI, ML, ETL, TM (создаются автоматически):

- `_00UserData_spm`
- `_01RawData_spm`
- `_02PreparedData_spm`
- `_03ResearchData_spm`
- `_01DictionaryData`
- `TaskMining`
- `ch_storage_spm`