Сведения об установке экземпляра программы

Система управления платежами Forward PMS

Правообладатель:
ООО «Форвард-Телеком»

Forward PMS (Payment Management System) – Система управления платежами, представляет собой распределенную масштабируемую систему для организации и автоматизации процессов приема и обработки платежей, поддерживающую интеграцию с внешними платёжными системами для приёма транзакций, управления жизненным циклом платежей (создание, изменение, удаление, получение статуса, поиск), а также загрузки и обработки реестров платежей.

Система обеспечивает поддержку работы с рекуррентными платежами и мёрчантами, предоставляет гибкие интерфейсы для настройки бизнес-логики и интеграции с внешними учётными и биллинговыми системами, а также реализует механизмы мониторинга, журналирования и анализа операций по платёжным событиям различной структуры.

Использование данного решения позволяет операторам существенно снизить затраты на интеграцию с различными платёжными провайдерами, автоматизацию обработки транзакций, снижение числа ошибок при ручной обработке данных, а также ускорить запуск новых платёжных сценариев за счёт гибкой настройки бизнес-правил и процессов.

Контактная информация: support@fw-t.ru

1 Дистрибутив

1.1 Распространение дистрибутива

Дистрибутив комплекта программ распространяется на физических носителях (USB-flash накопителе) или по ссылке на облачное хранилище.

Дистрибутив формируется для каждого заказчика с учетом технологической экосистемы и необходимых модулей расширения функционала.

Подробная документация поставляется после адаптации системы в процессе внедрения с учетом особенностей кастомизации.

1.2 Содержание дистрибутива

Перечень устанавливаемых компонентов осуществляется в соответствии со спецификацией поставки.

Дистрибутивы поставляются как docker-образ и обеспечивают в дальнейшем корректную установку PMS.

2 Порядок установки программного обеспечения

Модуль PMS устанавливается из докер-образа.

Специалистами Форвард-Телеком проводится групповое обучение специалистов Заказчика по развертыванию и настройкам системы в целях обеспечения её дальнейшей корректной эксплуатации.

2.1 Подготовка к установке

Перед установкой PMS необходимо получить актуальный релиз докер-образа и запустить сервис PostgreSQL – установить и настроить среду:

PostgreSQL

- Запуск сервиса PostgreSQL
- Создание схемы с названием, который указан в настройках Finances

Subscriber

- Установка/Изменение

  а)

  Установка через опцию option_microservices_integration:

  Создание пользователя "finances_ms"

  Создание типа проводки с внешним идентификатором "finances_ms"

  Создание источника платежа с внешним идентификатором "finances_ms"

  б)

  Изменение пакета на уровне настроек приложения (app.finances.pack)
- Запуск сервиса Subscriber

Finances

- Создание директорий на сервера для реестров (in, done, error)
- Запуск Finances

2.2 Установка

Порядок инсталляции PMS представляет собой последовательное выполнение действий по:

1. Получению Docker-образа PMS
2. Конфигурированию параметров приложения
3. Настройке профилей PMS

2.2.1 Получение Docker-образа

Получить docker-образ и выполнить команду вида:

```
docker pull registry.gitlab.ru/group/project/release:latest
```

Для запуска/остановки сервиса из docker compose файла используется команды:

```
Названия сервисов: postgres, finances, subscriber-oracle
// Запуск
docker compose up -d --force-recreate НАЗВАНИЕ СЕРВИСА
// Остановка
docker compose stop НАЗВАНИЕ СЕРВИСА
// Просмотр логов
docker compose logs НАЗВАНИЕ СЕРВИСА -f --tail 1000
```

2.2.2 Конфигурирование параметров приложения

Выполнить конфигурацию параметров приложения.

Опция	Значение по умолчанию	Комментарий
spring.application.name	Finances	Название приложения
server.port	8080	Порт приложения
spring.datasource.hikari.connection-timeout	5000	Таймаут подключения к БД
spring.datasource.hikari.maximum-pool-size	5	Размер пула подключений
spring.jpa.properties.hibernate.jdbc.batch_size	30	Размер пачки выполнения batch операций в БД
grpc.client.subscriber.address	static://localhost:8051	Путь к gRPC сервису сабскрайбинга
spring.profiles.active	paysystems,export-kafka,export-subscriber-ora,registers-load,recurrents	Профили приложения активируют разные аспекты работы модуля: • paysystems - интеграция с платёжными системами (платёжные сервлеты) • export-kafka - выгрузка

Опция	Значение по умолчанию	Комментарий
		информации о платежах в Kafka • export-subscriber-ora - выгрузка платежей напрямую в сабскрайбер на Оракле • registers-load - модуль загрузки реестров платежей (сканирование директорий) • recurrents - подсистема приёма рекуррентных платежей
app.default.lockAtMostFor	PT10M	Максимальное время, на которое устанавливается распределённая блокировка shedlock
app.default.currency	RUB	Валюта по умолчанию
app.scheduler.initialDelay	PT5S	Время задержки перед первым выполнением задачи
app.paysystems.header-real-ip	X-Real-IP	Http header, указывающий на реальный IP
app.paysystems.header-payment-system-ident	X-PaymentSystem	Заголовок с названием платёжной системы, в сторону которой идёт обращение
app.paysystems.audit-requests	true	Признак записи в аудит платёжных систем - таблица payment_system_audit
app.paysystems.skip-save-headers	accept,accept-encoding,connection,cookie,postman-token	Список заголовков http запроса, которые не будут сохранены в аудит
app.paysystems.json		Текст конфигурационного файла активных платёжных систем в

Опция	Значение по умолчанию	Комментарий
		формате json
app.paysystems.json-path	classpath:paysystems.json	Путь к конфигурационному файлу активных платёжных систем в формате json
app.scheduler.payments.topic	payments-flow	Название топика Kafka для отправки событий о приёме/удалении платежа
app.scheduler.payments.exportFixedRate	PT5S	Частота запуска шедулера - экспорт платежей в Kafka
app.scheduler.payments.lockAtLeastFor	PT5S	Минимальное время блокировки шедулера - экспорт платежей в Kafka
app.scheduler.payments.exportTo	Kafka,SubscriberOra	Системы для экспорта платежей • Kafka - выгрузка событий в Kafka. Выгрузку осуществляет процесс finances с профилем export-kafka • SubscriberOra - выгрузка событий в Subscriber-Ora. Выгрузку осуществляет процесс finances с профилем export-subscriber-ora
app.registers-load.threads	5	Кол-во потоков загрузки реестров платежей
app.registers-load.loadFixedRate	PT5S	Частота запуска шедулера - загрузка реестров платежей
app.registers-load.lockAtLeastFor	PT5S	Минимальное время блокировки шедулера - загрузка реестров платежей
app.registers-load.maximumFileLock	PT5M	Максимальное время

Опция	Значение по умолчанию	Комментарий
		обработки файла реестра платежей. Если обработка файла не закончена в указанное время - файл будет взят на обработку другим потоком.
app.registers-load.fillContractRegCreated	PT72H	Заполнять контракт для реестров, созданных в указанный период от текущего момента
app.registers-load.fillContractCheckDelay	PT1H	Задержка повторной проверки контракта
app.registers-load.json		Текст конфигурационного файла активных загрузчиков реестров платежей в формате json
app.registers-load.json-path	classpath:registers.json	Путь к конфигурационному файлу активных загрузчиков реестров платежей в формате json
app.registers-lead.leadFixedRate	PT5S	Частота запуска шедулера - поиск контрактов для платежа реестра
app.registers-lead.lockAtLeastFor	PT5S	Минимальное время блокировки шедулера - поиск контрактов для платежей реестра
app.recurrents.json		Текст конфигурационного файла активных обработчиков рекуррентных платежей в формате json
app.recurrents.json-path	classpath:recurrents.json	Путь к конфигурационному файлу активных обработчиков

Опция	Значение по умолчанию	Комментарий
		рекуррентных платежей в формате json
app.recurrent.max-retry-count	3	Кол-во попыток выполнения рекуррентного платежа при возникновении ошибок
app.cloudpayments.base-url	https://api.cloudpayments.ru	Адрес сервиса cloudpayments
app.paymaster.base-url	https://paymaster.ru/api/v2	Адрес сервиса paymaster
app.yookassa.base-url	https://api.yookassa.ru	Адрес сервиса yookassa
app.recurrent.scheduler.initialDelay	PT5S	Время задержки перед первым выполнением задачи
app.recurrent.recurrentFixedRate	PT5S	Выражение для фиксированного выполнения задачи (обработка очереди рекуррентных платежей)
app.recurrent.lockAtLeastFor	PT5S	Время блокировки обработчика задач (обработка очереди рекуррентных платежей)
app.recurrent.recurrentProcessedRequestsFixedRate	PT5S	Выражение для фиксированного выполнения задачи (обработка рекуррентных платежей с зависшим статусом)
app.recurrent.token.scheduler.initialDelay	PT5S	Время задержки перед первым выполнением задачи (обработка очереди токенов)
app.recurrent.token.recurrentFixedRate	PT5S	Выражение для фиксированного выполнения задачи (обработка очереди токенов)

Опция	Значение по умолчанию	Комментарий
app.recurrent.token.lockAtLeastFor	PT5S	Время блокировки обработчика задач (обработка очереди токенов)
app.scheduler.paymentTokens.topic	paymentTokens-flow	Название топика, по которому будет создано событие в кафку о токене
app.scheduler.paymentTokens.exportFixedRate	PT5S	Выражение для фиксированного выполнения задачи (экспорт токенов в кафку)
app.scheduler.paymentTokens.lockAtLeastFor	PT5S	Время блокировки обработчика задач (экспорт токенов в кафку)

2.2.3 Настройка профилей PMS

2.2.3.1 Профиль: paysystems

Необходимо активировать платёжные системы. Например – для активации двух таких систем Osmp и Rps – используемый json будет выглядеть так:

```
[
    {
        "paymentSystemIdent": "Osmp",
        "paymentSourceIdent": "online-Osmp",
        "currencyCode": null,
        "description": "Приём платежей через Обмен с ОСМП/QIWI",
        "protocol": "Osmp"
    },
    {
        "paymentSystemIdent": "Rps",
        "paymentSourceIdent": "online-Rps",
        "currencyCode": null,
        "description": "Приём платежей через РПС",
        "protocol": "Rps"
    }
]
```

Далее конфиг необходимо или передать в приложение в виде текста в параметре app.paysystems.json, или в виде файла, пусть к которому указывается в параметре app.paysystems.json-path. Первая настройка, в виде текста, имеет приоритет.

2.2.3.2 Профиль: registers-load

Аналогично платёжным системам, активные загрузчики онлайн реестров платежей описываются в json формате и передаётся либо в виде текста в параметре app.registers-load.json, или в виде файла, пусть к которому указывается в параметреapp.registers-load.json-path. Первая настройка, в виде текста, имеет приоритет.

Пример настройки двух парсеров:

```
[
    {
        "paymentSourceIdent": "register-genbank",
        "currencyCode": null,
        "description": "Реестр из Генбанка",
        "parser": "Genbank",
        "directory": {
            "ext": [
                ".txt"
            ],
            "dirIn": "${app.regs.workdir}/genbank/in",
            "dirError": "${app.regs.workdir}/genbank/error",
            "dirDone": "${app.regs.workdir}/genbank/done"
        }
    },
    {
        "paymentSourceIdent": "register-megabank",
        "currencyCode": null,
        "description": "Реестр из Мегабанка",
        "parser": "Megabank",
        "directory": {
            "ext": [
                ".txt"
            ],
            "dirIn": "${app.regs.workdir}/megabank/in",
            "dirError": "${app.regs.workdir}/megabank/error",
```

```
            "dirDone": "${app.regs.workdir}/megabank/done"
        }
    }
]
```

В данном случае активируется два парсера - Genbank и Megabank

2.2.3.3 Профиль: export-kafka

Выгрузка платежей в Kafka.

Используется, если задано значение настройки:

```
app.scheduler.payments.exportTo=Kafka
```

2.2.3.4 Профиль: recurrents

Конфиг активированных подсистем приёма рекуррентных платежей

Задаётся в виде json. Например:

```
[
    {
        "merchant": {
            "name": "CloudPayments",
            "ident": "CloudPaymentsIdent"
        },
        "paymentSourceIdent": "recurrent-CloudPayments",
        "currencyCode": "RUB",
        "description": "Приём рекуррентных платежей через CloudPayments",
        "protocol": "CloudPayments",
        "options": {
            "publicId": "~change-me~",
            "secretKey": "~change-me~",
            "updateTimeMinute": "15"
        }
    }
]
```

Далее данный конфиг необходимо или передать в приложение в виде текста в параметре app.recurrents.json, или в виде файла, пусть к которому указывается в параметре app.recurrents.json-path. Первая настройка, в виде текста, имеет приоритет.

2.3 Дополнительные сведения

Изменения на схему БД накатываются через Liquibase.

Активация новой платёжной системы производится путём модификации json файла активных платёжных систем и последующим рестартом сервиса. При работе под управлением K8, рекомендуется запускать 2 и более экземпляров приложения для обеспечения бесшовного рестарта сервиса.

2.4 Тестовый сценарий

Для добавления/изменения настроек реестров нужно изменить файл вида `/opt/pms/finances/config/registers.json` и перезапустить сервис `Finances`.

Этапы тестирования:

- Добавить файлы в директорию /opt/pms/finances/genbank/in (Тестовый файл genbank)
- Дождаться обработки файлов и выполнить запрос:

  ```
  SELECT * FROM finances.payments;
  ```

- После обработки файла платежи появятся с признаком payments.confirmed=false.

Дальше периодический процесс на стороне finances выполнит проверку на наличие контракта у сервиса Subscriber, и проставит payments.confirmed=true.

Примечание: для очистки данных использовать:

```
TRUNCATE TABLE finances.payments CASCADE;
TRUNCATE TABLE finances.register_file CASCADE;
TRUNCATE TABLE finances.registers CASCADE;
```