

Инструкция администратора

Лояльность для BPMSoft

Программные технологии

Содержание

Введение.....	3
Технические требования.....	4
Установка Лояльность для BPMSoft.....	5
Установка RabbitMQ.....	6
Установка PostgreSQL.....	7
Установка Docker	8
Установка сервисов	9
Процессинг покупок	9
Контакт-сервис и SMS сервис.....	10
Настройка конфигов.....	12
Процессинг покупок	12
Контакт-сервис и SMS сервис.....	13
Конфиги для теста.....	16
Установка и настройка Nginx	18
Подключение BPMSoft к процессингу	19
Лицензирование	22

Введение

В данной инструкции описан процесс разворачивания Лояльность для BPMSoft, сервиса процессинга, сервиса Core и Сервис отправки SMS и Viber сообщений 2.0 для BPMSoft..

Необходимо следующее ПО (рекомендуется устанавливать последние доступные версии, если не указана конкретная):

Для установки Лояльность для BPMSoft:

- BPMSoft Маркетинг или BPMSoft Bundle , версии 7.15.4+
- Redis Server
- База данных (PostgreSQL)

Для установки сервиса процессинга, контакт-сервиса и SMS-сервиса:

- Postgre Pro Standard 13
- RabbitMQ
- Docker

Технические требования

Процессинг рекомендуется разворачивать на unix-подобных операционных системах. Процесс установки будет описан для Ubuntu 18.04, процесс установки на других unix-подобных системах может отличаться лишь незначительно.

Минимальные технические требования

Для Лояльности для BPMSoft:

- BPMSoft Marketing или BPMSoft Bundle, версии 7.14.x до версии 7.16.3 (Если используется Лояльность для BPMSoft версии 2.14 и ниже)
- BPMSoft Marketing или BPMSoft Bundle, версии 7.16.4+ (Если используется Лояльность для BPMSoft 2.15 и выше)
- Redis 3.0 или 3.2
- PostgreSQL последней официальной версии на дату релиза BPMSoft

Для сервиса процессинга, контакт-сервиса и SMS-сервиса:

- Postgre Pro Standard 13 или выше (<https://postgrespro.ru/products/download>)
- RabbitMQ 3.5.x+ (<https://www.rabbitmq.com/download.html>)
- Docker 18.03.1 или выше

Рекомендуется использовать последние релиз-версии, если не указана конкретная. Разворачивать каждый компонент на отдельном сервере.

Установка BPMSoft

Установка BPMSoft в нашем случае ничем не отличается от стандартного процесса разворачивания, описанного в [инструкции](#).

Лояльность для BPMSoft 2.14 и ниже совместима с сборками BPMSoft Marketing или Bundle версии 7.14.x. до версии 7.16.3 включительно.

Если у вас установлена версия платформы BPMSoft 7.16.4 +, пакет продукт Лояльность для BPMSoft должен быть версии 2.15 и выше.

Для вашего удобства установочные файлы BPMSoft Marketing 7.17.3 для PostgreSQL на Linux доступны по [ссылке](#)

После разворачивания системы установите на нее пакеты приложения Лояльность для BPMSoft (пакет прилагается к инструкции)

1. В разделе [*Установка и удаление приложений*] нажмите кнопку [*Добавить приложение*] —> [*Установить из файла*].
2. На странице мастера установки приложений Marketplace нажмите кнопку [*Выбрать файл*] и укажите путь к файлам приложения. Установка запустится автоматически, как только вы выберете файл приложения.

В результате новое приложение будет добавлено в раздел [*Установка и удаление приложений*].

Далее необходимо добавить рабочее место Лояльность в число доступных.

Для перехода к **настройке рабочих мест**:

1. Перейдите в дизайнер системы, например, по кнопке .
2. В группе [*Настройка внешнего вида*] кликните по ссылке [*Настройка рабочих мест*].
3. Перейдите в рабочее место Лояльность
4. На панели инструментов детали [*Группы пользователей*] нажмите кнопку +. Откроется окно, в котором отображаются все организационные и функциональные роли пользователей
5. Отметьте роль All employees и нажмите кнопку [*Выбрать*]. Выбранная роль будет добавлена на деталь [*Группы пользователей*] рабочего места.

В результате все пользователи, которые входят в выбранные роли, получат доступ к рабочему месту.

Установка RabbitMQ

RabbitMQ – это приложение для работы с очередями сообщений **(message-queueing)**, еще его называют меседж брокер **(message broker)** или менеджер очередей **(queue manager)**. Простыми словами – это программное обеспечение, в котором могут быть определены очереди к которому могут подключаться различные приложения и передавать/получать сообщения.

Вам необходимо установить RabbitMQ на отдельный сервер (Linux) в соответствии с [инструкцией](#) на официальном сайте приложения. Перед установкой самого RabbitMQ необходимо установить esl-erlang, данный момент также содержится в инструкции. После успешной установки RabbitMQ в консоли необходимо запустить:

```
sudo service rabbitmq-server start
```

Для доступности консоли, позволяющей мониторить процессы сервера RabbitMQ через графический интерфейс, необходимо активировать плагин. Команда активации плагина:

```
sudo rabbitmq-plugins enable rabbitmq_management
```

Консоль будет доступна по адресу `http://имя-сервера:15672` (порт стандартный)

Примечание: инструкция предназначена для разворачивания боевой системы. Для теста системы все компоненты (RabbitMQ, PostgreSQL, Docker. Процессинг покупок, Контакт-сервис и SMS Сервис) можно развернуть на одной машине. Пример конфигов для разворачивания на одной машине представлен в разделе «Конфиги для теста»

Установка PostgreSQL

PostgreSQL — это свободно распространяемая объектно-реляционная система управления базами данных, наиболее развитая из открытых СУБД в мире и являющаяся реальной альтернативой коммерческим базам данных.

1. Установите PostgreSQL на отдельный сервер (Linux) для процессинга покупок в соответствии с инструкцией [на официальном сайте](#).
2. Установите PostgreSQL на отдельный сервер (Linux) для Контакт-сервиса и СМС сервиса в соответствии с инструкцией [на официальном сайте](#).

Примечание: инструкция предназначена для разворачивания боевой системы. Для теста системы все компоненты (RabbitMQ, PostgreSQL, Docker. Процессинг покупок, Контакт-сервис и SMS Сервис) можно развернуть на одной машине. Пример конфигов для разворачивания на одной машине представлен в разделе «Конфиги для теста»

Установка Docker

Docker — программное обеспечение с открытым исходным кодом, применяемое для разработки, тестирования, доставки и запуска веб-приложений в средах с поддержкой контейнеризации. Он нужен для более эффективного использования системы и ресурсов, быстрого развертывания готовых программных продуктов, а также для их масштабирования и переноса в другие среды с гарантированным сохранением стабильной работы.

1. Установите Docker на отдельный сервер (Linux) для процессинга покупок в соответствии с [инструкцией](#)
2. Установите Docker на отдельный сервер (Linux) для Контакт-сервиса и СМС сервиса в соответствии с [инструкцией](#).

Примечание: инструкция предназначена для разворачивания боевой системы. Для теста системы все компоненты (RabbitMQ, PostgreSQL, Docker. Процессинг покупок, Контакт-сервис и SMS Сервис) можно развернуть на одной машине. Пример конфигов для разворачивания на одной машине представлен в разделе «Конфиги для теста»

Установка сервисов

Процессинг покупок

Сервис процессинга предназначен для расчёта скидки, запроса бонусного баланса, подтверждения покупки, совершения оплаты бонусными баллами, совершения возвратов и тд. В качестве основного протокола используется HTTP, а значит, подходит для разработки на любом языке программирования, который умеет работать с HTTP-библиотеками. API процессинга является RPC и использует POST-запросы с телом в JSON формате. API всегда возвращает ответ в формате JSON. Сервис процессинга принимает запросы от касс, мобильных приложений или сайтов. Далее через RabbitMQ данные синхронизируются с BPMSoft. В случае инициализации запросов из BPMSoft данные в сервис процессинга передаются через RabbitMQ в порядке очереди

Для установки и запуска сервиса процессинга необходимо:

- установить BPMSoft, установить RabbitMQ,
- установить PostgreSQL для базы сервиса процессинга покупок,
- установить Docker на машине, где будет развернут сервис процессинга покупок и настроить конфигу.

После установки всего необходимого нужно создать и запустить Docker-compose.yml для процессинга покупок. Не забудьте поправить конфигу в соответствии с вашими настройками.

Проверить, что запуск процессинга прошел успешно, можно перейдя по адресу **http://имя_сервера:порт/health** (например, http://srv-processing1:5000/health)

При отсутствии подключения к RabbitMQ, либо к базе данных будет выведен соответствующий статус

```
{
  "status": "Unhealthy",
  "errors": {
    "Rabbit": {
      "status": "Unhealthy",
      "description": "NotConnected"
    },
    "DataBase": {
      "status": "Unhealthy",
      "description": "NotConnected"
    }
  }
}
```

Для того, чтобы сервис процессинга работал и после закрытия консоли, необходимо сделать процесс фоновым. Для этого запустите контейнер с указанием флага -d

```
$ sudo docker-compose up -d
```

Теперь сервис процессинга работает и после закрытия консоли.

Примечание: инструкция предназначена для разворачивания боевой системы. Для теста системы все компоненты (RabbitMQ, PostgreSQL, Docker. Процессинг покупок, Контакт-сервис и SMS Сервис) можно развернуть на одной машине. Пример конфигов для разворачивания на одной машине представлен в разделе «Конфиги для теста»

Контакт-сервис и SMS сервис

Контакт сервис применяется для регистрации клиентов в системе, получения и изменения данных профиля, получения информации о покупках и бонусном балансе клиентов. Контакт сервис принимает запросы от касс, мобильных приложений или сайтов. Далее через RabbitMQ данные синхронизируются с BPMSoft в порядке очереди. В случае инициализации запросов из BPMSoft данные в контакт сервис передаются через RabbitMQ в порядке очереди.

SMS сервис предназначен для верификации клиента по мобильному телефону. Касса, мобильное приложение или сайт направляет в Контакт сервис запрос на отправку кода подтверждения (sendCode), после чего Контакт сервис перенаправляет его в RabbitMQ. SMS сервис принимает сообщение из RabbitMQ и отправляет SMS через HTTP API [ZAGRUZKA](#), [Rapporto](#), [MTC-коммуникатор](#).

Вы можете развернуть все сервисы отдельно. Мы допускаем установку этих двух сервисов на одной машине.

Для установки и запуска Контакт-сервиса и SMS сервиса необходимо:

- установить BPMSoft, установить RabbitMQ,
- установить PostgreSQL для базы Контакт-сервиса и SMS сервиса,
- установить Docker на машине, где будут развернуты Контакт-сервиса и SMS сервиса и настроить конфиги.

После установки всего необходимого нужно создать и запустить Docker-compose.yml для Контакт-сервиса и SMS сервиса. Не забудьте поправить конфиги в соответствии с вашими настройками.

Проверить, что запуск процессинга прошел успешно, можно перейдя по адресу **http://имя_сервера:порт/health** (например, http://srv-personalarea:5010/health)

Для того, чтобы сервис процессинга работал и после закрытия консоли, необходимо сделать процесс фоновым. Для этого запустите контейнер с указанием флага -d

Примечание: инструкция предназначена для разворачивания боевой системы. Для теста системы все компоненты (RabbitMQ, PostgreSQL, Docker. Процессинг покупок, Контакт-сервис и SMS Сервис) можно развернуть на одной машине. Пример конфигов для разворачивания на одной машине представлен в разделе «Конфиги для теста»

Настройка конфигов

Процессинг покупок

Для корректной работы Лояльность для BPMSoft важно правильно настроить конфигурацию сервисов. На стороне сервиса процессинга покупок необходимо настроить все подключения.

Пример Docker-compose файла представлен ниже:

```
version: '3.3'

services:
  processing:
    image: registry.gitlab.com/samarasoft/samarasoft.loyalty/master
    environment:
      ASPNETCORE_URLS: "http://*:5000"
      DB_HOST: 172.20.102.11
      DB_PORT: 5435
      DB_NAME: [Адрес БД процессинга покупок]
      DB_USER: postgres
      DB_PASSWORD: 123
      RMQ_HOST: [Адрес RabbitMQ]
      RMQ_USER: admin
      RMQ_PASSWORD: password
      RMQ_QUEUE: LoyaltyProcessing
      API_AUTH_TOKEN: secret
    restart: on-failure
    ports:
      - "5000:5000"
    labels:
      NAME: "processing"
    networks:
      - mynetwork

networks:
  mynetwork:
```

DB_HOST — ip-адрес машины, на которой развернута Ваша база данных в формате x.x.x.x (например, 172.68.0.2)

DB_PORT — порт машины, который слушает подключение к Вашей базе данных (например, 5432)

DB_NAME — имя базы данных (например, loyalty)

DB_USER — имя пользователя Вашей базы данных (например, test)

DB_PASSWORD — пароль пользователя Вашей базы данных (например, test)

RMQ_HOST— ip-адрес машины, на которой развернут RabbitMQ (например, 172.68.0.3)

RMQ_USER — имя пользователя RabbitMQ (например, guest)

RMQ_PASSWORD— пароль пользователя RabbitMQ (например, guest)

RMQ_QUEUE — имя очереди, которая будет создана для передачи сообщений в процессинг (например, LoyaltyProcessing). При запуске процессинга будет создана автоматически с указанными в этом параметре именем.

API_AUTH_TOKEN — Ваш токен аутентификации (например, secret). Набор символов, задается пользователем в данном параметре, рекомендуется использовать сложный произвольный ряд символов одинаковый для кассы и процессинга.

Для удобства и простоты конфигурирования вы можете использовать переменные окружения в unix-системах.

Контакт-сервис и SMS сервис

Пример Docker-compose файла представлен ниже:

```
version: '3.3'
```

```
services:
```

```
  personal_area:
```

```
    image: registry.gitlab.com/samarasoft/samarasoft.loyalty.personalarea/master
    environment:
```

```
      ASPNETCORE_URLS: "http://*:5010"
```

```
      DB_HOST: [Адрес БД contact service]
```

```
      DB_PORT: 5435
```

```
      DB_NAME: personal_area
```

```
      DB_USER: postgres
```

```
      DB_PASSWORD: 123
```

```
      RMQ_HOST: [Адрес RabbitMQ]
```

```
      RMQ_USER: admin
```

```
      RMQ_PASSWORD: password
```

```
      RMQ_QUEUE: LoyaltyPersonalArea
```

```
      JWT_SIGNING_KEY: qecvnbjshisisdr47wduhfjxnkxnvcsdgfwern
```

```
      JWT_EXPIRES_SEC: 43200000
```

```
      API_AUTH_TOKEN: "secret"
```

```
    restart: on-failure
```

```
    ports:
```

```
      - "5010:5010"
```

```
    labels:
```

```
      NAME: "personalarea"
```

```
    networks:
```

```
      - mynetwork
```

```
  sms:
```

```
    image: registry.gitlab.com/samarasoft/samarasoft.sms/master
```

```

environment:
  ASPNETCORE_URLS: "http://*:5020"
  DB_HOST: [Адрес БД SMS]
  DB_PORT: 5432
  DB_NAME: sms
  DB_USER: admin
  DB_PASSWORD: password
  RMQ_HOST: [Адрес RabbitMQ]
  RMQ_USER: admin
  RMQ_PASSWORD: password
  RMQ_QUEUE: SamarasoftSms
  SERVICE_LOGIN: SmsServiceLogin
  SERVICE_PASSWORD: SmsServicePassword
  API_AUTH_TOKEN: secret
restart: on-failure
ports:
  - "5020:5020"
labels:
  NAME: "sms"
networks:
  - mynetwork

```

```

networks:
  mynetwork:

```

DB_HOST — ip-адрес машины, на которой развернута Ваша база данных в формате x.x.x.x (например, 172.68.0.2)

DB_PORT — порт машины, который слушает подключение к Вашей базе данных (например, 5432)

DB_NAME — имя базы данных (например, loyalty)

DB_USER — имя пользователя Вашей базы данных (например, test)

DB_PASSWORD — пароль пользователя Вашей базы данных (например, test)

RMQ_HOST — ip-адрес машины, на которой развернут RabbitMQ (например, 172.68.0.3)

RMQ_USER — имя пользователя RabbitMQ (например, guest)

RMQ_PASSWORD — пароль пользователя RabbitMQ (например, guest)

RMQ_QUEUE — имя очереди, которая будет создана для передачи сообщений в процессинг (например, LoyaltyProcessing). При запуске процессинга будет создана автоматически с указанными в этом параметре именем.

API_AUTH_TOKEN — Ваш токен аутентификации (например, secret). Набор символов, задается пользователем в данном параметре, рекомендуется использовать сложный произвольный ряд символов одинаковый для кассы и процессинга.

Для удобства и простоты конфигурирования вы можете использовать переменные окружения в unix-системах.

Конфиги для теста

Чтобы протестировать Лояльность для BPMSoft на локальном стенде вы можете RabbitMQ, PostgreSQL и Docker на одной машине и создать Docker-compose файл, включающий в себя все сервисы. Пример файла ниже

```
version: '2.4'
```

```
services:
```

```
  personal_area:
    image: registry.gitlab.com/samarasoft/samarasoft.loyalty.personalarea/master
    environment:
      ASPNETCORE_URLS: "http://*:5010"
      DB_HOST: postgres
      DB_PORT: 5432
      DB_NAME: personal_area
      DB_USER: stroilov
      DB_PASSWORD: 123QWEasd!
      RMQ_HOST: rabbitmq
      RMQ_USER: admin
      RMQ_PASSWORD: password
      RMQ_QUEUE: LoyaltyPersonalArea
      JWT_SIGNING_KEY: qecvnbjshisisdr47wduhfjxnkxnvcsdgfwern
      JWT_EXPIRES_SEC: 43200000
      API_AUTH_TOKEN: "secret"
    restart: on-failure
    ports:
      - "5010:5010"
    labels:
      NAME: "personalarea"
    depends_on:
      postgres:
        condition: service_started
      rabbitmq:
        condition: service_healthy
    networks:
      - mynetwork
```

```
  sms:
```

```
    image: registry.gitlab.com/samarasoft/samarasoft.sms/master
    environment:
      ASPNETCORE_URLS: "http://*:5020"
      DB_HOST: postgres
      DB_PORT: 5432
      DB_NAME: sms
      DB_USER: loyalty
      DB_PASSWORD: stroilov
      RMQ_HOST: rabbitmq
```



```

    RMQ_USER: admin
    RMQ_PASSWORD: password
    RMQ_QUEUE: SamarasoftSms
    SERVICE_LOGIN: SmsServiceLogin
    SERVICE_PASSWORD: SmsServicePassword
    API_AUTH_TOKEN: secret
restart: on-failure
ports:
  - "5020:5020"
labels:
  NAME: "sms"
networks:
  - mynetwork

processing:
  image: registry.gitlab.com/samarasoft/samarasoft.loyalty/master
  environment:
    ASPNETCORE_URLS: "http://*:5000"
    DB_HOST: postgres
    DB_PORT: 5432
    DB_NAME: loyalty
    DB_USER: stroilov
    DB_PASSWORD: 123QWEasd!
    RMQ_HOST: rabbitmq
    RMQ_USER: admin
    RMQ_PASSWORD: password
    RMQ_QUEUE: LoyaltyProcessing
    API_AUTH_TOKEN: secret
restart: on-failure
ports:
  - "5000:5000"
labels:
  NAME: "processing"
depends_on:
  postgres:
    condition: service_started
  rabbitmq:
    condition: service_healthy
networks:
  - mynetwork

rabbitmq:
  image: rabbitmq:3-management
  hostname: rabbitmq
  environment:
    RABBITMQ_DEFAULT_USER: "admin"
    RABBITMQ_DEFAULT_PASS: "password"
    RABBITMQ_DEFAULT_VHOST: "/"
  ports:
    - "15672:15672"
    - "5672:5672"
  labels:

```

```

    NAME: "rabbitmq"
  healthcheck:
    timeout: 5s
    interval: 5s
    retries: 5
    test:
      - "CMD"
      - "rabbitmqctl"
      - "status"
  networks:
    - mynetwork

postgres:
  image: postgres
  environment:
    POSTGRES_USER: stroilov
    POSTGRES_PASSWORD: 123QWEasd!
    POSTGRES_DB: loyalty
  volumes:
    - pgdata:/var/lib/postgresql/data
  ports:
    - "5432:5432"
  labels:
    NAME: "postgres"
  networks:
    - mynetwork

volumes:
  pgdata:

networks:
  mynetwork:

```

Установка и настройка Nginx

nginx — это HTTP-сервер и обратный прокси-сервер, почтовый прокси-сервер, а также TCP/UDP прокси-сервер общего назначения.

Данный этап требуется, если вы планируете открыть доступ внешним системам. Для тестирования в этом нет необходимости

Разверните Nginx на отдельном сервере в соответствии с [инструкцией](#) и настроить https (например через Certbot)

Настройте в Nginx перенаправление трафика на процессинг покупок Контакт-сервис и СМС сервис в соответствии с требованиями вашей компании.

Примечание: перед настройкой конфигов nginx проверьте nginx.conf, чтобы уточнить, откуда система подтягивает конфиги

Подключение Лояльность для BPMSoft к процессингу

Для подключения BPMSoft к процессингу в Дизайнере системы необходимо выбрать Настройка программы лояльности. Доступ к данному разделу регулируется системной операцией «CanManageAdministration».

Дизайнер системы

портала

Настройка страницы профиля организации



Импорт и интеграции ^

Импорт данных

Настройка интеграции с LDAP

Настройка интеграции с веб-сервисами

Журнал отправки email-рассылок

Интеграция с телефонией

Настройки смс сендера

Настройка чатов

Настройка сервиса трекинга событий

Настройка сервиса лидогенирации
социальных сетей

Настройка трекинга событий сайта

Настройка программы лояльности

Настройка карт лояльности

Сертификаты электронных кошельков

На странице отображается две детали: RabbitMQ, Тестирование акций.

Настройка программы лояльности

Сохранить

Отмена

^ Настройки подключения к процессингу лояльности

^ RabbitMQ

Адрес сервиса RabbitMQ*

stnd-load-01

Пользователь RabbitMQ*

admin

Пароль RabbitMQ*

RabbitMQ Virtual Host

loyaltys

● Подключение активно

*Если поле RabbitMQ Virtual Host пустое, подключение будет произведено к хосту по умолчанию

В детали RabbitMQ необходимо указать настройки подключения к процессингу через сервис RabbitMQ, имя сервера, на котором развернут Rabbit-MQ, пользователь и пароль RabbitMQ.

В детали Тестирование акций необходимо указать имя сервера, на котором развернут процессинг и токен аутентификации в соответствующие поля.

После сохранения настроек они применяются в течение минуты.

Подключение к процессингу можно проверить, сохранив верные настройки. Индикатор должен выглядеть так:

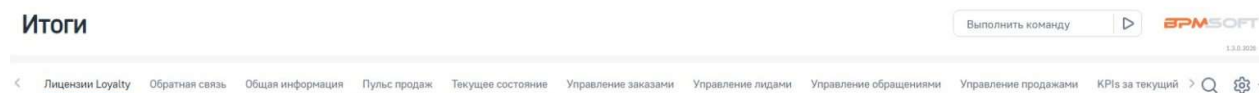
● Подключение активно

Лицензирование

Лицензии программы лояльности распространяются на активных пользователей программы.

Активным пользователем программы считается контакт, совершивший хотя бы одну покупку за последний год.

Текущее количество активных контактов и текущее количество приобретенных лицензий отображено в разделе [Итоги] на вкладке [Лицензии Лояльность для BPMSoft].



При превышении оплаченного лимита вводятся ограничения на запуск новых акций, создание и изменение объектов [Точка продаж] и [Промокод]. На остальные функции и объекты системы ограничения не накладываются.

Если количество доступных лицензий снизится до 10% от общего количества оплаченных лицензий, пользователям системы будет отправлено уведомление для обеспечения своевременного заказа новых лицензий.

