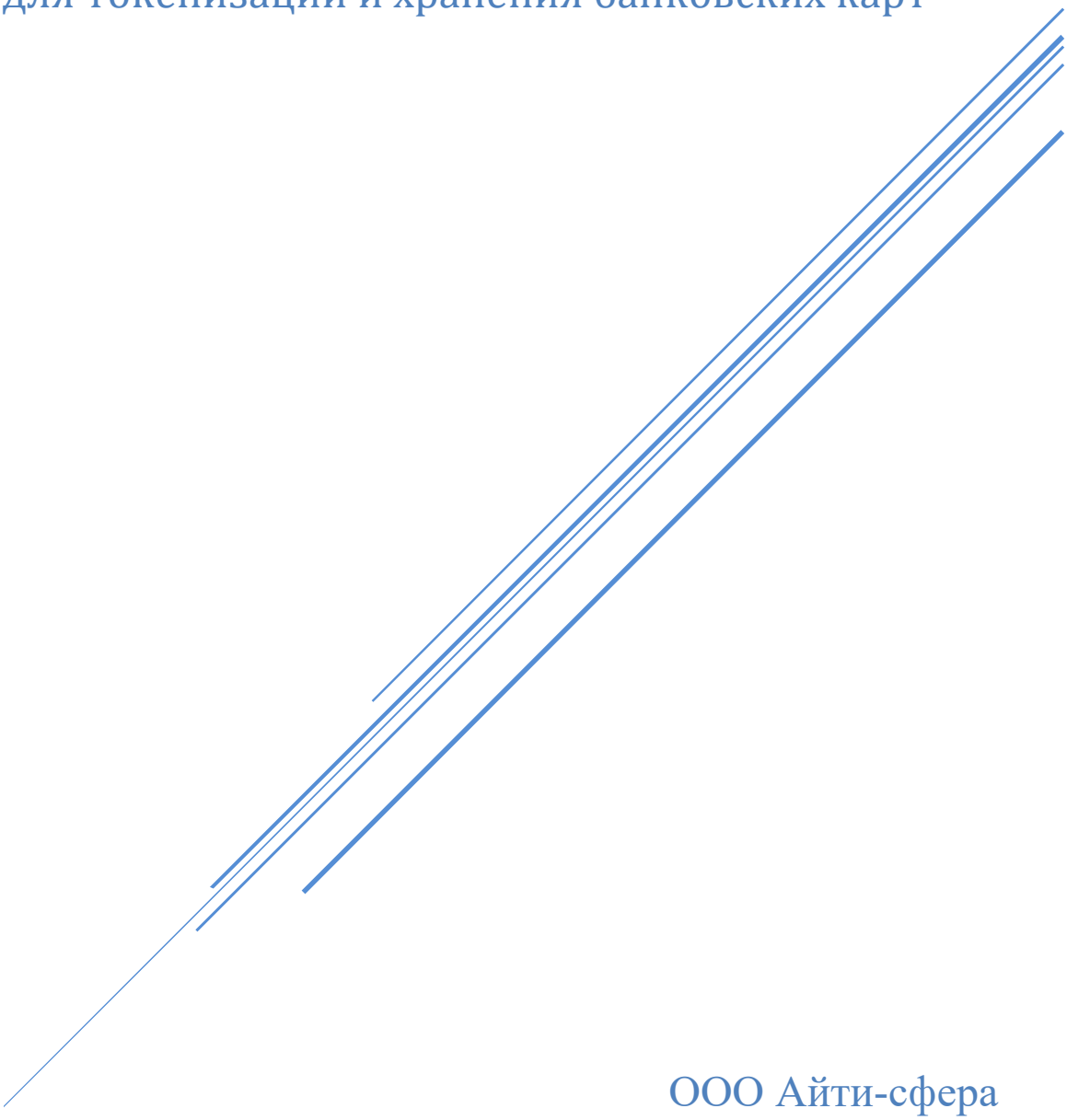
ИНСТРУКЦИЯ ПО РАЗВЕРТЫВАНИЮ ЭКЗЕМПЛЯРА ПО

Сервис для токенизации и хранения банковских карт



ООО Айти-сфера

Москва, 2023 г.

Оглавление

1. Введение

1.1 Наименование ПО

Наименование программного обеспечения “Сервис для токенизации и хранения банковских карт”.

1.2 Назначение и характеристика

“Сервис для токенизации и хранения банковских карт” должен представлять собой техническое решение, предоставляющее возможность безопасного хранения данных банковских карт (pan, card holder, expire).

1.3 Описание документа

Документ содержит сведения о порядке установки программного обеспечения “Сервис для токенизации и хранения банковских карт”. Также в документе приводятся требования к программному и аппаратному обеспечению.

1.4 Определения

ПО	Программное обеспечение – совокупность компьютерных программ и связанных с ними данных, которая содержит инструкции по указанию компьютеру, что и как делать.
Конфиденциальные данные	Информация, доступ к которой ограничен в соответствии с законами государства и нормами, которые компании устанавливаются самостоятельно.
Банковская карта	Пластиковая, металлическая или виртуальная карта, обычно привязанная к одному или нескольким расчётным счетам в банке. Используется для оплаты товаров и услуг, в том числе через интернет, с использованием бесконтактной технологии, совершения переводов, а также снятия наличных.
Держатель карты	Физическое лицо, на имя которого выпущена пластиковая или виртуальная карта, в том числе, физическое лицо-владелец счёта либо другое лицо, указанное владельцем счета. Сама карта при этом является собственностью банка.

Номер карты	Комбинация из 13, 16, 18 или 19 цифр, размещенных на лицевой или оборотной стороне пластика (или виртуальной карты). Набор цифр уникален, он не повторяется и привязан к кредитной организации, держателю пластика и счету. В комбинации зашифрована информация о банке, который выдал карту, и платежной системе, обслуживающей пластик.
API	Программный интерфейс приложения — это набор способов и правил, по которым различные программы общаются между собой и обмениваются данными.
Очередь	Совокупность объектов, которые поддерживаются в последовательности и могут быть изменены путем добавления объектов на одном конце последовательности и удаления объектов с другого конца последовательности. Операции очереди делают ее структурой данных, которая обеспечивает хранение и передачу двоичных данных между различными участниками системы.
Инстанс	Дубликат объекта, сохраняющий неразрывную связь с оригиналом и полную зависимость от него. Модификация любого образца равносильна модификации оригинала - результаты сказываются как на оригинале объекта, так и на всех образцах.
Мерчант	Партнер, пользователь ПО, имеющий специальный счёт, который позволяет торговой компании принимать платежи с помощью банковских карт.
Сервис	Независимо компилируемый программный модуль, динамически подключаемый к основной программе и предназначенный для расширения и/или использования её возможностей. Сервисы обычно выполняются в виде библиотек общего пользования.
Токенизация карты	Обмен конфиденциальных данных банковской карты на специальный токен, который позволяет использовать банковскую карту без ввода личных данных банковской карты.

1.5 Перечень программной документации

- “Сервис для токенизации и хранения банковских карт”. Техническое задание (ГОСТ 19.201-78);
- “Сервис для токенизации и хранения банковских карт”. Руководство пользователя;
- “Сервис для токенизации и хранения банковских карт”. Описание функциональных характеристик программного обеспечения;
- “Сервис для токенизации и хранения банковских карт”. Описание жизненного цикла разработки ПО;
- “Сервис для токенизации и хранения банковских карт”. Инструкция по развертыванию экземпляра ПО;
- “Сервис для токенизации и хранения банковских карт”. Инструкция для проверки в тестовой среде ПО.

2. Требования

2.1 Требования к эксплуатации серверной части

Обеспечение функционирования ПО серверной части “Сервис для токенизации и хранения банковских карт” реализовано на базе серверной операционной системы Linux. Минимальной конфигурацией аппаратной составляющей являются:

- ✓ Современная ОС: Linux;
- ✓ Оперативная память: 2 ГБ;
- ✓ Свободное дисковое пространство: не менее 20 Gb;
- ✓ Количество логических ядер процессора: 2;
- ✓ Частота процессора: 3.50 GHz.

Возможно разворачивание экземпляра ПО и на других ОС, поддерживающих платформу для автоматизации развёртывания и управления приложениями в средах с поддержкой контейнеризации Docker, например Windows 10 (Профессиональная или Корпоративная).

2.2 Уровень подготовки пользователей

Для интеграции API пользователь должен иметь квалификацию разработчика не ниже уровня Regular Middle.

3. Установка

3.1 Описание компонентов

Состав компонентов для установки:

- card-storage-service – сервис для сохранения личной информации банковских карт и токенизации карт;
- gateway-api – сервис для передачи информации от мерчантов и первичной обработки информации;
- rabbitmq-service – программный брокер сообщений на основе стандарта **AMQP**;
- db-card-storage – свободная объектно-реляционная система управления базами данных;
- db-pay – свободная объектно-реляционная система управления базами данных.

3.2 Порядок установки

Для разворачивания экземпляра ПО необходим docker-compose (инструментальное средство, входящее в состав **Docker**. Оно предназначено для решения задач, связанных с развертыванием проектов). Его можно установить, следуя инструкции на официальном сайте *https://docs.docker.com/compose/install/*.

Необходимо создать файл docker-compose.yml с содержимым:

```
version: '2.4'
services:

 cards-storage-service:
  image: 'docker.itisphera-test.ru/paymentboom_cards-storage-service:feat-import.22'
  restart: always
  environment:
    - NGINX_WORKERS=2
    - TRANSPORT_CONNECTION_PROTOCOL=amqp
    - TRANSPORT_CONNECTION_HOSTNAME=rabbitmq-service
    - TRANSPORT_CONNECTION_PORT=5672
    - TRANSPORT_CONNECTION_VHOST=/
    - TRANSPORT_CONNECTION_FRAME_MAX=0
    - TRANSPORT_CONNECTION_CHANNEL_MAX=0
    - TRANSPORT_CONNECTION_HEARTBEAT=0
    - TRANSPORT_AUTO_RECONNECT=true
    - TRANSPORT_AUTO_RECONNECT_TIMEOUT=5000
```

```
  - TRANSPORT_SERVICE_NAME=cards_storage
  - TRANSPORT_SHARED_SERVICE_QUEUE=true
  - TRANSPORT_SHARED_SERVICE_QUEUE_MESSAGE_TTL=10000
  - TRANSPORT_EXCLUSIVE_QUEUE_ACK=false
  - TRANSPORT_SHARED_SERVICE_QUEUE_ACK=false
  - TRANSPORT_MAIN_EXCHANGE_NAME=main
  - TRANSPORT_MAIN_REPLY_EXCHANGE_NAME=main_reply
  - TRANSPORT_PUBLISH_AWAIT_TIMEOUT=10000
  - TRANSPORT_PREFETCH=100
  - TRANSPORT_MAIN_SCHEDULE_EXCHANGE_NAME=main_schedule
  - TRANSPORT_MAIN_SCHEDULE_QUEUE_NAME=main_schedule
  - TRANSPORT_MAIN_GARBAGE_EXCHANGE_NAME=main_garbage
  - TRANSPORT_MAIN_GARBAGE_QUEUE_NAME=main_garbage
  - DB_CARDS_STORAGE_HOST=db-pay-service
  - DB_CARDS_STORAGE_PORT=5432
  - DB_CARDS_STORAGE_DATABASE=db_cards_storage
  - DB_CARDS_STORAGE_LOGGING=false
  - DB_CARDS_STORAGE_POOLSIZE=10
  - LOGGER_LEVEL=debug
  - DATABASE_MASTER_PASSWORD=STRONG_password
  - DATABASE_MASTER_USER=postgres
  - DATABASE_SLAVE_PASSWORD=STRONG_password
  - DATABASE_SLAVE_USER=postgres
  - K8S_POD_NAME=cards-storage-service-pod-name
  - CLIENTS={}
  - DB_CARDS_STORAGE_USER=postgres
  - DB_CARDS_STORAGE_PASSWORD=STRONG_password
  - CRYPTO_INTERNAL_KEY=test2
  - TRANSPORT_CONNECTION_PASSWORD=rmpassword
  - TRANSPORT_CONNECTION_USERNAME=rmuser
ports:
  - "3000:3000"
depends_on:
  - rabbitmq-service
  - db-pay-service
```

```
gateway-api-service:
 image: 'docker.itisphera-test.ru/paymentboom_gateway-api:feat-import.194'
 restart: always
 environment:
  - NGINX_WORKERS=2
  - API_NAME=gateway-api
  - API_VERSION=v1
  - API_PORT=8080
  - TRANSPORT_INIT_TIMEOUT=5000
  - TRANSPORT_CONNECTION_HOSTNAME=rabbitmq-service
  - TRANSPORT_CONNECTION_PORT=5672
  - TRANSPORT_CONNECTION_VHOST=/
  - TRANSPORT_AUTO_RECONNECT=true
  - TRANSPORT_AUTO_RECONNECT_TIMEOUT=5000
  - TRANSPORT_SERVICE_NAME=rmuser
  - TRANSPORT_PUBLISH_AWAIT_TIMEOUT=15000
  - SETTINGS_TRANSPORT_SERVICES_NAMES_PAYMENT_STATUS=ph_payment-status
  - SETTINGS_TRANSPORT_SERVICES_NAMES_PAYMENT_NOTIFICATION=ph_payment-notification
  - SETTINGS_TRANSPORT_SERVICES_NAMES_PAYMENT_REGISTER=ph_payment-registration
  - SETTINGS_TRANSPORT_SERVICES_NAMES_SECOND_CPS=second-cps_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_ECOMMPAY=ecommpay_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_ASTROPAY_DIRECT=directa24_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_FIRST_CPS=first-cps_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_QIWI_TOP_UP_CARD=qiwi-top-up-card_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_ASTROPAY_CARD=astropay-card_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_CONTACT=contact_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_ECOPAYZ=ecopayz_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_WIRECARD=wirecard_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_HALYK_BANK=halyk-bank_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_RBK_MONEY=rbk_money_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_PAYEER=payeer_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_PAYSAGE=paysage_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_DAOWALLET=daowallet_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_SETTLE_PAY=settle_pay_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_ASTROPAY_ONE_TOUCH=astopay_onetouch_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_COINSPAID=coinspaid_plugin
  - SETTINGS_TRANSPORT_SERVICES_NAMES_REFERENCE_API=ph_reference-service
```

- SETTINGS_TRANSPORT_SERVICES_NAMES_APPLICATION_BALANCES_SERVICE=ph_application-balances-service
- SETTINGS_TRANSPORT_SERVICES_NAMES_JETON=jeton_plugin
- SETTINGS_TRANSPORT_SERVICES_NAMES_LEOGAMING=leogaming_plugin
- SETTINGS_TRANSPORT_SERVICES_NAMES_APS=aps_plugin
- SETTINGS_TRANSPORT_SERVICES_NAMES_MUCHBETTER=muchbetter_plugin
- SETTINGS_TRANSPORT_SERVICES_NAMES_EZEEWALLET=ezeewallet_plugin
- SETTINGS_TRANSPORT_SERVICES_NAMES_BTQFINANCE=btqfinance_plugin
- SETTINGS_TRANSPORT_SERVICES_NAMES_CONNECTUM=connectum_plugin
- SETTINGS_TRANSPORT_SERVICES_NAMES_INTERKASSA=interkassa_plugin
- SETTINGS_TRANSPORT_SERVICES_NAMES_BCPO=bcpo_plugin
- SETTINGS_TRANSPORT_SERVICES_NAMES_PAYDEX=paydex_plugin
- SETTINGS_TRANSPORT_SERVICES_NAMES_GETAPAY_PSP=getapay_psp_plugin
- SETTINGS_TRANSPORT_SERVICES_NAMES_JETONCASH=jetoncash_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_INWIZO=inwizo_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_PAYCOS=paycos_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_PAYMEGA=paymega_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_WALLET_COM=wallet_com_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_MERCURYO=mercuryo_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_FINANA=finana_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_DECARDS2P=decards2p_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_P2PAY=p2pay_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_PAYDO=paydo_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_ZIPPY=zippy_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_0XPROCESSING=oxprocessing_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_CONNPAY=connpay_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_PAY_PORT=pay_port_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_GATE_EXPRESS=gate_express_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_PAYMENTS_ROUTER=payments_router
- SETTINGS_TRANSPORT_SERVICES_NAMES_PAYES=payes_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_PAY4FUN=pay4fun_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_PAYRETAILERS=payretailers_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_CYPIX=cypix_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_JUMPFINANCE=jumpfinance_integration_service
- SETTINGS_TRANSPORT_SERVICES_NAMES_MAGIC_PAYMENTS=magic_payments_integration_service
- DB_MASTER_NAME=db_pay
- DB_MASTER_HOST=db-pay-service
- DB_MASTER_PORT=5432

```
  - DB_MASTER_CONN_TIMEOUT_MILLIS=3000
  - DB_MASTER_IDLE_TIMEOUT_MILLIS=10000
  - DB_MASTER_MAX_CLIENTS=10
  - DB_MASTER_PING_TIMEOUT_MILLIS=60000
  - DB_SLAVE_NAME=db_pay
  - DB_SLAVE_HOST=db-pay-service
  - DB_SLAVE_PORT=5432
  - DB_SLAVE_CONN_TIMEOUT_MILLIS=3000
  - DB_SLAVE_IDLE_TIMEOUT_MILLIS=10000
  - DB_SLAVE_MAX_CLIENTS=10
  - DB_SLAVE_PING_TIMEOUT_MILLIS=60000
  - LOG_LEVEL=0
  - CARD_ENCRYPTION_KEY=enckey
  - DATABASE_MASTER_PASSWORD=STRONG_password
  - DATABASE_MASTER_USER=postgres
  - DATABASE_SLAVE_PASSWORD=STRONG_password
  - DATABASE_SLAVE_USER=postgres
  - TRANSPORT_CONNECTION_PASSWORD=rmpassword
  - TRANSPORT_CONNECTION_USERNAME=rmuser
  - K8S_POD_NAME=gateway-api-pod-name
  - DB_MASTER_USER=postgres
  - DB_MASTER_PASSWORD=STRONG_password
  - DB_SLAVE_USER=postgres
  - DB_SLAVE_PASSWORD=STRONG_password
 ports:
  - "8080:8080"
 depends_on:
  - rabbitmq-service
  - db-pay-service



rabbitmq-service:


 image: rabbitmq:3.11.8-management
 hostname: rabbitmq-service
 restart: always
```

```
 environment:
   - RABBITMQ_DEFAULT_USER=rmuser
   - RABBITMQ_DEFAULT_PASS=rmpassword
   - RABBITMQ_SERVER_ADDITIONAL_ERL_ARGS=-rabbit channel_max 0
 volumes:
   - ./rabbitmq:/var/lib/rabbitmq
 command:
   - bash
   - -c
   - |
     cd /opt/rabbitmq/plugins/
     apt update
     apt install -y wget
     wget https://github.com/rabbitmq/rabbitmq-delayed-message-
exchange/releases/download/3.11.1/rabbitmq_delayed_message_exchange-3.11.1.ez
     rabbitmq-plugins enable rabbitmq_delayed_message_exchange
     rabbitmq-server start
 ports:
   - "15672:15672"
   - "5672:5672"


db-pay-service:


 container_name: db-pay-service
 image: 'postgres:14'
 environment:
   POSTGRES_PASSWORD: STRONG_password
   POSTGRES_USER: postgres
   POSTGRES_DB: db_pay
 restart: on-failure
 volumes:
   - ./postgres:/var/lib/postgresql/
 ports:
   - "5432:5432"
 command: [ "postgres", "-c", "wal_level=logical" ]


volumes:
```

```
rabbitmq:
  external: true
postgres:
  external: true
```

Запускается командой:

```
docker-compose up -d
```