

Правообладатель: ООО «Глобал Пэй»

**ОПИСАНИЕ ФУНКЦИОНАЛЬНЫХ ХАРАКТЕРИСТИК
ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ
И ИНФОРМАЦИЯ, НЕОБХОДИМАЯ ДЛЯ ЕГО УСТАНОВКИ
И ЭКСПЛУАТАЦИИ**

**Программный комплекс «Тессера»
— платформа выпуска и обслуживания виртуальных карт
(ПК «Тессера»)**

Версия 0.51.0

Документ подготовлен для включения программного обеспечения
в единый реестр российских программ для электронных вычислительных машин
и баз данных (постановление Правительства Российской Федерации № 1236)

Содержание

1. Общие сведения
 1. Назначение программного обеспечения
 2. Область применения
 3. Состав документа и границы описания
2. Функциональные характеристики
 1. Общее описание функциональности
 2. Состав функциональных блоков
 3. Основные пользовательские сценарии
 4. Программные интерфейсы
3. Архитектура и состав программного обеспечения
 1. Компонентный состав
 2. Технологическая основа
 3. Внутренняя организация серверной части
4. Требования к среде функционирования
 1. Требования к аппаратному обеспечению
 2. Требования к системному программному обеспечению
 3. Внешние взаимодействия
5. Установка программного обеспечения
 1. Подготовка среды
 2. Получение дистрибутива
 3. Конфигурирование
 4. Развёртывание компонентов
 5. Инициализация данных
 6. Проверка работоспособности
6. Эксплуатация программного обеспечения
 1. Запуск и остановка
 2. Обработка фоновых заданий
 3. Регламентные операции
 4. Контроль состояния
 5. Обновление версии
 6. Типовые нештатные ситуации
7. Требования к квалификации персонала

1. Общие сведения

1.1. Назначение программного обеспечения

Программный комплекс «Тессера» — платформа выпуска и обслуживания виртуальных карт (далее — ПК «Тессера», ПО, программа) предназначено для выпуска и обслуживания виртуальных платёжных карт и обслуживания связанных с ними денежных операций. Программа обеспечивает полный жизненный цикл виртуальной карты: подбор доступного карточного продукта, выпуск карты через внешнего эмитента, приём оплаты выпуска и пополнения через российский платёжный шлюз по Системе быстрых платежей, отображение баланса и истории операций, возвраты денежных средств, а также сопутствующие сервисы — реферальную и партнёрскую программы, уведомления и аналитический учёт.

Программа реализована как серверное приложение с программным интерфейсом (API) и набором клиентских приложений. Основной канал взаимодействия с конечным пользователем — мини-приложение внутри мессенджера; дополнительно поддерживаются веб-режим и режим мультибрендового использования (white-label), при котором организация-партнёр публикует собственное мини-приложение под своим брендом, работающее на той же серверной части.

1.2. Область применения

Область применения — финансовые сервисы для физических лиц и организаций-партнёров:

- предоставление конечным пользователям виртуальных платёжных карт для оплаты товаров и услуг;
- приём платежей в рублях Российской Федерации и их конвертация при пополнении карт;
- предоставление организациям-партнёрам инструментов для запуска собственных брендированных сервисов на общей платформе;
- учёт доходов и расходов по операциям, формирование отчётности и передача аналитических событий во внешние системы аналитики.

1.3. Состав документа и границы описания

Документ описывает программу версии 0.51.0 (от 03.09.2026). При выпуске новых версий программы документ актуализируется. В состав программы входят компоненты, перечисленные в разделе 3.1. Административная панель правообладателя и иные приложения, взаимодействующие с программой через её программные интерфейсы, в состав программы не входят.

Документ содержит описание функциональных характеристик программы (разделы 2 и 3), требования к среде функционирования (раздел 4), сведения, необходимые для установки экземпляра программы (раздел 5), и сведения, необходимые для её эксплуатации (раздел 6).

В настоящем документе не приводятся: наименования организаций-контрагентов и реквизиты договоров с ними; сетевые адреса, доменные имена и реквизиты доступа к промышленному контуру правообладателя; значения ключей и паролей. Указанные сведения не требуются для установки и эксплуатации экземпляра программы и предоставляются правообладателем отдельно в порядке, установленном договором.

Описание процессов, обеспечивающих поддержание жизненного цикла программного обеспечения, включая процессы устранения неисправностей, совершенствования и технической поддержки, приводится в отдельном документе.

2. Функциональные характеристики

2.1. Общее описание функциональности

Функциональность программы охватывает четыре укрупнённых направления.

Карточный контур. Управление каталогом доступных карточных продуктов, выпуск виртуальной карты через внешнего эмитента, выдача реквизитов карты пользователю, отображение остатка, синхронизация операций по карте с системой эмитента, блокировка и закрытие карты, перевыпуск карты и перенос остатка между картами одного пользователя.

Платёжный контур. Приём платежей за выпуск и пополнение карт через платёжный шлюз по Системе быстрых платежей, маршрутизация платежей между платёжными терминалами, рекуррентные (сохранённые) способы оплаты, возвраты денежных средств, конвертация валют по курсу, антифрод-проверки платежей.

Клиентский контур. Регистрация и аутентификация пользователя, ведение профиля, подтверждение адреса электронной почты, промокоды и скидки, реферальная программа, уведомления по нескольким каналам доставки, сообщения внутри приложения, поддержка пользователей.

Партнёрский и учётный контур. Ведение организаций-партнёров и их мультибрендовых мини-приложений через программный интерфейс партнёрского контура, партнёрская программа с оплатой за действие, учёт доходов и расходов по каждой финансово значимой операции, формирование отчётности, передача аналитических событий во внешние системы, служебный программный интерфейс для операционного управления.

2.2. Состав функциональных блоков

Серверная часть программы построена как модульная система. Каждый функциональный блок реализован отдельным модулем с собственной областью ответственности и границами. Ниже приведён состав функциональных блоков и решаемые ими задачи.

Функциональный блок	Решаемые задачи
Виртуальные карты	Каталог карточных продуктов, доступных для выпуска; выпуск карты через внешнего эмитента; хранение и выдача реквизитов карты; получение остатка и операций по карте; блокировка, разблокировка и закрытие карты; перевыпуск и перенос остатка. Крупнейший блок программы
Платежи	Приём платежей через Систему быстрых платежей; управление платёжными терминалами и маршрутизацией; рекуррентные токены и автоплатежи; возвраты; антифрод-проверки

Функциональный блок	Решаемые задачи
Входящие уведомления платёжных систем	Приём, проверка подлинности, сохранение и асинхронная обработка входящих уведомлений (вебхуков) от платёжных систем
Пользователи	Регистрация, аутентификация, профиль пользователя, смена адреса электронной почты, слияние учётных записей, события жизненного цикла пользователя
Веб-аутентификация	Вход по адресу электронной почты с одноразовым кодом подтверждения, ограничение частоты попыток, выдача токенов доступа
Интеграция с мессенджерами	Авторизация пользователя внутри мини-приложения, приём событий бота, доставка уведомлений, подтверждение адреса электронной почты кодом
Уведомления	Асинхронная доставка уведомлений по каналам: мессенджеры, браузерные push-уведомления, прямые push-уведомления мобильного приложения для операционных систем iOS и Android
Сообщения пользователю	Информационные и предупреждающие сообщения внутри приложения с признаком прочтения и возможностью удаления
Реферальная программа	Реферальные коды, привязка приглашённого пользователя, настройки вознаграждений, начисление и вывод реферального баланса на карту
Партнёрская программа (оплата за действие)	Учёт источника привлечения пользователя, отправка уведомлений партнёрам о событиях, начисление вознаграждений за выпуск карты и за операции по ней
Партнёрский контур и мультибрендовость	Программный интерфейс партнёрского контура: организации-партнёры, их брендированные мини-приложения и пользователи партнёров, привязка пользователей к мини-приложению, процесс поэтапного возврата средств клиентам. Отдельного пользовательского интерфейса для партнёра в составе программы нет — взаимодействие выполняется через программный интерфейс
Учёт доходов и расходов	Формирование учётной записи по каждой финансово значимой операции: платёж, комиссия, пополнение карты, вознаграждение, возврат. Полностью асинхронная обработка на доменных событиях
Курсы валют	Получение, кеширование и применение курсов валют из внешних источников; выбор источника курса; защита от аномальных значений
Отчётность по операциям	Формирование единого списка операций пользователя из разнородных источников — платежей, торговых операций, бонусов — и приведение к общему формату
Уведомления о предстоящих списаниях	Определение регулярных списаний внешних сервисов по истории операций и предупреждение пользователя о предстоящем списании при нехватке остатка
Акционные механики	Предложения по улучшению карточного продукта, компенсационные механики при закрытии карты эмитентом, сезонные и персональные предложения

Функциональный блок	Решаемые задачи
Промокоды и скидки	Применение промокода к выпуску и пополнению карты, расчёт итоговой стоимости
Профиль клиента и внешние CRM	Ведение атрибутов профиля клиента, пороговые счётчики, сценарии по неактивности и жизненному циклу, доставка событий во внешние системы работы с клиентами
Сопоставление торговых точек	Приведение технических наименований торговых точек из платёжных систем к человекочитаемым наименованиям для отображения пользователю
Аналитика	Сбор аналитических событий, их хранение и асинхронная передача во внешнюю систему веб-аналитики; реестр событий и параметров
Безопасность	Блокировка и разблокировка пользователей, списки разрешений, защита уровня HTTP-запросов
Списки разрешённых пользователей	Ограничение доступа к программным интерфейсам на непромышленных средах по идентификатору пользователя или адресу электронной почты. На промышленной среде проверка отключена
Короткие ссылки	Упаковка произвольного набора параметров перехода в компактный токен и его применение при первом входе пользователя
Служебный интерфейс администрирования	Программный интерфейс операционного управления картами, пользователями и связанными сущностями для внешних средств администрирования. Доступ по отдельному служебному токenu, независимо от пользовательской аутентификации
Служебные команды	Регламентные операции по расписанию и разовые операции обслуживания, запускаемые администратором

2.3. Основные пользовательские сценарии

Выпуск виртуальной карты

1. Пользователь открывает клиентское приложение и проходит аутентификацию.
2. Программа возвращает перечень карточных продуктов, доступных для выпуска в данном мини-приложении.
3. Пользователь выбирает продукт; программа рассчитывает стоимость выпуска с учётом курса валют и применённого промокода.
4. Программа создаёт платёж и возвращает клиентскому приложению ссылку на оплату по Системе быстрых платежей.
5. После подтверждения оплаты платёжной системой программа принимает входящее уведомление, проверяет его подлинность и переводит платёж в оплаченное состояние.
6. Программа обращается к внешнему эмитенту за выпуском карты. Обращение выполняется фоновым заданием с повторными попытками.
7. При успешном выпуске карта переводится в рабочее состояние, пользователю направляется уведомление, формируются учётные записи по доходам и расходам и аналитические события.
8. При отказе эмитента после состоявшейся оплаты платёж переводится в возврат.

Пополнение карты

1. Пользователь указывает сумму пополнения; программа рассчитывает итоговую сумму с учётом курса и комиссий.
2. Создаётся платёж; оплата выполняется по Системе быстрых платежей либо ранее сохранённым способом оплаты.
3. После подтверждения оплаты программа направляет эмитенту команду на зачисление средств на карту.
4. Остаток карты обновляется, пользователю направляется уведомление, операция отражается в учёте.
5. При отказе эмитента с терминальным кодом (например, сумма ниже минимально допустимой) платёж направляется в возврат без ожидания повторных попыток.

Возврат денежных средств

1. Возврат инициируется автоматически (несостоявшийся выпуск, отказ зачисления) либо оператором через служебный интерфейс администрирования.
2. Программа формирует заявку на возврат и направляет её в платёжную систему.
3. Состояние возврата отслеживается по входящим уведомлениям платёжной системы.
4. Результат отражается в учёте и в истории операций пользователя.

Реферальная программа

1. Пользователь активирует участие в программе и получает персональный реферальный код.
2. Приглашённый пользователь регистрируется по коду; программа фиксирует привязку.
3. При наступлении вознаграждаемого события начисляется вознаграждение на реферальный баланс.
4. Пользователь выводит накопленный баланс на собственную карту.

2.4. Программные интерфейсы

Программа предоставляет интерфейс прикладного программирования на основе протокола HTTP с обменом данными в формате JSON. Актуальная версия интерфейса — вторая (/api/v2).

Доступ к защищённым методам предоставляется по токену доступа. Обработка запроса последовательно проходит проверку токена, проверку отсутствия блокировки пользователя, проверку прав доступа к мини-приложению и проверку разрешения на конкретную операцию. Разрешения выдаются пооперационно: просмотр профиля, просмотр карт, выпуск карты, пополнение карты, просмотр доступных карточных продуктов, просмотр операций, получение курсов валют, участие в реферальной программе.

Основные группы методов:

Группа	Назначение
Карты	Выпуск карты, расчёт стоимости выпуска и пополнения, список карт пользователя, сведения по карте, получение реквизитов, пополнение
Каталог продуктов	Перечень карточных продуктов, доступных в мини-приложении
Операции и отчёты	Список операций по картам, отчётный список операций
Подтверждение операций	Приём и применение подтверждений операций по протоколу 3-D Secure (при поддержке протокола эмитентом)
Платежи	Список платежей пользователя
Пользователь	Текущий пользователь, состояние участия в реферальной программе, отчёт и его детализация, активация программы, отметка прохождения онбординга, сообщения пользователю
Сохранённые способы оплаты	Список сохранённых способов оплаты и их удаление
Промокоды	Применение промокода

Программа поддерживает автоматическую генерацию спецификации интерфейса в формате OpenAPI. Генерация включается параметром конфигурации (SCRAMBLE_ENABLED); спецификация доступна по служебному адресу экземпляра только с сетевых адресов, перечисленных в параметре SCRAMBLE_ALLOWED_IPS.

Отдельный служебный интерфейс администрирования защищён статическим служебным токеном и предназначен для подключения внешних средств операционного управления. Доступ к нему предоставляется только из доверенного сегмента сети. Административная панель правообладателя является самостоятельным приложением, взаимодействует с программой через этот интерфейс и в состав программы не входит.

3. Архитектура и состав программного обеспечения

3.1. Компонентный состав

Программа состоит из следующих компонентов, каждый из которых поставляется в виде образа контейнера:

Компонент	Назначение
Серверное приложение	Обработка запросов программного интерфейса, бизнес-логика, взаимодействие с внешними системами. Основной компонент программы
Обработчик фоновых заданий	Исполнение асинхронных заданий из очередей: обращения к внешним системам, доставка уведомлений, обработка входящих уведомлений, учётные операции
Планировщик	Запуск регламентных операций по расписанию
Сервер событий реального времени	Доставка событий клиентским приложениям по протоколу WebSocket
Клиентское приложение	Пользовательский интерфейс. Единая кодовая база собирается в двух вариантах: мини-приложение внутри мессенджера и веб-режим для работы через браузер; вариант задаётся при сборке
Вспомогательные служебные сервисы	Получение сведений из внешних источников, в том числе курсов валют
Обратный прокси-сервер	Терминирование TLS, маршрутизация запросов к компонентам, раздача статических файлов клиентских приложений

Хранилища данных — реляционная система управления базами данных и хранилище данных в оперативной памяти — являются внешними по отношению к программе и разворачиваются отдельно (см. раздел 4.2).

3.2. Технологическая основа

Элемент	Технология
Язык программирования серверной части	PHP версии 8.4
Каркас серверной части	Laravel 11, архитектура «сначала интерфейс» (API-first)
Сервер приложений	FrankenPHP (сервер приложений со встроенным интерпретатором PHP)
Язык программирования клиентской части	TypeScript
Библиотека пользовательского интерфейса	React 19
Управление состоянием клиентской части	Effector
Сборка клиентской части	Vite
Архитектура клиентской части	Feature-Sliced Design
Система управления базами данных	MySQL 9.3
Кеш и очереди заданий	Redis
Аутентификация	Токены доступа с пооперационными разрешениями (Laravel Sanctum)
Средства контейнеризации	Docker, Docker Compose
Автоматизированное тестирование	PHPUnit (серверная часть); Vitest и Playwright (клиентская часть: модульные и сквозные тесты)

3.3. Внутренняя организация серверной части

Серверная часть представляет собой модульный монолит. Каждый функциональный блок из раздела 2.2 реализован отдельным модулем с типовой внутренней структурой (набор каталогов зависит от модуля):

```

app/{Модуль}/
├── Actions/      – бизнес-операции; возвращают унифицированный результат
│   └── DTOs/     – объекты передачи данных операций
├── Domain/       – доменная логика: события, объекты-значения, примеси
├── External/     – интеграции и слой доступа к данным
│   └── DB/Repositories/
├── Http/
│   ├── Controllers/ – обработчики запросов без бизнес-логики
│   ├── Requests/   – правила проверки входных данных
│   └── Resources/   – правила формирования ответа
├── Jobs/         – фоновые задания
├── Listeners/    – обработчики доменных событий
└── Models/       – модели данных

```

Приняты следующие архитектурные соглашения:

- проверка входных данных выполняется декларативно, до вызова бизнес-логики;
- бизнес-логика сосредоточена в операциях, возвращающих унифицированный результат с кодом исхода;
- изменение данных и постановка фоновых заданий выполняются согласованно: задания ставятся в очередь только после успешного завершения транзакции базы данных;
- обращение к внутренним элементам другого модуля запрещено — взаимодействие идёт через явные точки интеграции: операцию, интерфейс или доменное событие;
- сведения ограниченного доступа, включая реквизиты карт, хранятся в зашифрованном виде.

Фоновая обработка построена на очередях заданий. Задания разделены по назначению (обращения к внешним системам, уведомления, учёт, аналитика), что позволяет масштабировать обработку и изолировать сбои одного направления от остальных. Для заданий, обращающихся к внешним системам, заданы ограничения на число повторных попыток, интервал между ними и предельное время исполнения; задания спроектированы идемпотентными — повторный запуск не приводит к дублированию операций.

4. Требования к среде функционирования

4.1. Требования к аппаратному обеспечению

Программа не предъявляет требований к специализированному оборудованию и функционирует на серверах общего назначения с архитектурой x86-64.

Параметр	Минимально	Рекомендуется для промышленной эксплуатации
Процессор	4 ядра	16 ядер и более (в промышленном контуре правообладателя — 32 ядра)
Оперативная память	8 ГБ	32 ГБ и более (в промышленном контуре правообладателя — 62 ГБ)
Дисковое пространство	40 ГБ	определяется объёмом хранимых операций и журналов; для компонентов программы — от 40 ГБ, хранилища данных размещаются отдельно
Сетевое подключение	Постоянное подключение к сети передачи данных с возможностью исходящих соединений к внешним системам по протоколу HTTPS	

4.2. Требования к системному программному обеспечению

Компонент среды	Требование
Операционная система	Операционная система семейства GNU/Linux с поддержкой контейнеризации. Проверенная конфигурация — Ubuntu Server 24.04 LTS
Среда контейнеризации	Docker Engine версии 24 и выше, Docker Compose версии 2 и выше
Интерпретатор PHP	Версия 8.4 и выше. Поставляется в составе образа контейнера серверного приложения, отдельная установка не требуется
Расширения PHP	dom, openssl, pdo, simplexml. Входят в состав образа контейнера
Система управления базами данных	MySQL версии 9.3 или совместимая. Может разворачиваться как в составе комплекта поставки, так и в виде внешнего управляемого сервиса
Хранилище данных в оперативной памяти	Redis версии 7 и выше — используется как кеш и как хранилище очередей заданий
Обратный прокси-сервер	nginx версии 1.26 и выше либо иной совместимый — терминирование TLS и маршрутизация
Средства сборки клиентской части	Node.js версии 22 и выше, менеджер пакетов npm — требуются только при самостоятельной сборке клиентских приложений из исходного текста

Все перечисленные компоненты среды распространяются на условиях свободных лицензий и не требуют приобретения лицензий у иностранных правообладателей.

4.3. Внешние взаимодействия

Для полнофункциональной работы экземпляра программы требуется наличие действующих подключений к внешним системам следующих типов:

- система эмитента платёжных карт — выпуск карт и операции по ним;
- платёжный шлюз, обеспечивающий приём платежей по Системе быстрых платежей;
- платформа мессенджера — размещение мини-приложения и доставка уведомлений;
- источник курсов валют;
- внешняя система веб-аналитики (необязательно);
- внешние системы работы с клиентами и технической поддержки (необязательно).

Реквизиты подключения к внешним системам задаются параметрами конфигурации (раздел 5.3) и предоставляются правообладателем. Программа предусматривает режим изолированной работы без обращения к внешним системам (см. раздел 5.3), предназначенный для проверки развёртывания и демонстрации.

5. Установка программного обеспечения

Установка выполняется администратором, обладающим квалификацией согласно разделу 7. Приведённый порядок описывает развёртывание одного экземпляра программы на одном сервере. Промышленная эксплуатация предполагает вынесение хранилищ данных на отдельные узлы; такая конфигурация задаётся параметрами подключения и не изменяет порядок установки.

5.1. Подготовка среды

1. Подготовить сервер, удовлетворяющий требованиям разделов 4.1 и 4.2.
2. Установить среду контейнеризации Docker Engine и Docker Compose согласно документации их поставщиков.
3. Убедиться, что учётной записи, от имени которой выполняется установка, разрешено управление контейнерами.
4. Обеспечить сетевую доступность внешних систем, перечисленных в разделе 4.3, а при использовании внешних хранилищ данных — доступность серверов баз данных.

5.2. Получение дистрибутива

Дистрибутив поставляется одним из двух способов:

- **исходный текст программы** — архив либо доступ к репозиторию системы контроля версий; образы контейнеров собираются на месте установки средствами, входящими в комплект поставки (файлы сборки размещены в каталоге `docker/`);
- **собранные образы контейнеров** — из реестра образов правообладателя; в этом случае этап сборки не выполняется.

Комплект поставки исходного текста включает: файлы описания сервисов (`compose.yaml` — профиль разработки с режимом отслеживания изменений, отладчиком и тестовой базой данных; промышленный профиль, включающий все компоненты раздела 3.1, поставляется правообладателем отдельным файлом), файлы сборки образов (`docker/`), файл автоматизации операций (`Makefile`), шаблон конфигурации (`.env.example`) и исходный текст серверной и клиентской частей.

5.3. Конфигурирование

Конфигурация экземпляра задаётся файлом переменных окружения. Файл создаётся копированием шаблона:

```
cp .env.example .env
```

Шаблон содержит полный перечень параметров (около 300 параметров) с описанием назначения каждого. Параметры сгруппированы по назначению:

Группа параметров	Содержание
Общие параметры приложения	Наименование, среда исполнения, режим отладки, часовой пояс, языковые настройки
Подключение к базе данных	Тип, адрес, порт, имя базы, учётные данные
Подключение к хранилищу в оперативной памяти	Адрес, порт, пароль, префикс кеша, драйвер кеша
Очереди заданий	Драйвер очередей, параметры обработчика
Интеграция с эмитентом карт	Адреса программных интерфейсов, реквизиты доступа, признак изолированного режима
Интеграция с платёжным шлюзом	Адреса, пути к ключам подписи, идентификаторы проекта, признак изолированного режима
Курсы валют	Источник курса, параметры кеширования, предельное отклонение от официального курса
Мессенджеры и уведомления	Реквизиты ботов, параметры доставки push-уведомлений для мобильных операционных систем
Ограничения и защита	Ограничения частоты выпуска карт, разрешённые источники запросов, параметры проверки одноразовых кодов
Внешние системы аналитики и поддержки	Идентификаторы счётчиков, признаки включения интеграций
Служебные параметры развёртывания	Имя проекта контейнеризации, порты, публикуемые на сервере

Обязательными к заполнению являются: параметры подключения к базе данных, параметры подключения к хранилищу в оперативной памяти, имя проекта контейнеризации и публикуемые порты. Параметры интеграций с внешними системами заполняются в соответствии с составом подключаемых систем.

Изолированный режим. Параметры `VCC_SANDBOX` и `PAYMENT_SANDBOX` в значении `true` (значение по умолчанию) переводят взаимодействие с эмитентом карт и платёжным шлюзом в изолированный режим: обращения к внешним системам не выполняются, ответы формируются программой. Режим предназначен для проверки развёртывания, демонстрации и функционального тестирования без реальных денежных операций. Для промышленной эксплуатации оба параметра устанавливаются в значение `false` и заполняются реквизиты подключения.

Значения ключей, паролей и токенов не входят в состав документации и не подлежат хранению в системе контроля версий. Рекомендуется размещать их в системе управления секретами и передавать в среду исполнения на этапе развёртывания.

5.4. Развёртывание компонентов

Запуск компонентов выполняется средствами Docker Compose с файлом промышленного профиля. Имя файла задаётся параметром `COMPOSE_FILE` в файле конфигурации экземпляра, поэтому все команды ниже приводятся без явного указания файла:

```
docker compose up -d
```

Команда `make up` из файла автоматизации предназначена для среды разработки: она запускает профиль разработки и режим отслеживания изменений исходного текста, удерживает терминал и при эксплуатации не используется.

Состав запускаемых компонентов определяется файлом описания сервисов и соответствует разделу 3.1: серверное приложение, обработчик фоновых заданий, планировщик, сервер событий реального времени, вспомогательные служебные сервисы, а при развёртывании хранилищ в составе комплекта — также сервер базы данных и хранилище в оперативной памяти.

Клиентские приложения собираются в статические файлы и раздаются обратным прокси-сервером. Сборка выполняется в контейнере: сначала устанавливаются зависимости и выполняется сборка, затем результат помещается в образ веб-сервера. Все параметры клиентской части фиксируются на этапе сборки, поэтому их полный набор должен быть задан до её выполнения; изменение параметров клиентской части требует повторной сборки.

5.5. Инициализация данных

После первого запуска компонентов создаётся структура базы данных и заполняются справочные таблицы значениями, необходимыми для работы:

```
docker compose exec app php artisan migrate
docker compose exec app php artisan db:seed
```

Для нового непромышленного экземпляра (демонстрационного или тестового) допускается разовая инициализация одной командой `docker compose exec app php artisan app:init`. Команда пересоздаёт базу данных: повторный запуск удаляет все данные, поэтому её выполняют только один раз, при первой установке. На промышленном экземпляре команда структуры базы данных не изменяет.

5.6. Проверка работоспособности

Признаками успешной установки являются:

1. все компоненты находятся в состоянии «запущен», а компоненты, для которых определена проверка состояния, — в состоянии «работоспособен»:

```
docker compose ps
```

2. серверное приложение отвечает на запросы по опубликованному порту; при включённом параметре генерации спецификации доступно описание программного интерфейса по служебному адресу экземпляра;
3. обработчик фоновых заданий запущен и обрабатывает очереди — отсутствует рост числа необработанных заданий;
4. клиентское приложение открывается в браузере и отображает экран аутентификации;
5. в журналах компонентов отсутствуют сообщения об ошибках подключения к базе данных и хранилищу в оперативной памяти.

Для функциональной проверки в изолированном режиме допускается пройти сценарий выпуска карты (раздел 2.3): при установленных признаках изолированного режима выпуск завершается без обращения к внешним системам и без списания денежных средств.

6. Эксплуатация программного обеспечения

6.1. Запуск и остановка

Операция	Команда
Запуск всех компонентов	<code>docker compose up -d</code>
Остановка всех компонентов	<code>docker compose down</code>
Перезапуск компонента	<code>docker compose restart <компонент></code>
Просмотр состояния компонентов	<code>docker compose ps</code>
Просмотр журнала компонента	<code>docker compose logs -f <компонент></code>
Командная строка внутри компонента	<code>make shell</code>

Остановка серверного приложения не приводит к потере данных: состояние хранится во внешних хранилищах, необработанные фоновые задания сохраняются в очереди и исполняются после возобновления работы обработчика.

6.2. Обработка фоновых заданий

Существенная часть операций выполняется асинхронно: обращения к внешним системам, доставка уведомлений, обработка входящих уведомлений платёжных систем, учётные и аналитические операции. Обработчик фоновых заданий запускается как отдельный компонент и управляет пулом рабочих процессов по очередям.

При эксплуатации следует контролировать:

- число необработанных заданий по каждой очереди — устойчивый рост указывает на недостаточную производительность обработчика либо на недоступность внешней системы;
- число заданий, завершившихся с ошибкой, — они сохраняются отдельно и могут быть перезапущены администратором после устранения причины;
- время нахождения задания в очереди — рост означает задержку в доставке уведомлений и в операциях с картами.

6.3. Регламентные операции

Регламентные операции запускаются компонентом-планировщиком по расписанию, определённому в конфигурации программы. К ним относятся: синхронизация операций по картам с системами эмитентов, обновление курсов валют, рассылка уведомлений о предстоящих списаниях, обслуживание акционных механик, формирование отчётности, очистка устаревших служебных записей.

Планировщик работает автономно и не требует вмешательства администратора. Отдельные операции обслуживания могут быть запущены вручную из командной строки компонента серверного приложения.

6.4. Контроль состояния

Программа предоставляет средства контроля состояния:

- служебный адрес с метриками в формате, пригодном для сбора внешней системой мониторинга; метрики публикуются на отдельном порту и включают длину очередей, число заданий с ошибками, время нахождения задания в очереди и показатели предметной области;
- проверки состояния компонентов средствами среды контейнеризации — состояние отображается командой просмотра состояния компонентов;
- журналирование событий в стандартный вывод компонентов с возможностью передачи во внешнюю систему сбора журналов.

Рекомендуется настроить внешнюю систему мониторинга на сбор публикуемых метрик и оповещение при превышении пороговых значений длины очередей и числа заданий с ошибками.

6.5. Обновление версии

1. Ознакомиться с перечнем изменений устанавливаемой версии.
2. Создать резервную копию базы данных.
3. Получить новую версию дистрибутива (исходный текст либо образы контейнеров).
4. При необходимости внести новые параметры конфигурации, добавленные в шаблоне.
5. Перезапустить компоненты с новыми образами.
6. Выполнить обновление структуры базы данных: `make migrate`.
7. Проверить работоспособность согласно разделу 5.6.

Обновление структуры базы данных выполняется последовательно и фиксирует применённые изменения, что исключает их повторное применение.

6.6. Типовые нештатные ситуации

Признак	Вероятная причина	Действия
Компонент не запускается, в журнале — ошибка подключения к базе данных	Неверные параметры подключения либо недоступность сервера базы данных	Проверить параметры подключения в файле конфигурации и сетевую доступность сервера базы данных
Запросы к программному интерфейсу возвращают ошибку авторизации	Недействительный или истёкший токен доступа; отсутствие требуемого разрешения	Повторить аутентификацию; проверить состав разрешений, выданных токену
Рост числа необработанных заданий	Обработчик фоновых заданий остановлен либо недоступна внешняя система	Проверить состояние обработчика; проверить доступность внешних систем; при необходимости увеличить число рабочих процессов
Операции по картам не отображаются	Сбой синхронизации с системой эмитента	Проверить журнал соответствующих фоновых заданий; убедиться в доступности системы эмитента и действительности реквизитов доступа
Платёж оплачен, но операция не завершена	Не доставлено входящее уведомление платёжной системы либо задание завершилось с ошибкой	Проверить поступление входящего уведомления; при необходимости перезапустить задание, завершившееся с ошибкой
Изменения параметров клиентской части не применяются	Параметры клиентской части фиксируются на этапе сборки	Выполнить повторную сборку клиентского приложения с обновлёнными параметрами

7. Требования к квалификации персонала

Для установки и эксплуатации экземпляра программы требуется системный администратор, обладающий:

- навыками администрирования операционных систем семейства GNU/Linux;
- навыками работы со средствами контейнеризации (Docker, Docker Compose);
- базовыми навыками администрирования систем управления базами данных;
- пониманием принципов работы веб-приложений и протокола HTTP.

Специальная подготовка для работы с программой не требуется. Конечные пользователи работают с программой через клиентское приложение и не нуждаются в специальных знаниях.

Численность персонала, необходимого для сопровождения экземпляра программы, а также состав процессов поддержания жизненного цикла программного обеспечения приводятся в отдельном документе.