Руководство администратора ПО

«Сервис реферальной системы»

1. ВВЕДЕНИЕ

1.1. Область применения

Настоящий документ предназначен для администраторов, обеспечивающих эксплуатацию программного обеспечения «Сервис реферальной системы» (далее — Система).

Система представляет собой серверное приложение, предоставляющее API для управления реферальными маркетинговыми кампаниями. Функциональные возможности включают регистрацию участников реферальных программ, фиксацию выполнения условий, расчёт вознаграждений и формирование статистических данных. Взаимодействие с системой осуществляется посредством JSON-RPC API по протоколу HTTP.

Руководство содержит сведения, необходимые для установки, первичной настройки, администрирования, мониторинга работоспособности и восстановления системы после сбоев.

1.2. Перечень выполняемых функций администратора/оператора

Администратор обеспечивает установку и функционирование Системы в вычислительной среде организации. В обязанности администратора входит подготовка серверного окружения, развертывание базы данных, конфигурация параметров подключения, настройка ключей доступа к API, контроль доступности сервисов и анализ журналов работы.

В процессе эксплуатации администратор контролирует корректность работы контейнеров, состояние соединений с базой данных и доступность API-методов. При необходимости администратор выполняет обновление версии Системы, восстановление данных из резервных копий и изменение конфигурационных параметров.

Также администратор отвечает за обеспечение безопасности — ограничение доступа к API, хранение ключей доступа и настройку сетевых правил.

1.3. Уровень подготовки администратора/оператора

Администратор должен обладать навыками работы с операционными системами семейства Linux, уметь использовать контейнерную среду Docker и понимать принципы функционирования сетевых сервисов HTTP. Необходимы базовые знания СУБД PostgreSQL, включая создание баз данных, проверку соединений и выполнение резервного копирования.

Для эксплуатации системы достаточно одного администратора, прошедшего подготовку по использованию программного обеспечения и ознакомленного с настоящим руководством.

1.4. Перечень документации

В состав документации, с которой необходимо ознакомиться администратору Системы входят:

- Функциональная спецификация системы
- Техническое задание на разработку системы
- Настоящее руководство администратора

2. УСТАНОВКА СИСТЕМЫ

2.1. Системные требования

Система предназначена для работы на сервере под управлением Linux. Минимальная конфигурация вычислительных ресурсов предполагает наличие двух логических ядер процессора, не менее 2 ГБ оперативной памяти и 10 ГБ дискового пространства. Рекомендуется использование 4 ГБ оперативной памяти для стабильной работы при высокой нагрузке.

В программной части требуется установленный Docker версии не ниже 20.10 и СУБД PostgreSQL версии 16 или выше. Дополнительно может использоваться система мониторинга Prometheus и средство визуализации Grafana.

Сервис реализован на языке программирования Go и не требует установки дополнительных библиотек в операционной системе, кроме средств запуска контейнеров.

2.2. Порядок установки

1. Смонтируйте диск с дистрибутивом в папку /mnt
2. Скопируйте из дистрибутива исходники из папки /mnt в папку /opt/inviteflow
3. Отредактируйте файл docker-compose.yml в соответствии с разделом 3.2 данного документа
4. Смените текущую папку на /opt/inviteflow и выполните команду: docker compose up -d --build
5. Проверьте работоспособность системы.

3. НАСТРОЙКА СИСТЕМЫ

3.1. Общие сведения

В данном документе приводятся примеры настройки Системы с использованием среды Docker Compose. Настройка операционной системы, СУБД, а также возможная настройка использования систем оркестрации, находятся вне компетенции этого документа и не будут тут описаны.

3.2. Конфигурируемые параметры

Настройка Системы осуществляется посредством переменных окружения контейнера приложения. Значения параметров определяют сетевые адреса сервиса, параметры подключения к базе данных, режимы журналирования и доступ к API.

Сетевой адрес, на котором HTTP-сервер принимает входящие соединения, задаётся переменной HOST. При отсутствии значения используется порт 80 внутри контейнера.

Порт публикации служебных метрик задаётся переменной METRICS_PORT. Данный порт используется системой мониторинга для получения показателей состояния сервиса и не предназначен для пользовательских запросов.

Доступ к приватным методам API защищён ключом, передаваемым в заголовке Authorization. Значение ключа определяется переменной ACCESS_KEY. При её отсутствии запуск сервиса невозможен.

Параметры подключения к базе данных PostgreSQL задаются следующими переменными:

- DB_HOST — адрес сервера базы данных;
- DB_PORT — порт подключения;
- DB_USER — имя пользователя;
- DB_PASSWORD — пароль пользователя;
- DB_NAME — имя базы данных.

Для управления производительностью соединений используются параметры пула:

- DB_MAX_OPEN_CONNECTIONS — максимальное количество одновременно открытых соединений с БД;
- DB_MAX_IDLE_CONNECTIONS — максимальное количество простаивающих соединений;
- DB_CONN_MAX_TIME — максимальное время жизни соединения, после которого оно пересоздаётся.

Сервис поддерживает дополнительные параметры диагностики. Переменная RPC_LOGGING_ENABLED включает журналирование вызовов JSON-RPC методов, что используется при анализе работы системы. Переменная RPC_METRICS_ENABLED включает сбор статистики по вызовам API, которая затем публикуется через endpoint метрик.

Изменение значений перечисленных параметров требует перезапуска контейнера приложения.

4. ОПИСАНИЕ API

4.1. Общие сведения

Система предоставляет JSON-RPC API версии 2.0 по протоколу HTTP.

Публичные методы доступны по адресу /public и не требуют авторизации. Приватные методы доступны по адресу / и требуют передачи ключа доступа в заголовке:

Authorization: Bearer <ACCESS_KEY>

Заголовок запроса должен содержать:

Content-Type: application/json

Во всех запросах JSON-RPC используются поля jsonrpc, method, id, params. Тип params зависит от метода: для одних методов он является объектом, для других — примитивом (например, число id).

4.2. Методы API

4.2.1. ping

Проверка доступности сервиса.

Endpoint: /public

Авторизация: не требуется

Запрос:

```
{  "jsonrpc": "2.0",  "method": "ping",  "id": 1 }
```

Ответ:

```
{  "jsonrpc": "2.0",  "result": "pong",  "id": 1 }
```

4.2.2. Кампании

campaign.create

создание кампании

Endpoint: /

Авторизация: требуется

Создаёт новую маркетинговую кампанию.

Параметры:

- name — название кампании, отображаемое в системе (обязательный);
- description — произвольное текстовое описание кампании;
- status — начальный статус кампании (active или inactive, по умолчанию inactive);
- starts_at — дата начала действия кампании;
- ends_at — дата завершения кампании.

Результат: идентификатор созданной кампании.

campaign.update

изменение кампании

Endpoint: /

Авторизация: требуется

Изменяет параметры существующей кампании. Передаются только изменяемые поля.

Параметры:

- id — идентификатор кампании (обязательный);
- name — новое название;
- description — новое описание;
- status — новый статус;
- starts_at — новая дата начала;
- ends_at — новая дата окончания.

Результат: идентификатор кампании.

campaign.delete

удаление кампании

Endpoint: /

Авторизация: требуется

Удаляет кампанию из системы.

Параметры:

- id — идентификатор кампании (обязательный).

Результат: булевое значение.

campaign.setStatus

изменение статуса кампании

Endpoint: /

Авторизация: требуется

Изменяет только статус кампании.

Параметры:

- id — идентификатор кампании (обязательный);
- status — новый статус (active или inactive, обязательный).

Результат: булевое значение.

campaign.list

получение списка кампаний

Endpoint: /

Авторизация: требуется

Возвращает список кампаний с возможностью фильтрации и постраничного вывода.

Параметры:

- status — фильтр по статусу;
- limit — максимальное количество записей;
- offset — смещение выборки.

Результат: массив объектов кампаний.

4.2.3. Правила начисления вознаграждений

campaign.createRewardRule

создание правила начисления

Endpoint: /

Авторизация: требуется

Создаёт правило начисления бонуса участникам кампании.

Параметры:

- campaign_id — идентификатор кампании (обязательный);
- beneficiary_role — получатель награды (referrer — пригласивший, referee — приглашённый);
- condition_type — тип условия (registration, first_payment, custom);
- reward_type — тип награды (fixed или percent);
- reward_value — величина награды (должна быть больше 0);
- currency — валюта начисления;
- is_active — признак активности правила.

Результат: идентификатор правила.

campaign.updateRewardRule

изменение правила начисления

Endpoint: /

Авторизация: требуется

Изменяет параметры правила начисления.

Параметры:

- id — идентификатор правила (обязательный);
- beneficiary_role — получатель награды;
- condition_type — тип условия;
- reward_type — тип награды;
- reward_value — величина награды;
- currency — валюта;
- is_active — признак активности.

Результат: идентификатор правила.

campaign.deleteRewardRule

удаление правила начисления

Endpoint: /

Авторизация: требуется

Параметры:

- id — идентификатор правила (обязательный).

Результат: булево значение.

campaign.listRewardRules

список правил кампании

Endpoint: /

Авторизация: требуется

Параметры:

- campaign_id — идентификатор кампании (обязательный);
- only_active — возвращать только активные правила.

Результат: массив правил начисления.

4.2.4. Реферальные идентификаторы

referral.createIdentifier

создание реферального кода

Endpoint: /

Авторизация: требуется

Создаёт уникальный код приглашения. Если код не передан, он генерируется автоматически.

Параметры:

- campaign_id — идентификатор кампании (обязательный);
- referrer_id — идентификатор пригласившего пользователя (обязательный);
- code — желаемый код приглашения;
- link — готовая ссылка приглашения;
- base_url — базовый адрес для автоматической генерации ссылки.

Результат: идентификатор созданного идентификатора.

referral.deactivateIdentifier

деактивация кода

Endpoint: /

Авторизация: требуется

Параметры:

- id — идентификатор кода (обязательный).

Результат: булево значение.

referral.listIdentifiers

список кодов

Endpoint: /

Авторизация: требуется

Параметры:

- campaign_id — фильтр по кампании;
- referrer_id — фильтр по пригласившему;
- is_active — фильтр по активности;
- limit — ограничение количества;
- offset — смещение.

Результат: массив идентификаторов.

4.2.5. Рефералы

referral.register

регистрация реферала

Endpoint: /

Авторизация: требуется

Фиксирует факт приглашения пользователя.

Параметры:

- campaign_id — идентификатор кампании (обязательный);
- referrer_id — пригласивший пользователь (обязательный);
- referee_id — приглашённый пользователь (обязательный).

Результат: идентификатор записи реферала.

referral.setStatus

изменение статуса реферала

Параметры:

- id — идентификатор реферала (обязательный);
- status — статус (pending, qualified, paid, invalid).

Результат: булево значение.

referral.get

получение рефералов

Параметры:

- id — идентификатор;
- campaign_id — фильтр по кампании;
- referrer_id — фильтр по пригласившему;
- referee_id — фильтр по приглашённому;
- status — фильтр по статусу;
- limit — ограничение;
- offset — смещение.

Результат: массив рефералов.

referral.addCondition

фиксация выполнения условия

Endpoint: /

Авторизация: требуется

Регистрирует выполнение пользователем условия кампании.

Параметры:

- referral_id — идентификатор реферала (обязательный);
- condition_type — тип условия (registration, first_payment, custom);
- metadata — дополнительные данные события (JSON).

Результат: идентификатор записи условия.

4.2.6. Вознаграждения

reward.calculate

расчёт награды

Endpoint: /

Авторизация: требуется

Выполняет начисление вознаграждения по правилам кампании.

Параметры:

- referral_id — идентификатор реферала (обязательный);
- condition_type — выполненное условие;
- base_amount — базовая сумма для процентных наград;
- currency — валюта начисления.

Результат: массив идентификаторов созданных транзакций.

reward.list

список начислений

Endpoint: /

Авторизация: требуется

Параметры:

- campaign_id — фильтр по кампании;
- user_id — фильтр по получателю;
- status — статус транзакции;
- limit — ограничение;
- offset — смещение.

Результат: массив транзакций вознаграждений.

4.2.7. Статистика

stats.getCampaignStats

статистика кампании

Endpoint: /

Авторизация: требуется

Параметры:

- campaign_id — идентификатор кампании (обязательный);
- from — начало периода;
- to — конец периода.

Результат: агрегированные показатели кампании.

stats.getUserInfo

статистика пользователя

Endpoint: /

Авторизация: требуется

Параметры:

- user_id — идентификатор пользователя (обязательный);
- role — роль (referrer, referee, all).

Результат: статистика участия пользователя.

stats.export

экспорт данных

Endpoint: /

Авторизация: требуется

Endpoint: /

Авторизация: требуется

Параметры:

- campaign_id — идентификатор кампании (обязательный);
- from — начало периода;
- to — конец периода;
- limit — ограничение;
- offset — смещение.

Результат: массив строк экспорта.

5. МОНИТОРИНГ

Система предоставляет HTTP-endpoint метрик, предназначенный для подключения систем мониторинга. Метрики содержат информацию о количестве вызовов API и времени обработки запросов.

Проверка доступности осуществляется посредством вызова публичного метода проверки работоспособности. Отсутствие корректного ответа свидетельствует о недоступности сервиса либо базы данных.

6. ЛОГИРОВАНИЕ

Сервис выводит структурированные журналы в стандартный поток вывода контейнера. Журналы включают информацию о вызовах методов, ошибках обработки и проблемах подключения к базе данных.

Просмотр журналов выполняется средствами Docker либо системой централизованного сбора логов.

7. РЕЗЕРВНОЕ КОПИРОВАНИЕ

Резервное копирование выполняется средствами PostgreSQL. Рекомендуется регулярно сохранять дамп базы данных и хранить его вне сервера приложения. Восстановление производится загрузкой дампа в чистую базу данных и последующим запуском сервиса.