

CrossPay MVP Documentation

Оглавление

1. Введение	1
2. Архитектура системы	1
2.1. Обзор	1
2.2. Диаграмма взаимодействия	2
3. API Reference	3
3.1. Инициация перевода	3
4. Установка и настройка	4
4.1. Предварительные требования	4
4.2. Конфигурация	4
5. Деплой	4
5.1. Локальная разработка	4
6. Тестирование	5
6.1. Сценарии тестирования	5
7. Часто задаваемые вопросы	5
7.1. Какие комиссии?	5
7.2. Сколько времени занимает перевод?	5
8. Поддержка	5
9. История изменений	5
10. 0.1.0 (2024-01-01)	6

1. Введение

CrossPay — это решение для международных переводов заработной платы из России в другие страны через Telegram Bot.

- **Статус:** MVP
- **Версия:** 0.1.0
- **Целевая аудитория:** Экспаты, получающие зарплату в RUB

2. Архитектура системы

2.1. Обзор

Основные компоненты системы:

- **Telegram Bot** — интерфейс взаимодействия с пользователем

- **Backend Server** — ядро системы (Python + Flask)
- **PostgreSQL** — база данных
- **ЮKassa API** — прием платежей в RUB
- **BaaS Provider** — выпуск карт и управление счетами

2.2. Диаграмма взаимодействия

```
@startuml diagram
```

```
actor "Клиент" as Client
participant "Telegram Bot" as Bot
participant "Backend Server" as Backend
participant "База данных" as DB
participant "ЮKassa API" as YooMoney
participant "BaaS Provider\n(A2PAY/Aftab)" as BaaS
actor "Администратор" as Admin
```

```
== Клиент инициирует перевод ==
```

```
Client -> Bot: /transfer <сумма_в_rub>
Bot -> Backend: POST /transfer {user_id, amount_rub}
Backend -> Backend: Генерирует ID транзакции
Backend -> DB: Сохраняет транзакцию (PENDING)
Backend -> Backend: Рассчитывает курс и сумму в EUR
Backend -> Bot: {transaction_id, amount_eur, payment_details}

Bot -> Client: "Переведите <amount_rub> RUB\nРеквизиты: ...\nКомментарий:
<transaction_id>"
```

```
== Клиент оплачивает ==
```

```
Client -> YooMoney: Оплата с комментарием <transaction_id>
YooMoney -> Backend: Webhook о платеже (автоматически)
Backend -> DB: Обновляет статус транзакции (PAID)
```

```
== Админ подтверждает получение ==
```

```
Admin -> Backend: Просматривает список транзакций (ручная проверка)
Backend -> DB: Получает транзакции со статусом PAID
Backend -> Admin: Список транзакций для обработки
```

```
Admin -> Backend: Подтверждает транзакцию <transaction_id>
Backend -> BaaS: POST /cards/create {user_data}
BaaS -> Backend: {card_id, card_details}
Backend -> BaaS: POST /accounts/transfer {card_id, amount_eur}
Backend -> DB: Обновляет статус транзакции (COMPLETED)
```

```
== Уведомление клиента ==
```

```
Backend -> Bot: Уведомление о готовности
Bot -> Client: "Ваша карта готова! \nНомер: XXXX-XXXX-XXXX-1234\nСрок: MM/YY\nCVV: 123"
```

== Альтернативный поток: Автоматизация через n8n ==

```
group Автоматическая обработка [Опционально]
  participant "n8n" as N8N
  participant "Email Bank" as Email

  Email -> N8N: Новое письмо от банка
  N8N -> N8N: Парсит transaction_id из письма
  N8N -> Backend: POST /payment/confirm {transaction_id}
  Backend -> DB: Обновляет статус транзакции (PAID)
  Backend -> N8N: 200 OK
end

@enduml
```

3. API Reference

3.1. Инициация перевода

```
POST /transfer HTTP/1.1
Content-Type: application/json
```

```
{
  "user_id": 123456789,
  "amount_rub": 50000,
  "currency": "EUR"
}
```

Response:

```
{
  "transaction_id": "txn_001",
  "amount_eur": 476.19,
  "payment_details": {
    "account": "40702810000000000001",
    "comment": "txn_001"
  },
  "exchange_rate": 105.0
}
```

4. Установка и настройка

4.1. Предварительные требования

- Python 3.9+
- PostgreSQL 12+
- Telegram Bot Token
- ЮKassa API keys
- BaaS Provider account

4.2. Конфигурация

Create `.env` file:

```
# Database
DATABASE_URL=postgresql://user:pass@localhost/crosspay

# Telegram
TELEGRAM_BOT_TOKEN=your_bot_token

# Payment Providers
YOO_MONEY_SHOP_ID=your_shop_id
YOO_MONEY_SECRET_KEY=your_secret_key

# BaaS
BAAS_API_KEY=your_baas_key
BAAS_BASE_URL=https://api.baas-provider.com

# Exchange Rates
DEFAULT_EXCHANGE_RATE=105.0
COMMISSION_RATE=0.02
```

5. Деплой

5.1. Локальная разработка

```
# Клонирование репозитория
git clone https://github.com/yourname/crosspay.git
cd crosspay

# Установка зависимостей
pip install -r requirements.txt

# Настройка базы данных
```

```
python scripts/init_db.py
```

```
# Запуск приложения  
python app.py
```

6. Тестирование

6.1. Сценарии тестирования

Успешный перевод:

- Пользователь отправляет /transfer 50000
- Система генерирует transaction_id
- Пользователь оплачивает через ЮKassy
- Админ подтверждает платеж
- Система создает карту через BaaS API
- Пользователь получает данные карты

7. Часто задаваемые вопросы

7.1. Какие комиссии?

- Конвертация RUB/EUR: 2%
- Выпуск виртуальной карты: бесплатно
- Выпуск физической карты: 5 EUR

7.2. Сколько времени занимает перевод?

- Обработка платежа: 1-24 часа
- Выпуск карты: мгновенно

8. Поддержка

- **Telegram канал:** https://t.me/crosspay_support
- **Issues:** <https://github.com/yourname/crosspay/issues>
- **Email:** support@crosspay.com

9. История изменений

10. 0.1.0 (2024-01-01)

- Первый рабочий MVP
- Базовая интеграция с ЮKassa и A2PAY
- Telegram Bot интерфейс