

## API эндпоинты

### 1. Аутентификация и регистрация

#### 1.1 Регистрация пользователя (Sign Up)

Метод: POST

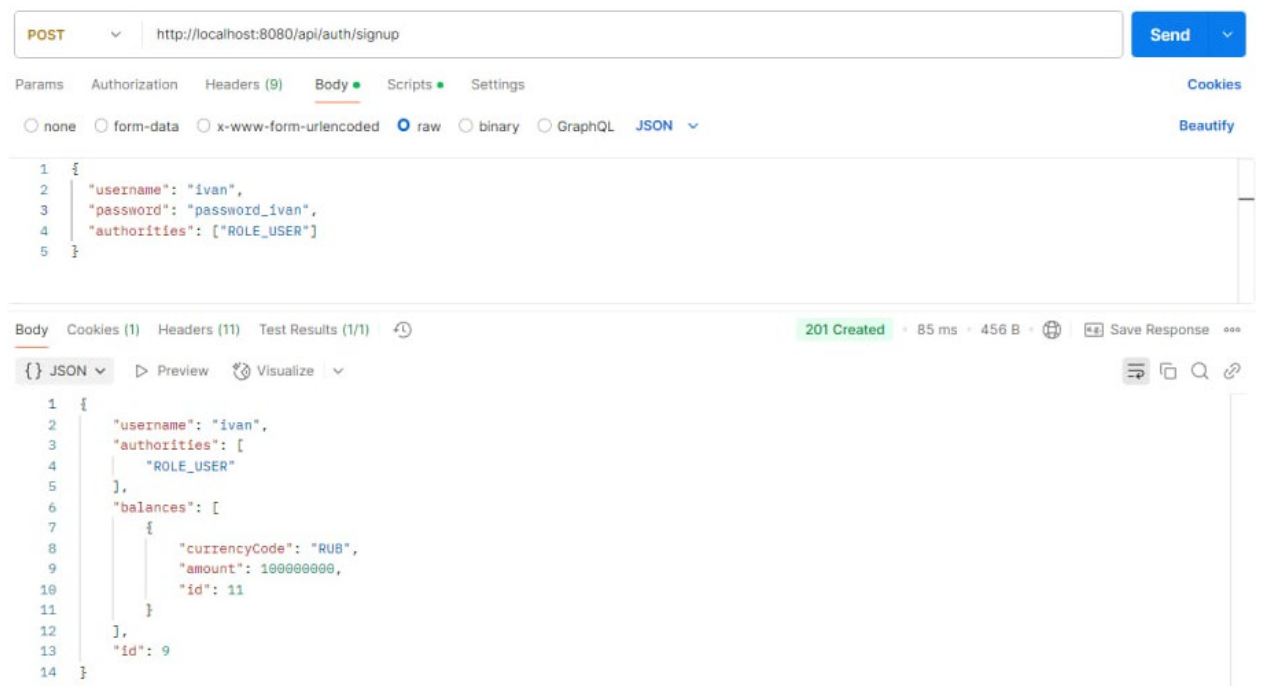
URL: /api/auth/signup

Тело запроса (JSON):

```
{
  "username": "ivan",
  "password": "password_ivan",
  "authorities": ["ROLE_USER"]
}
```

Назначение: Создаёт нового пользователя с указанными ролями и начальным балансом (например, RUB).

Результат:



#### 1.2 Аутентификация (Sign In)

Метод: POST

URL: /api/auth/signin

Тело запроса (JSON):

```
{
  "username": "bob",
```

```
"password": "password_bob"
}
```

Назначение: Проверяет учётные данные и возвращает JWT-токен для последующих запросов.

Результат:

POST http://localhost:8080/api/auth/signin

Params Authorization Headers (8) Body Scripts Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "username": "bob",
3   "password": "password_bob"
4 }
5
```

Body Cookies Headers (11) Test Results (1/1) 200 OK · 79 ms · 463 B Save Response

Raw Preview Visualize

```
1 eyJhbGciOiJIUzI1NiJ9.eyJzdWUiOiJib2IiLCJpYXQiOiJlY3NDE5NDA4NjUsImV4cCI6MTc0MTk0MTc2NX0.g0SskIaxR-Af9gAvs6UWC-yQaQ1nATS--tedvI1vtPw
```

POST http://localhost:8080/api/auth/signin

Params Authorization Headers (8) Body Scripts Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "username": "admin",
3   "password": "admin"
4 }
```

Body Cookies Headers (11) Test Results (1/1) 200 OK · 479 ms · 466 B Save Response

Raw Preview Visualize

```
1 eyJhbGciOiJIUzI1NiJ9.eyJzdWUiOiJhZG1pb2IiLCJpYXQiOiJlY3NDE5NDA4NjUsImV4cCI6MTc0MTk0MTc2NX0.g0SskIaxR-Af9gAvs6UWC-yQaQ1nATS--tedvI1vtPw
```

## 2. Работа с данными пользователя

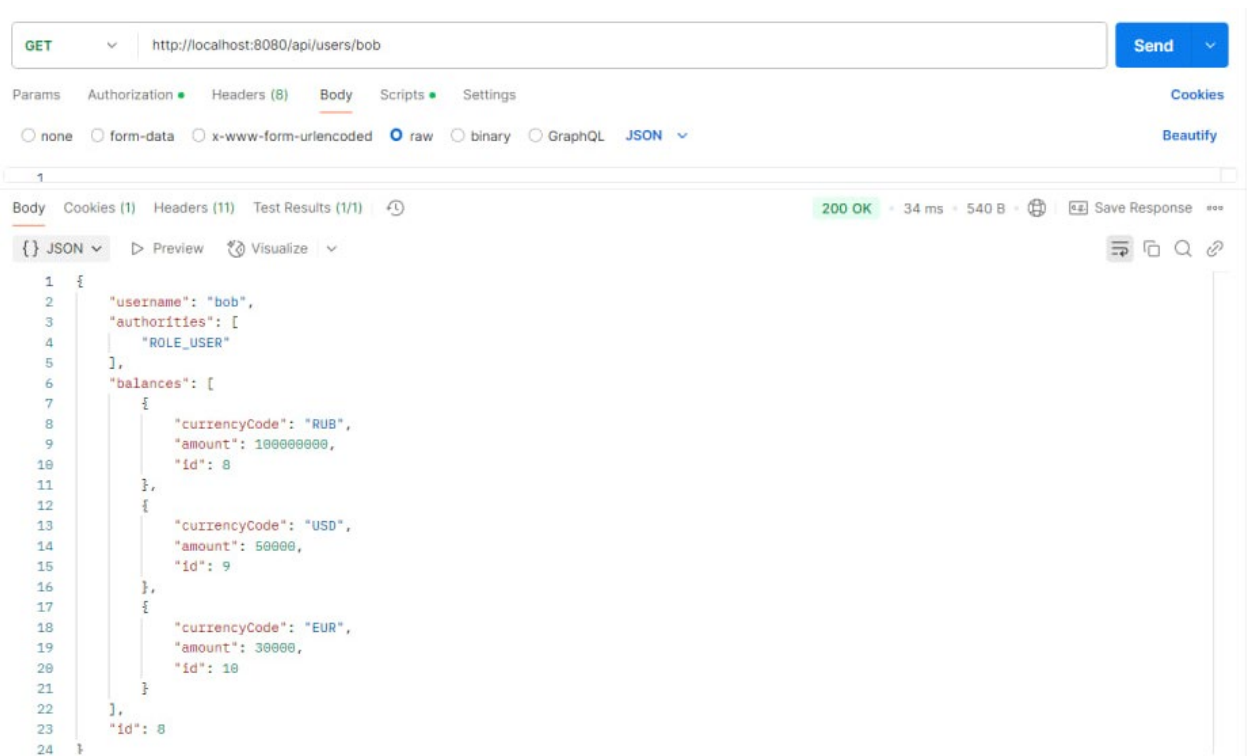
### 2.1 Получение данных пользователя

Метод: GET

URL: /api/users/{username}

Назначение: Возвращает информацию о пользователе (имя, список ролей, балансы и пр.).

Результат:



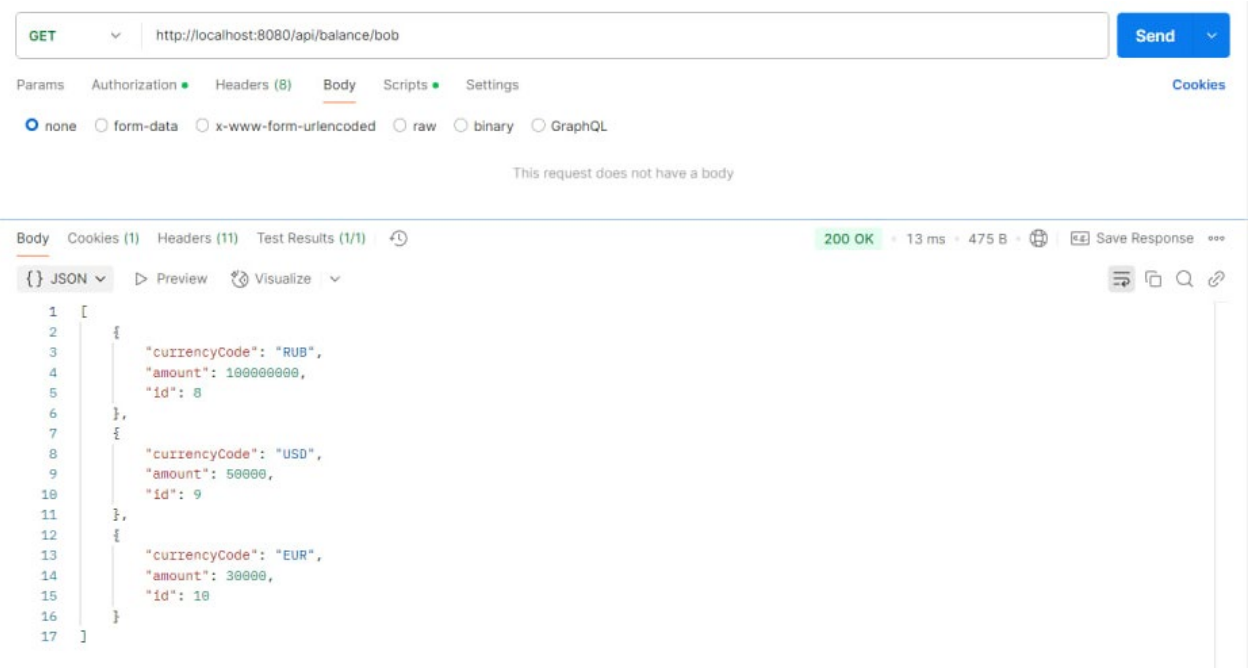
## 2.2 Получение балансов пользователя

Метод: GET

URL: /api/balance/{username}

Назначение: Возвращает список балансов пользователя по разным валютам.

Результат:



## 2.3 Обновление (пополнение) балансов пользователя (только для ADMIN)

Метод: POST

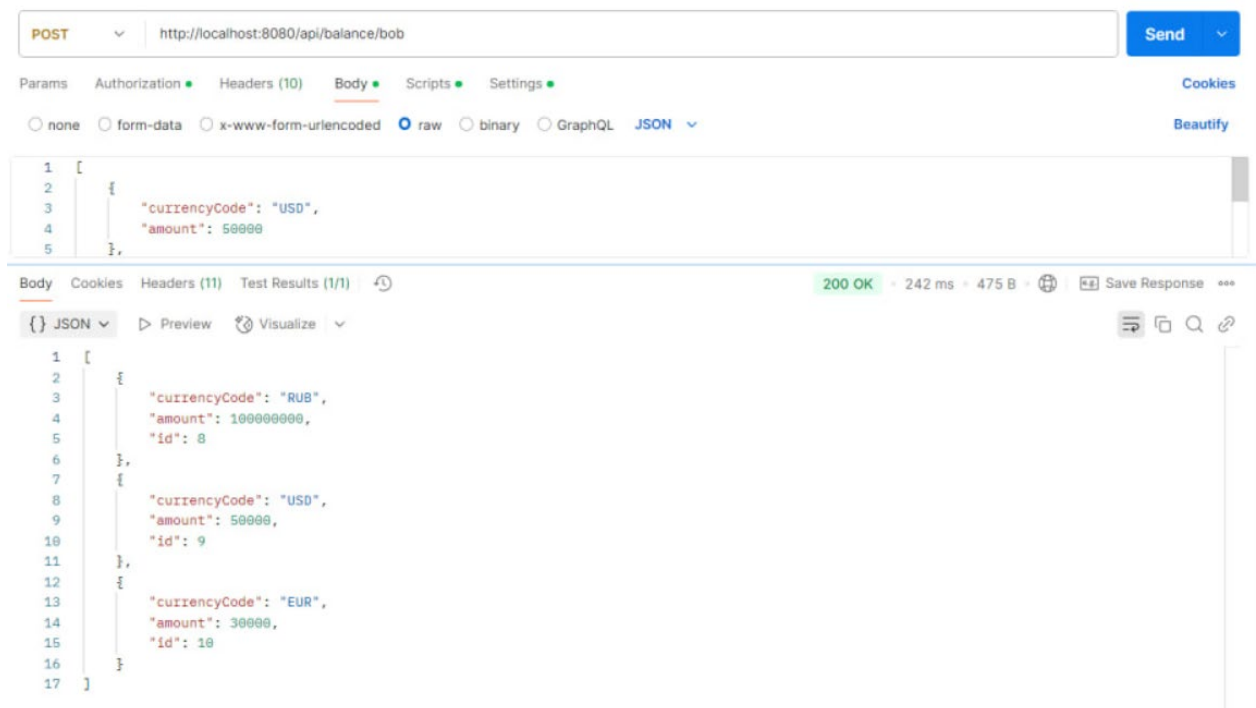
URL: /api/balance/{username}

Тело запроса (JSON):

```
[
  {
    "currencyCode": "USD",
    "amount": 50000
  },
  {
    "currencyCode": "EUR",
    "amount": 30000
  }
]
```

Назначение: Администратор может пополнить или создать балансы для пользователя. Если валюта отсутствует – создаётся новый баланс, иначе сумма добавляется к существующему значению.

Результат:



### 3. Балансы терминала (Exchanger Balance)

Для работы с балансами терминала (отдельной сущностью) создан отдельный контроллер.

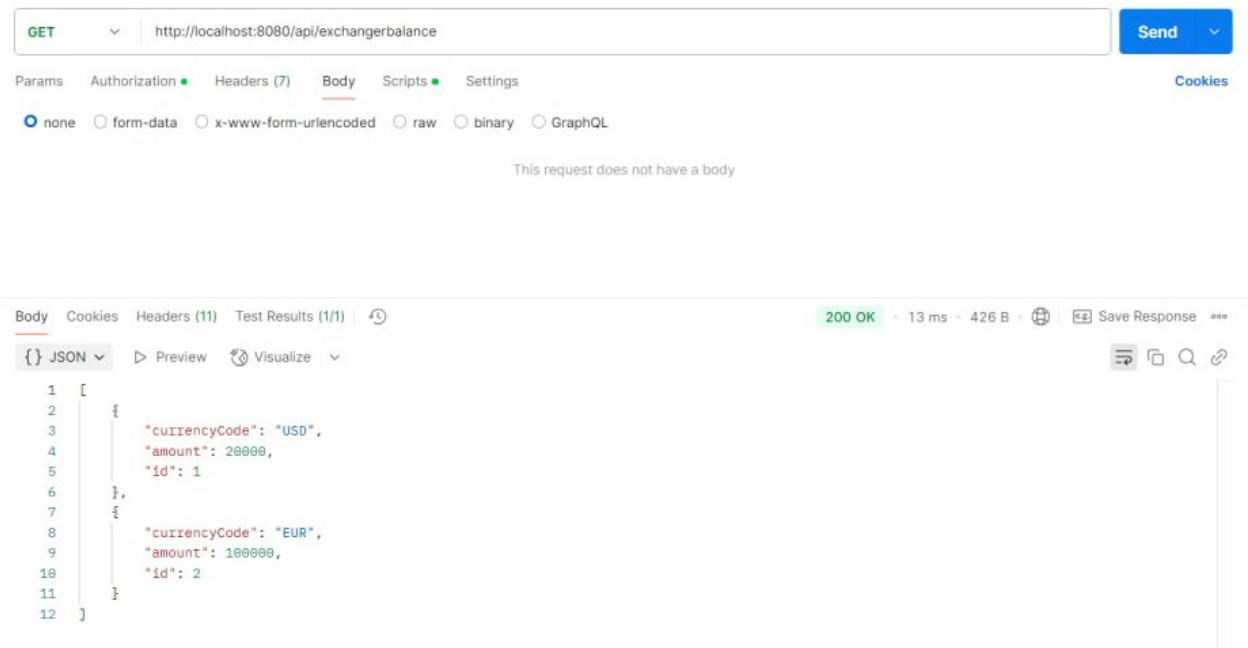
#### 3.1 Получение балансов терминала

Метод: GET

URL: `/api/exchangerbalance`

Назначение: Возвращает список всех балансов терминала. Запрос может выполняться любым аутентифицированным пользователем (например, bob).

Результат:



### 3.2 Обновление балансов терминала (только для ADMIN)

Метод: POST

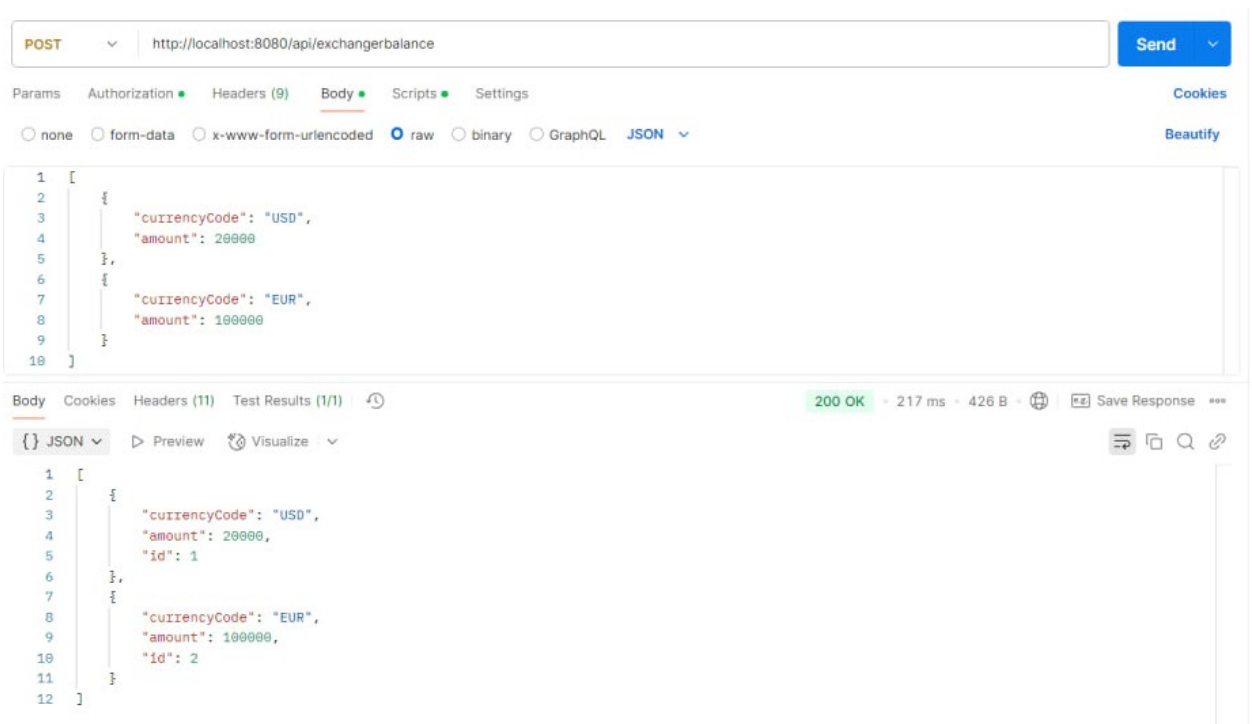
URL: /api/exchangerbalance

Тело запроса (JSON):

```
[
  {
    "currencyCode": "USD",
    "amount": 20000
  },
  {
    "currencyCode": "EUR",
    "amount": 100000
  }
]
```

Назначение: Позволяет администратору пополнить балансы терминала. Если баланс по указанной валюте уже существует, то сумма увеличивается, иначе создаётся новая запись.

Результат:



## 4 Работа с валютными курсами (Rates)

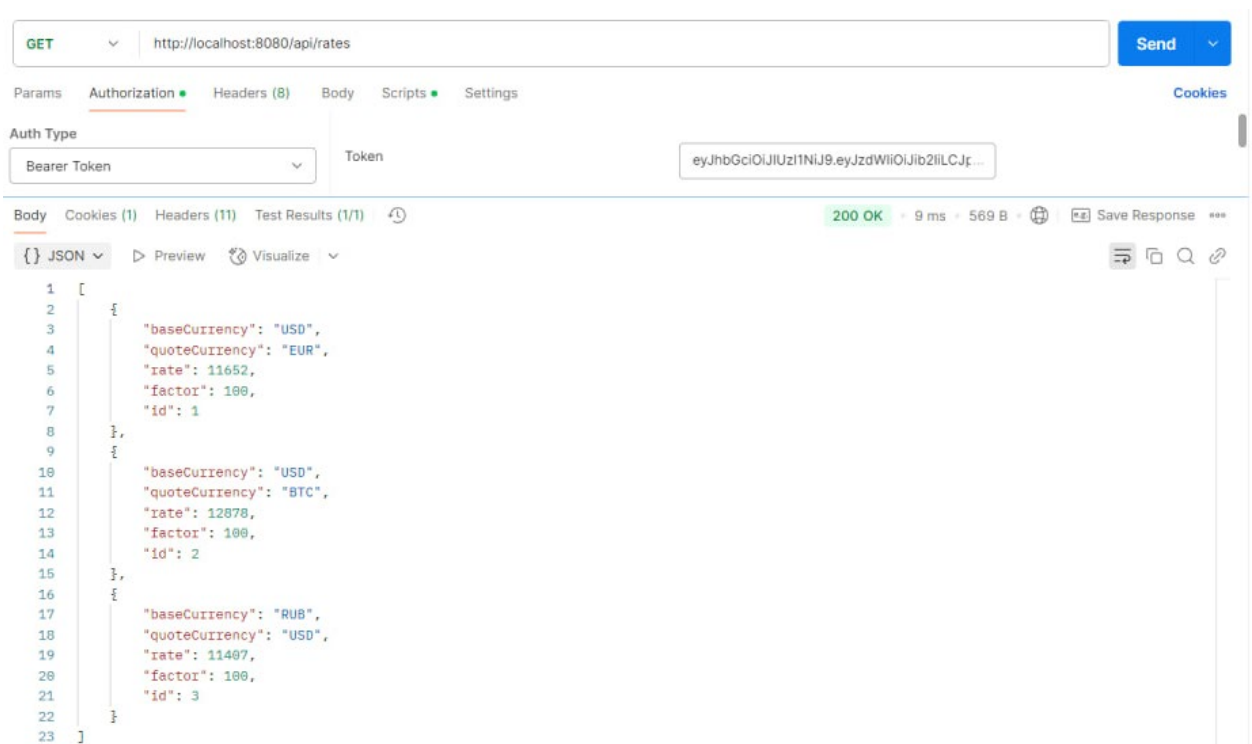
### 4.1 Просмотр валютных пар и курсов

Метод: GET

URL: /api/rates

Назначение: Позволяет получить список валютных пар с текущими курсами обмена.

Результат:



## 4.2 Добавление новой валютной пары (только для ADMIN)

Метод: POST

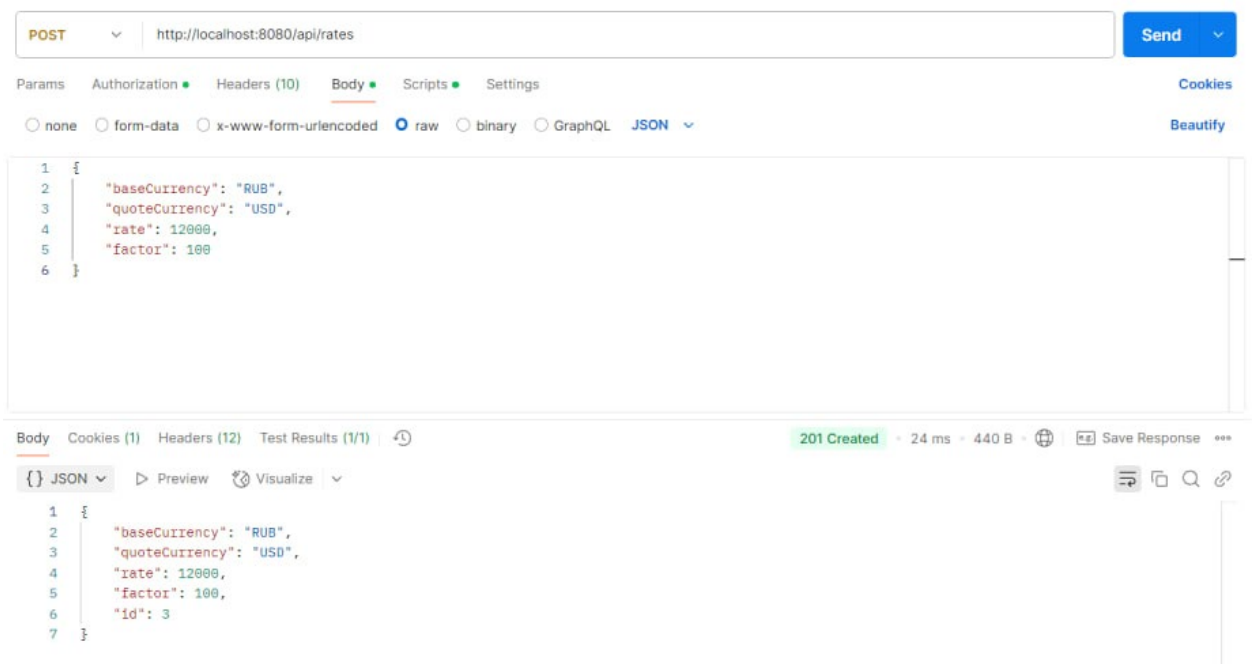
URL: /api/rates

Тело запроса (JSON):

```
{
  "baseCurrency": "RUB",
  "quoteCurrency": "USD",
  "rate": 9000,
  "factor": 100
}
```

Назначение: Создаёт новую валютную пару (например, RUB/USD) и задаёт обменный курс с коэффициентом.

Результат:



## 4.3 Обновление валютных курсов (только для ADMIN)

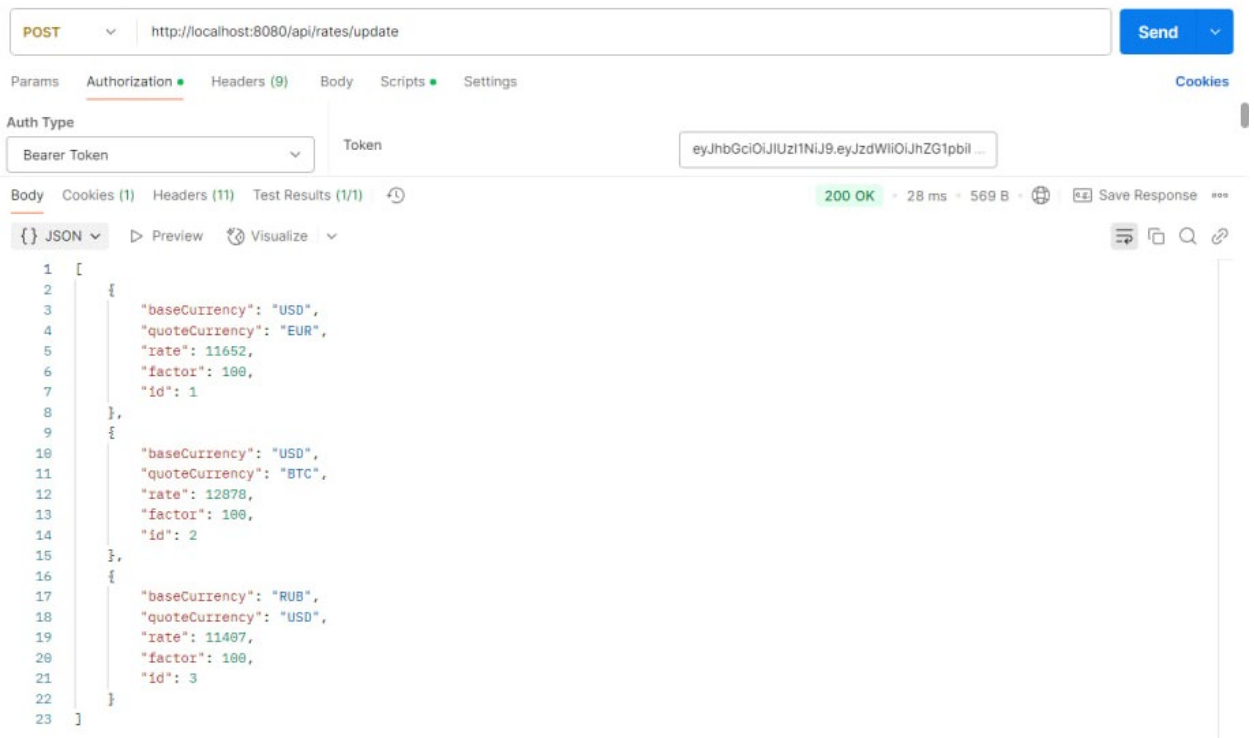
Метод: POST

URL: /api/rates/update

Тело запроса: отсутствует или пустое

Назначение: Пересчитывает и возвращает обновлённые курсы для всех валютных пар.

Результат:



## 5. Операции обмена валют (Transactions)

### 5.1 Проведение обмена валют

Метод: POST

URL: /api/transactions/exchange

Тело запроса (JSON):

```
{
  "username": "bob",
  "fromCurrency": "RUB",
  "toCurrency": "USD",
  "amount": 1000
}
```

Назначение: Пользователь (например, bob) может провести операцию обмена, в результате которой происходит списание суммы в базовой валюте и зачисление суммы в валюте назначения по определённому курсу. При этом в контроллере может использоваться проверка, чтобы имя пользователя в токене совпадало с полем "username".



Результат:

POST http://localhost:8080/api/transactions/exchange

Params Authorization Headers (9) Body Scripts Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "username": "bob",
3   "fromCurrency": "RUB",
4   "toCurrency": "USD",
5   "amount": 1000
6 }
7
```

Body Cookies Headers (11) Test Results (1/1)

200 OK • 72 ms • 465 B

Save Response

JSON Preview Visualize

```
1 {
2   "id": 1,
3   "user": "bob",
4   "currencyPair": "RUB/USD",
5   "rate": 11407,
6   "fromDelta": -1000,
7   "toDelta": 0,
8   "timestamp": "2025-03-15T16:32:16.8528969"
9 }
```

## 5.2 Получение истории транзакций пользователя

Метод: GET

URL: /api/transactions/user/{username}

Назначение: Возвращает список всех проведённых транзакций для указанного пользователя.

Результат:

GET http://localhost:8080/api/transactions/user/bob

Params Authorization Headers (7) Body Scripts Settings

none form-data x-www-form-urlencoded raw binary GraphQL

This request does not have a body

Body Cookies Headers (11) Test Results (1/1)

200 OK • 23 ms • 596 B

Save Response

JSON Preview Visualize

```
1 [
2   {
3     "id": 1,
4     "user": "bob",
5     "currencyPair": "RUB/USD",
6     "rate": 11407,
7     "fromDelta": -1000,
8     "toDelta": 0,
9     "timestamp": "2025-03-15T16:32:16.852897"
10  },
11  {
12    "id": 2,
13    "user": "bob",
14    "currencyPair": "USD/EUR",
15    "rate": 11375,
16    "fromDelta": -100,
17    "toDelta": 0,
18    "timestamp": "2025-03-15T16:33:05.791811"
19  }
20 ]
```