

Money Transfer Application

The application is available to download in GitHub -> <https://github.com/marif88/revolut-moneyTransfer>

Once it is started it can be accessed in <http://localhost:8180>.

There are 3 APIs available to test this application.

- GET -> /api/account
 - This will list all the accounts stored in the memory
 - Sample response

```
{
  "accounts": [
    {
      "accountNo": 1001,
      "accountName": "Arif",
      "amount": "1599.50"
    },
    {
      "accountNo": 1002,
      "accountName": "Alex",
      "amount": "2500.00"
    },
    {
      "accountNo": 1003,
      "accountName": "Mohan Lal",
      "amount": "7644.00"
    },
    {
      "accountNo": 1004,
      "accountName": "Clay Loyd",
      "amount": "5000.00"
    }
  ]
}
```

- POST -> /api/account
 - This is to register the account in the memory, which accepts account name and the initial amount to be registered. The account number will be auto generated in the system and will returned as the response. For simplicity the account number is a four digit number starts with 1001
 - Sample payload it accepts
{"accountName":"Alex Mathew","amount":1599.50}
 - Response - it will returns the account number created.
- PATCH -> /api/account
 - This the API for money transfer, which accepts senders account number and receivers account number and the amount to be transferred. Once a successful operation the sender's account details will be returned with updated balance as response. The amount should be 10 or more than that.
 - Sample payload is look like
{"senderAccountNo":1003,"receiverAccountNo":1002,"amount":98.98}
 - Response – Senders account details with updated balance.


```
{
  "senderAccount": {
    "accountNo": 1003,
    "accountName": "Arif",
    "amount": "1700.00"
  }
}
```

Postman Results

1. POST

The image shows the Postman interface for a POST request. The request is sent to `http://localhost:8180/api/account` with a JSON body: `{"accountName": "Clay Loyd", "amount": 5000.00}`. The response is a 201 Created status with a JSON body: `{"accountNo": "1004"}`.

Request Details:

- Method: POST
- URL: `http://localhost:8180/api/account`
- Body Type: JSON (application/json)
- Body:

```
1 { "accountName": "Clay Loyd", "amount": 5000.00 }
```

Response Details:

- Status: 201 Created
- Time: 12 ms
- Size: 141 B
- Body:

```
1 {  
2   "accountNo": "1004"  
3 }
```

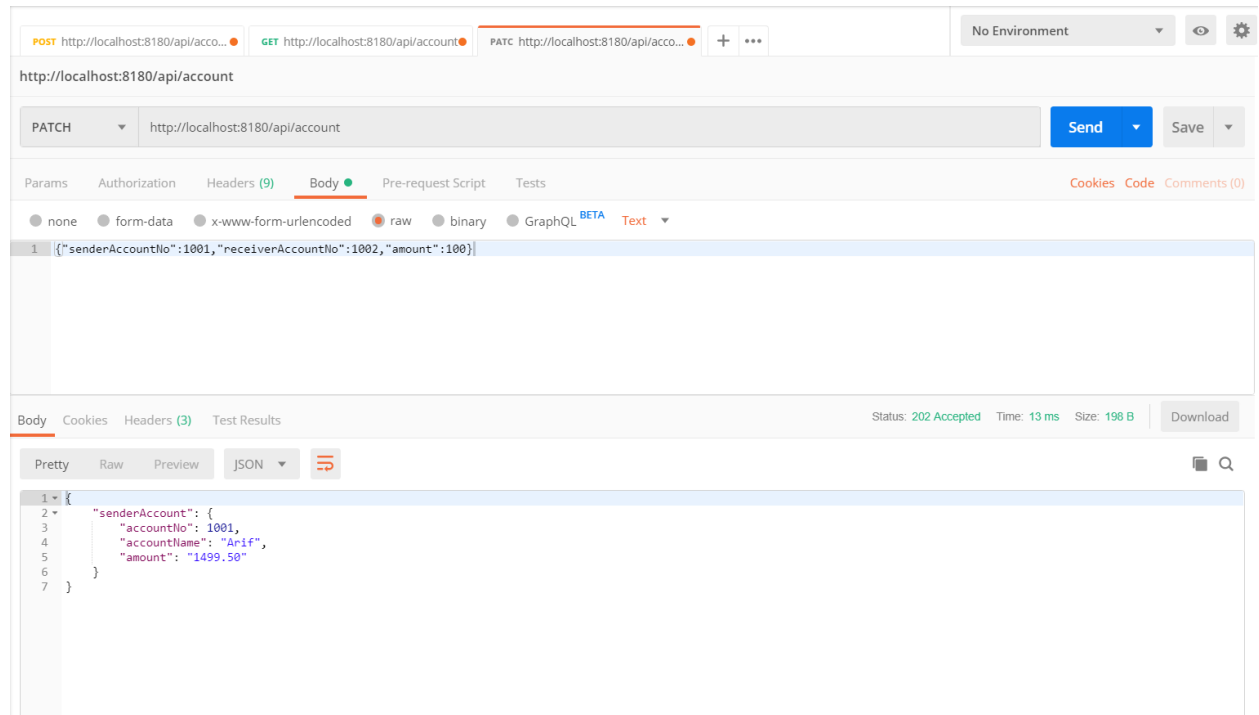
2. GET

The screenshot shows a REST client interface with a top bar containing three tabs: `POST http://localhost:8180/api/acco...`, `GET http://localhost:8180/api/account` (selected), and `PATCH http://localhost:8180/api/acco...`. On the right of the top bar is a dropdown menu set to `No Environment` and icons for eye and settings. Below the top bar, the `GET` method is selected, and the URL `http://localhost:8180/api/account` is entered. To the right of the URL are `Send` and `Save` buttons. Below this is a tab bar with `Pretty`, `Raw`, and `Preview` options, and a `JSON` dropdown menu. The main area displays the JSON response in a pretty-printed format with line numbers 1 through 24 on the left. The response is an object with a key `accounts` pointing to an array of four account objects.

```
1 {  
2   "accounts": [  
3     {  
4       "accountNo": 1001,  
5       "accountName": "Arif",  
6       "amount": "1599.50"  
7     },  
8     {  
9       "accountNo": 1002,  
10      "accountName": "Alex",  
11      "amount": "2500.00"  
12    },  
13    {  
14      "accountNo": 1003,  
15      "accountName": "Mohan Lal",  
16      "amount": "7544.00"  
17    },  
18    {  
19      "accountNo": 1004,  
20      "accountName": "Clay Loyd",  
21      "amount": "5000.00"  
22    }  
23  ]  
24 }
```

3. PATCH

3.1. Transferred 100 from account 1001 to 1002



The image shows a Postman interface for a PATCH request to `http://localhost:8180/api/account`. The request body is a JSON object: `{ "senderAccountNo": 1001, "receiverAccountNo": 1002, "amount": 100 }`. The response status is `202 Accepted` with a time of `13 ms` and size of `198 B`. The response body is a JSON object: `{ "senderAccount": { "accountNo": 1001, "accountName": "Arif", "amount": "1499.50" } }`.

POST `http://localhost:8180/api/acco...` GET `http://localhost:8180/api/account` PATCH `http://localhost:8180/api/acco...` + ... No Environment

`http://localhost:8180/api/account`

PATCH `http://localhost:8180/api/account` Send Save

Params Authorization Headers (9) Body Pre-request Script Tests Cookies Code Comments (0)

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

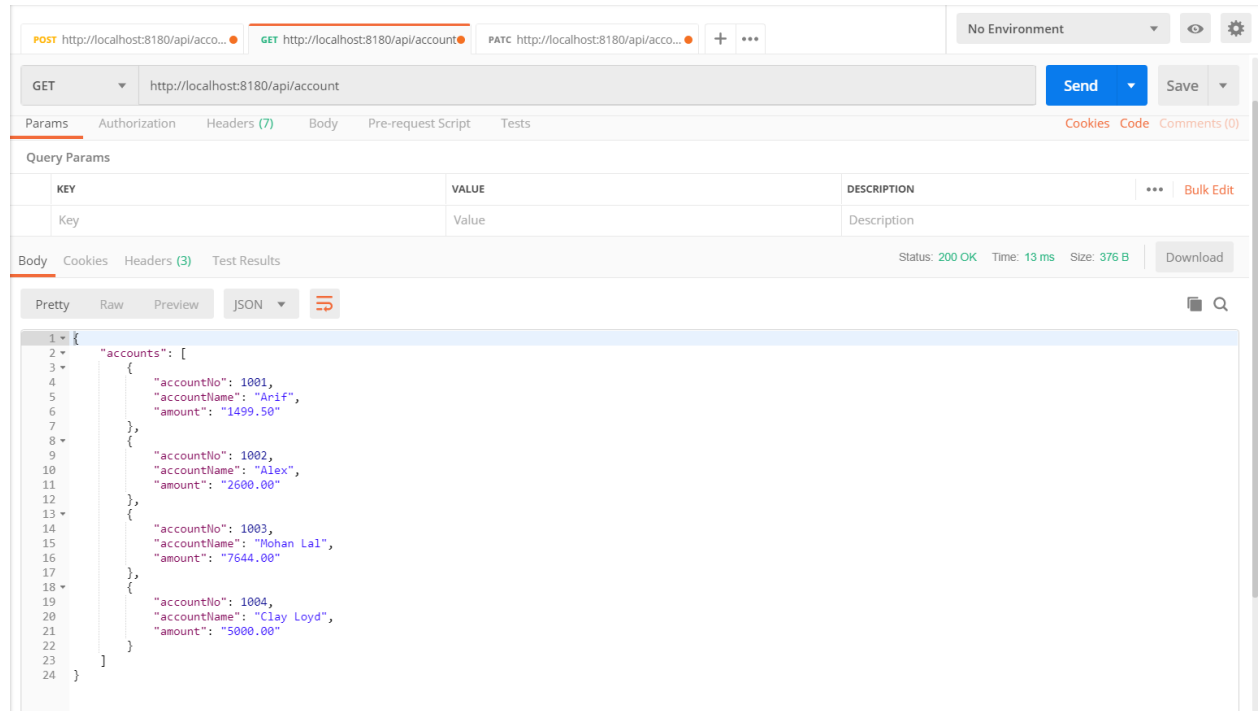
1 `{ "senderAccountNo": 1001, "receiverAccountNo": 1002, "amount": 100 }`

Body Cookies Headers (3) Test Results Status: 202 Accepted Time: 13 ms Size: 198 B Download

Pretty Raw Preview JSON

```
1 {
2   "senderAccount": {
3     "accountNo": 1001,
4     "accountName": "Arif",
5     "amount": "1499.50"
6   }
7 }
```

Updated Account details



The image shows a Postman interface for a GET request to `http://localhost:8180/api/account`. The response status is `200 OK` with a time of `13 ms` and size of `376 B`. The response body is a JSON object: `{ "accounts": [{ "accountNo": 1001, "accountName": "Arif", "amount": "1499.50" }, { "accountNo": 1002, "accountName": "Alex", "amount": "2600.00" }, { "accountNo": 1003, "accountName": "Mohan Lal", "amount": "7644.00" }, { "accountNo": 1004, "accountName": "Clay Lloyd", "amount": "5000.00" }] }`.

POST `http://localhost:8180/api/acco...` GET `http://localhost:8180/api/account` PATCH `http://localhost:8180/api/acco...` + ... No Environment

GET `http://localhost:8180/api/account` Send Save

Params Authorization Headers (7) Body Pre-request Script Tests Cookies Code Comments (0)

Query Params

| KEY | VALUE | DESCRIPTION |
|-----|-------|-------------|
| Key | Value | Description |

Body Cookies Headers (3) Test Results Status: 200 OK Time: 13 ms Size: 376 B Download

Pretty Raw Preview JSON

```
1 {
2   "accounts": [
3     {
4       "accountNo": 1001,
5       "accountName": "Arif",
6       "amount": "1499.50"
7     },
8     {
9       "accountNo": 1002,
10      "accountName": "Alex",
11      "amount": "2600.00"
12     },
13     {
14       "accountNo": 1003,
15       "accountName": "Mohan Lal",
16       "amount": "7644.00"
17     },
18     {
19       "accountNo": 1004,
20       "accountName": "Clay Lloyd",
21       "amount": "5000.00"
22     }
23   ]
24 }
```

3.2. Transferred more amount than the balance

The screenshot shows a REST client interface with a PATCH request to `http://localhost:8180/api/account`. The request body is a JSON object: `{"senderAccountNo":1002,"receiverAccountNo":1003,"amount":3000}`. The response status is `400 Bad Request` with a time of `11 ms` and size of `202 B`. The response body is a JSON object: `{"code": 400, "message": "Insufficient Balance - Amount more required : 400.00"}`.

3.3. Minimum transfer amount check – Minimum amount set 10

The screenshot shows a REST client interface with a PATCH request to `http://localhost:8180/api/account`. The request body is a JSON object: `{"senderAccountNo":1003,"receiverAccountNo":1004,"amount":9.99}`. The response status is `400 Bad Request` with a time of `14 ms` and size of `189 B`. The response body is a JSON object: `{"code": 400, "message": "Transfer amount should be 10 or more"}`.

3.4. Senders account number not found

The screenshot shows a REST client interface with a PATCH request to `http://localhost:8180/api/account`. The request body is a JSON object: `{"senderAccountNo":1009,"receiverAccountNo":1004,"amount":1000}`. The response status is `404 Not Found` with a time of `12 ms` and size of `184 B`. The response body is a JSON object: `{"code": 404, "message": "Sender's account is not found !!!"}`.

3.5. Receivers account number not found

The screenshot shows a REST client interface with a PATCH request to `http://localhost:8180/api/account`. The request body is a JSON object: `{"senderAccountNo":1001,"receiverAccountNo":1009,"amount":1000}`. The response status is `404 Not Found` with a time of `21 ms` and size of `183 B`. The response body is a JSON object: `{"code": 404, "message": "Receiver's account is not found !!!"}`.

3.6. Senders account number and receivers account are same

The screenshot displays a REST client interface with the following details:

- Request:**
 - Method: PATCH
 - URL: http://localhost:8180/api/account
 - Body: `{"senderAccountNo":1001,"receiverAccountNo":1001,"amount":1000}`
- Response:**
 - Status: 400 Bad Request
 - Time: 12 ms
 - Size: 201 B
 - Body (JSON):

```
1 {
2   "code": 400,
3   "message": "Sender account and receiver account can't be same !"
4 }
```