

## Money Transfer Application

The application is available to download in GitHub ->

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 screenshot shows the Postman interface for a POST request to `http://localhost:8180/api/account`. The request body is a JSON object: `{"accountName": "Clay Loyd", "amount": 5000.00}`. The response status is 201 Created, with a time of 12 ms and a size of 141 B. The response body is a JSON object: `{ "accountNo": "1004" }`.

Request Details:

- Method: POST
- URL: `http://localhost:8180/api/account`
- Body: `{"accountName": "Clay Loyd", "amount": 5000.00}`

Response Details:

- Status: 201 Created
- Time: 12 ms
- Size: 141 B
- Body: `{ "accountNo": "1004" }`

## 2. GET

The screenshot shows a REST client interface with a GET request to `http://localhost:8180/api/account`. The response is a JSON array of account objects, displayed in a 'Pretty' format. The response body is as follows:

```
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 }
```

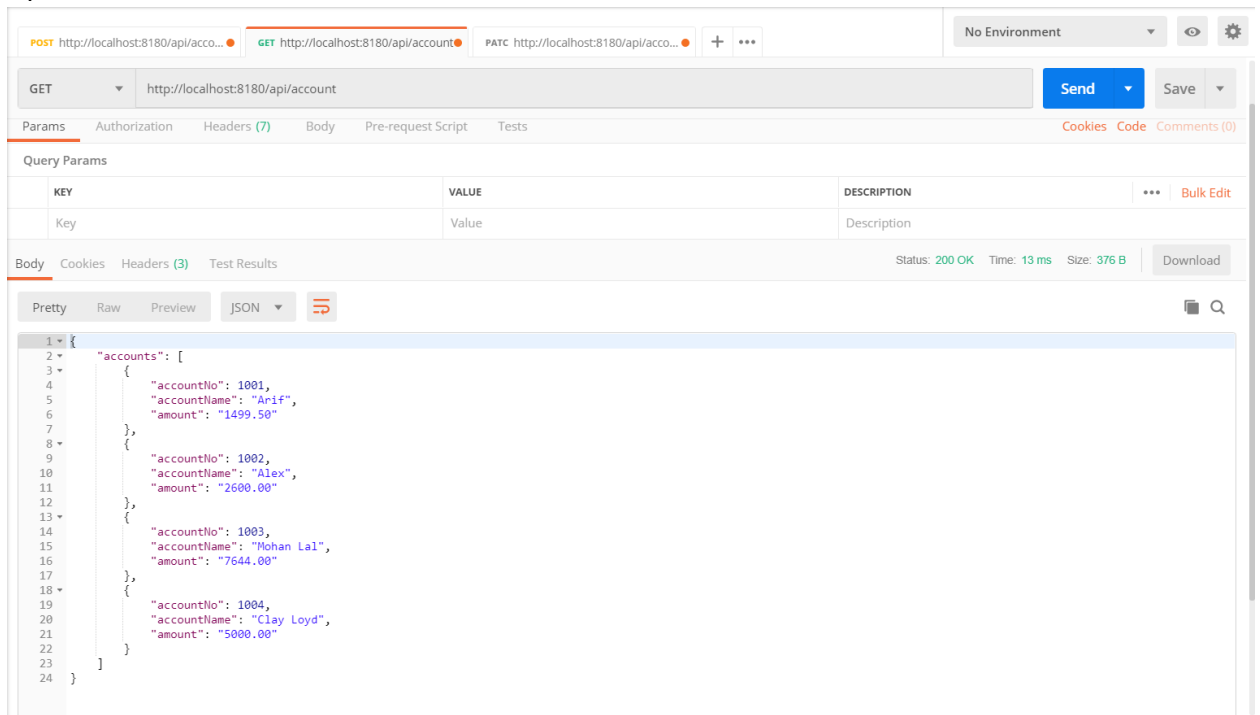
## 3. PATCH

### 3.1. Transferred 100 from account 1001 to 1002

The 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": 1002, "amount": 100 }`. The response is a JSON object showing the updated account information for account 1001, displayed in a 'Pretty' format. The response body is as follows:

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

## Updated Account details



POST http://localhost:8180/api/acco... GET http://localhost:8180/api/account PATC 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

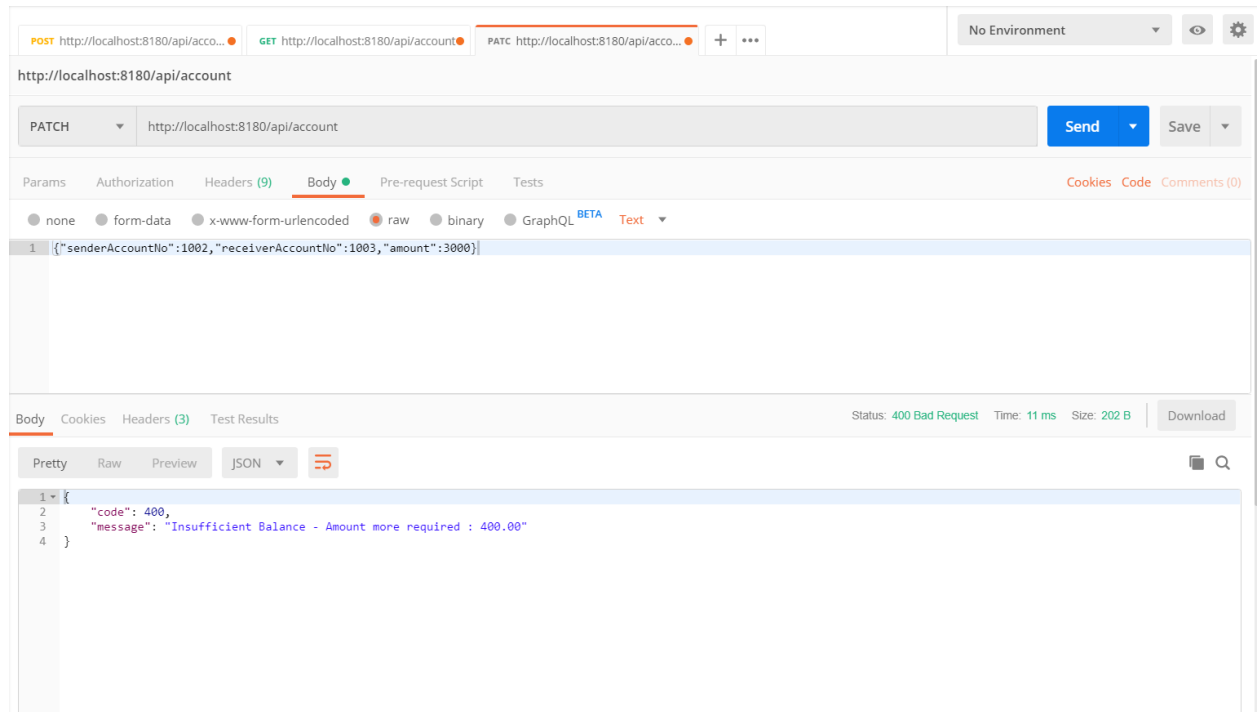
KEY	VALUE	DESCRIPTION
Key	Value	Description

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 Loyd",
21       "amount": "5000.00"
22     }
23   ]
24 }
```

### 3.2. Transferred more amount than the balance



POST http://localhost:8180/api/acco... GET http://localhost:8180/api/account PATC http://localhost:8180/api/acco... No Environment

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":1002,"receiverAccountNo":1003,"amount":3000}]

Body Cookies Headers (3) Test Results Status: 400 Bad Request Time: 11 ms Size: 202 B Download

Pretty Raw Preview JSON

```
1 {
2   "code": 400,
3   "message": "Insufficient Balance - Amount more required : 400.00"
4 }
```

### 3.3. Minimum transfer amount check – Minimum amount set 10

The screenshot shows a Postman interface for 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" }`.

```
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": 1003, "receiverAccountNo": 1004, "amount": 9.99 }
```

Body Cookies Headers (3) Test Results Status: 400 Bad Request Time: 14 ms Size: 189 B Download

Pretty Raw Preview JSON

```
1 {
2   "code": 400,
3   "message": "Transfer amount should be 10 or more"
4 }
```

### 3.4. Senders account number not found

The screenshot shows a Postman interface for 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 is not found !!!" }`.

```
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": 1009, "receiverAccountNo": 1004, "amount": 1000 }
```

Body Cookies Headers (3) Test Results Status: 404 Not Found Time: 12 ms Size: 184 B Download

Pretty Raw Preview JSON

```
1 {
2   "code": 404,
3   "message": "Sender's account is is not found !!!"
4 }
```

### 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 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":1001,"amount":1000}`. The response status is `400 Bad Request` with a time of `12 ms` and size of `201 B`. The response body is a JSON object: `{ "code": 400, "message": "Sender account and receiver account can't be same !" }`.