

Transaction-Details API

Sample Transactions

1) Post Transaction

URL: <http://localhost:8080/postTransaction>

Body

```
{
  "transactionDate": "20181212",
  "postedDate": "20181210",
  "transactionAmount": "33.45",
  "transactionReferenceId": "501",
  "transactionDescription": "Purchase",
  "debitCreditCode": "CR",
  "cardExpirationDate": "1234",
  "currencyCode": "840",
  "pointOfSaleCardPresenceCode": "12",
  "pointOfSaleCardPresenceDescription": "Card holder present at time of sale",
  "pointOfSaleCardUsageCode": "Chip",
  "cardReferenceId": "CF94ndQMq/bbjG2+zbgvhVTDikltNCZL2ptcPzDhWpo=",
  "merchantDetails": {
    "merchantName": "Abc Food Place",
    "category": "Dining",
    "categoryCode": 15,
    "storeNumber": 123456,
    "merchantAddress": {
      "addressLine1": "Address1",
      "addressLine2": "Address2",
      "addressLine3": "Address3",
      "city": "Chicago",
      "state": "IL",
      "postalCode": "60005",
      "country": "US"
    }
  }
}
```

Successful Response Code: 200 OK

Screenshot: Post Transaction - 200 OK

The screenshot displays a REST client interface with a POST request to `http://localhost:8080/postTransaction`. The request body is a JSON object representing a transaction. The response status is `200 OK` with a response time of `445 ms` and a size of `75 B`.

Request Details:

- Method: POST
- URL: `http://localhost:8080/postTransaction`
- Params: (empty)
- Body Type: JSON (application/json)

Request Body (JSON):

```
{
  "transactionDate": "20181212",
  "postedDate": "20181210",
  "transactionAmount": "33.45",
  "transactionDescription": "Purchase",
  "debitCreditCode": "CR",
  "cardExpirationDate": "1234",
  "currencyCode": "840",
  "pointOfSaleCardPresenceCode": "12",
  "pointOfSaleCardPresenceDescription": "Card holder present at time of sale",
  "pointOfSaleCardUsageCode": "Chip",
  "cardReferenceId": "CF94ndQMq/bbjG2+zbgyhVTDikltNCZL2ptcPzDhWpo=",
  "merchantDetails": {
    "merchantName": "Abc Food Place",
    "category": "Dining",
    "categoryCode": 15,
    "storeNumber": 123456,
    "merchantAddress": {
      "addressLine1": "Address1"
    }
  }
}
```

Response Details:

- Status: 200 OK
- Time: 445 ms
- Size: 75 B

The response body is currently empty, showing only a line number 1 in the Pretty view.

Get Transactions

URL : <http://localhost:8080/accounts/<encrpyted card number>/transactions>

<http://localhost:8080/accounts/CF94ndQMq%2FbbjG2%2BzbgvhVTDikltNCZL2ptcPzDhWpo%3D/transactions>

The screenshot shows a web browser window with a REST client interface. The top bar displays the environment as "No Environment". The request is a GET to `http://localhost:8080/accounts/CF94ndQMq%2FbbjG2%2BzbgvhVTDikltNCZL2ptcPzDhWpo%3D/transactions`. The response is a JSON array of three transaction objects, displayed in "Pretty" format.

```
1 [
2   {
3     "cardReferenceId": "CF94ndQMq/bbjG2+zbgvhVTDikltNCZL2ptcPzDhWpo=",
4     "merchantName": "Cheap car rentals",
5     "transactionAmount": "78.21",
6     "transactionDate": "20181217",
7     "transactionReferenceId": "X8MP18eI09q5e8jrR5TZU17Sw9s1I4bPn8S8AAbtAr8=",
8     "debitCreditCode": "CR"
9   },
10  {
11    "cardReferenceId": "CF94ndQMq/bbjG2+zbgvhVTDikltNCZL2ptcPzDhWpo=",
12    "merchantName": "Good Movie Place",
13    "transactionAmount": "12.45",
14    "transactionDate": "20181215",
15    "transactionReferenceId": "xejMLAXm6XQLiW/zfNuqHsbt2XKq+Ea081J/ecKgPUs=",
16    "debitCreditCode": "CR"
17  },
18  {
19    "cardReferenceId": "CF94ndQMq/bbjG2+zbgvhVTDikltNCZL2ptcPzDhWpo=",
20    "merchantName": "Abc Food Place",
21    "transactionAmount": "33.45",
22    "transactionDate": "20181212",
23    "transactionReferenceId": "HBPY60JgPwXmAREJ9pavabrplNOFl/KvP0kX76VS0IQ=",
24    "debitCreditCode": "CR"
25  }
26 ]
```

- Transactions returned are in descending order with latest transactions on top.
- Card Reference Id and Transaction Reference Id returned will be encrypted. Client will have to decrypt them with the key provided to get more details about a particular transaction.
- Transactions can be filtered based on amount as below. Only transactions which are greater than the specified amount will be returned.

The screenshot shows a REST client interface with a 'No Environment' dropdown and a search icon. The request is a GET to `http://localhost:8080/accounts/CF94ndQMq%2FbbjG2%2BzbgvhVTDikltNCZL2ptcPzDhWpo%3D/transactions?amount=45`. The response is a JSON object with the following fields:

```
1 {
2   "cardReferenceId": "CF94ndQMq/bbjG2+zbgvhVTDikltNCZL2ptcPzDhWpo=",
3   "merchantName": "Cheap car rentals",
4   "transactionAmount": "78.21",
5   "transactionDate": "20181217",
6   "transactionReferenceId": "X8MP18eI09q5e8jrR5TZU17Sw9sLI4bPn8S8AAbtAr8=",
7   "debitCreditCode": "CR"
8 }
9
10 ]
```

Get Transaction

To get more details of a particular transaction

URL :

<http://localhost:8080/accounts/<encrypted account reference id>/transactions/<encrypted transaction reference id>>

<http://localhost:8080/accounts/CF94ndQMq%2FbbjG2%2BzbgvhVTDikltNCZL2ptcPzDhWpo%3D/transactions/xejMLAXm6XQLiW%2FzfNuqHsbt2XKq%2BEaO8lJ%2FecKgPUs%3D>

The screenshot shows a web browser interface with a REST client. The URL bar shows the endpoint: `http://localhost:8080/accounts/CF94ndQMq%2FbbjG2%2BzbgvhVTDikltNCZL2ptcPzDhWpo%3D/transactions/xejMLAXm6XQLiW%2FzfNuqHsbt2XKq%2BEaO8lJ%2FecKgPUs%3D`. The method is set to GET. The response is displayed in a 'Pretty' view, showing a JSON object with the following structure:

```
1 {
2   "transactionDate": "20181215",
3   "postedDate": "20181218",
4   "transactionAmount": "12.45",
5   "transactionReferenceId": "xejMLAXm6XQLiW/zfNuqHsbt2XKq+EaO8lJ/ecKgPUs=",
6   "transactionDescription": "Purchase",
7   "debitCreditCode": "CR",
8   "cardExpirationDate": "1234",
9   "currencyCode": "840",
10  "pointOfSaleCardPresenceCode": "12",
11  "pointOfSaleCardPresenceDescription": "Card holder present at time of sale",
12  "pointOfSaleCardUsageCode": "Chip",
13  "cardReferenceId": "CF94ndQMq/bbjG2+zbgvhVTDikltNCZL2ptcPzDhWpo=",
14  "merchantDetails": {
15    "merchantName": "Good Movie Place",
16    "category": "Entertainment",
17    "categoryCode": "4",
18    "storeNumber": "342303",
19    "merchantAddress": {
20      "addressLine1": "Address1",
21      "addressLine2": "Address2",
22      "addressLine3": "Address3",
23      "city": "Chicago",
24      "state": "IL",
25      "postalCode": "60005",
26      "country": "US"
27    }
28  }
29 }
```

Sample Error Scenarios

- POST Transaction - When the Country passed in address is CA and zip code format is not A1A 1A1, validation fails.

The screenshot shows a REST client interface with a POST request to `http://localhost:8080/postTransaction`. The request body is a JSON object:

```
21  {
22    "addressLine3": "Address3",
23    "city": "Toronto",
24    "state": "ON",
25    "postalCode": "60005",
26    "country": "CA"
27  }
28 }
```

The response status is **400 Bad Request** with a time of 55 ms and size of 287 B. The response body is a JSON object:

```
1 {
2   "timestamp": "2018-09-21T14:23:25.318+0000",
3   "status": 400,
4   "error": "Bad Request",
5   "message": "Bad Request",
6   "path": "/postTransaction"
7 }
```

- POST Transaction – Individual Field level validation. In the below screenshot the transaction date passed was invalid format.

The screenshot shows a REST client interface with a POST request to `http://localhost:8080/postTransaction`. The request body is a JSON object representing a transaction. The response is a 400 Bad Request, indicating a validation error.

Request Body:

```
1 {
2   "transactionDate": "201812",
3   "postedDate": "20181220",
4   "transactionAmount": "78.21",
5   "transactionDescription": "Purchase",
6   "debitCreditCode": "CR",
7   "cardExpirationDate": "1234",
8   "currencyCode": "840",
9   "pointOfSaleCardPresenceCode": "12",
10  "pointOfSaleCardPresenceDescription": "Card holder present at time of sale"
}
```

Response Body:

```
1 {
2   "status": "BAD_REQUEST",
3   "message": "Validation failed for argument at index 0 in method: public void com.capone
   .transactions.controller.TransactionsController.postTransaction(com.capone.transactions
   .model.DetailedTransaction), with 1 error(s): [Field error in object
   'detailedTransaction' on field 'transactionDate': rejected value [201812]; codes
   [Pattern.detailedTransaction.transactionDate,Pattern.transactionDate,Pattern.java.lang
   .String,Pattern]; arguments [org.springframework.context.support
   .DefaultMessageSourceResolvable: codes [detailedTransaction.transactionDate
   ,transactionDate]; arguments []; default message [transactionDate],[Ljavax.validation
   .constraints.Pattern$Flag;@54d77241,org.springframework.validation.beanvalidation
   .SpringValidatorAdapter$ResolvableAttribute@1b36b52f]; default message [Invalid
   Transaction Date]] ",
4   "errors": [
5     "transactionDate: Invalid Transaction Date"
6   ]
7 }
```

- GET Transaction – Trying to get transactions on account with no transactions will return empty response with 204 Status

The screenshot shows a REST client interface with the following components:

- Top Bar:** Contains a list of tabs (http, http, http, http, http, http, http://loc), an environment dropdown set to "No Environment", and icons for visibility and settings.
- Request Bar:** Shows the method "GET", the URL "http://localhost:8080/accounts/cgfzw9OoEfwwMaj3spNb...", a "Params" button, a blue "Send" button, and a "Save" button.
- Request Tabs:** Includes "Authorization", "Headers (1)", "Body", "Pre-request Script", and "Tests".
- Authorization Panel:** Shows "TYPE" as "No Auth" and a message: "This request does not use any authorization. [Learn more about authorization](#)".
- Response Bar:** Shows "Body", "Cookies", "Headers (2)", and "Test Results". It also displays the status "204 No Content", time "28 ms", and size "110 B".
- Response Body Panel:** Includes tabs for "Pretty", "Raw", and "Preview", a "JSON" dropdown, a refresh icon, and a search icon. The body content is empty, with a line number "1" visible on the left.

- Get Transaction – Trying to fetch transactions and passing invalid account reference Id or transaction reference id will return error 400

The screenshot shows a REST client interface with a request bar at the top. The method is GET, and the URL is `http://localhost:8080/accounts/cgfzwNboFTDiklNCZL2pt...`. The environment is set to "No Environment". Below the request bar, the "Authorization" tab is selected, showing "No Auth" and a message: "This request does not use any authorization. [Learn more about authorization](#)".

The "Body" tab is also selected, showing the response body in JSON format. The response is a 400 Bad Request error with the following details:

```
1 {
2   "timestamp": "2018-09-21T14:31:47.768+0000",
3   "status": 400,
4   "error": "Bad Request",
5   "message": "Bad Request",
6   "path": "/accounts/cgfzwNboFTDiklNCZL2ptcPzDhWpo%3D/transactions"
7 }
```

The status bar at the bottom indicates the response status is "400 Bad Request", the time taken is "30 ms", and the size is "327 B".