SOFTWARE REQUIREMENT:

- JAVA 15

API:

Process Transaction: Process Transaction API is for adding a transaction into the system for a specific payer and date. This API requires a JSON input as given below.

API Call URL:

POST: http://localhost:9000/api/processTransaction

Input Format:

{ "payer": <String>, "points": <int>, "timestamp": <String> }

Example:

{ "payer": "DANNON", "points": 1000, "timestamp": "2020-11-02T14:00:00Z" }

WithDrawal: WithDrawal API is for deleting a specific amount from all users.

This API requires a JSON input as given below.

API Call URL:

POST: http://localhost:9000/api/withDrawal

Input Format:

{ "points": <int> }

Input Example:

{ "points": 5000 }

Output Format:

{ "payer": <String>, "points": <int> }

Output Example:

```
[
    { "payer": "DANNON", "points": -100 },
    { "payer": "UNILEVER", "points": -200 },
    { "payer": "MILLER COORS", "points": -4700 }
]
```

Balance: Balance API is getting all the balances of all users.

This API does not require a JSON input as given below.

API Call URL:

GET: http://localhost:9000/api/balance

Output Example:

```
[
    { "payer": "DANNON", "points": 1000 },
    { "payer": "UNILEVER", "points": 0 },
    { "payer": "MILLER COORS", "points": 5300 }
]
```

IMPORTANT: Cannot add transactions that makes the user account into negative. The application assumes that inputs will be valid transactions.

For example:

{ "payer": "DANNON", "points": -1000, "timestamp": "2020-11-02T14:00:00Z" }

Cannot add this transaction.

But the below example is a valid example.

{ "payer": "DANNON", "points": -1000, "timestamp": "2020-11-02T14:00:00Z" }

{ "payer": " DANNON ", "points": 1000, "timestamp": "2020-10-31T11:00:00Z" }

PROCEDURE:

Step 1: Download the Fetch Rewards zip file and extract it.

Step 2: Run the following instruction on command prompt

- Fetch Reward is the starting Directory.
- NAVIGATE to following directories.

  Fetch Rewards → demo → target

- Then Execute the following command in command prompt.
  Note: Before running the below command make sure PORT: 9000 is free.

```
java -jar demo-0.0.1-SNAPSHOT.jar
```

Step 3: Now you can call processTransaction API calls through various platforms like POSTMAN, CURL etc.

- For processTransaction API calls in CURL:

Input Format:

curl –X POST "http://localhost:9000/api/processTransaction" -H "accept: */*" -H "Content-Type: application/json" -d

"{\"points\":<int>,\ "timestamp\":\"<String>,\"payer\":\<String>\"}"

Input Example:

curl –X POST "http://localhost:9000/api/processTransaction" -H "accept: */*" -H "Content-Type: application/json" -d "{\"points\":1000,\ "timestamp\":\"2021-02-23T04:17:58.070Z\",\"payer\":\"ROHIT"\"}"

Input Format:

curl –X POST "http://localhost:9000/api/withDrawal" -H "accept: */*" -H "Content-Type: application/json" -d "{\"points\":<int >}"

Input Example:

curl –X POST "http://localhost:9000/api/withDrawal" -H "accept: */*" -H "Content-Type: application/json" -d "{\"points\":1000}"

- For balance API call in CURL:

Input Format:

curl –X POST "http://localhost:9000/api/balance" -H "accept: */*"

Input Example:

curl –X POST "http://localhost:9000/api/balance" -H "accept: */*"

processTransaction, withdrawal, balance API calls can be invoked using curl. These API can be invoked using POSTMAN too.

EXAMPLE:

{ "payer": "DANNON", "points": 1000, "timestamp": "2020-11-02T14:00:00Z" }

{ "payer": "UNILEVER", "points": 200, "timestamp": "2020-10-31T11:00:00Z" }

{ "payer": "DANNON", "points": -200, "timestamp": "2020-10-31T15:00:00Z" }

{ "payer": "MILLER COORS", "points": 10000, "timestamp": "2020-11-01T14:00:00Z" }

{ "payer": "DANNON", "points": 300, "timestamp": "2020-10-31T10:00:00Z" }

Transaction Call 1:

curl –X POST "http://localhost:9000/api/processTransaction" -H "accept: */*" -H “Content-Type: application/json" -d "{\"payer\":\"DANNON\",\"points\":1000,\"timestamp\":\"2020-11-02T14:00:00Z \"}"

Transaction Call 2:

curl –X POST "http://localhost:9000/api/processTransaction" -H "accept: */*" -H “Content-Type: application/json" -d "{\"payer\":\" UNILEVER\",\"points\":200,\"timestamp\":\"2020-10-31T11:00:00Z \"}"

Transaction Call 3:

curl –X POST "http://localhost:9000/api/processTransaction" -H "accept: */*" -H “Content-Type: application/json" -d "{\"payer\":\"DANNON\",\"points\":-200,\"timestamp\":\"2020-10-31T15:00:00Z \"}"

Transaction Call 4:

curl –X POST "http://localhost:9000/api/processTransaction" -H "accept: */*" -H “Content-Type: application/json" -d "{\"payer\":\"MILLER COORS \",\"points\":10000,\"timestamp\":\"2020-11-01T14:00:00Z \"}"

Transaction Call 5:

curl –X POST "http://localhost:9000/api/processTransaction" -H "accept: */*" -H “Content-Type: application/json" -d "{\"payer\":\"DANNON\",\"points\":300,\"timestamp\":\"2020-10-31T10:00:00Z \"}"

WithDrawal Call 1:

curl –X POST "http://localhost:9000/api/withDrawal" -H "accept: */*" -H "Content-Type: application/json" -d "{\"points\":5000}"

Output:

[{"payer":"DANNON","points":-100},{"payer":" UNILEVER","points":-200},{"payer":"MILLER COORS ","points":-4700}]

Balance Call 1:

curl –X GET "http://localhost:9000/api/balance" -H "accept: */*"

Output:

[{"payer":" UNILEVER","points":0},{"payer":"MILLER COORS ","points":5300},{"payer":"DANNON","points":1000}]