OptionA: A Web API For Bank Node App

SERVICE #1 - Create A new Bank Account

This service creates a new bank account. If there is already a bank account with the same account number, then this service returns an error message.

To Use The Create A new Bank Account Service

To create a bank account, make an HTTP POST request to this endpoint:

http://localhost:8080/save-new-customer

HTTP Request

Include the following JavaScript object (encoded as JSON) in the body of your POST request

```
{
 "accountNumber": 5500,
 "balance": 15000,
 "customerId": "002",
 "customerName": "Joe Swanson",
 "currencyType":"CAD"
}
```

HTTP Response

If the API request is successful the service will respond with a JavaScript object (encoded as JSON) similar to the example below:

```
{
   "message": "New Account Created Successfully With balance $15000 CAD "
}
```

If the API request is unsuccessful the service will respond with a JavaScript object (encoded as JSON) similar to the example below:

```
{
  "message": "Account Already exists. Please use another Account Number"
}
```

SERVICE #2 - Deposit

This service deposits money to bank accounts. If there is no matching account with the provided account number, then it returns an error message.

To Use The Service
To deposit, make an HTTP POST request to this endpoint:

http://localhost:8080/deposit

HTTP Request

Include the following JavaScript object (encoded as JSON) in the body of your POST request

```
{
 "accountNumber": 1010,
 "amount": 13726,
 "customerId": "002",
 "customerName": "Joe Swanson",
 "currencyType":"MXN"
}
```

1. currencyType can be "USD" or "CAD"

HTTP Response

If the API request is successful the service will respond with a JavaScript object (encoded as JSON) similar to the example below: But this message will be little change if currency is USD OR CAD

```
{
  "message": "Amount $1372.6 CAD  = $13726 MXN in account with account number 1010 by Joe
Swanson successfully deposited."
}
```

If the API request is unsuccessful the service will respond with a JavaScript object (encoded as JSON) similar to the example below:

```
{
  "message": "Account with account number 10100 don't exist"
}
```

SERVICE #3 - withdraw

This service withdraws money from bank accounts. If there is no matching account with the provided account number, it returns an error message. Customer Id and customer should match with the withdrawing account.

To Use The Service
To deposit, make an HTTP POST request to this endpoint:

http://localhost:8080/withdraw

HTTP Request

Include the following JavaScript object (encoded as JSON) in the body of your POST request

```
{
 "accountNumber": 5500,
 "amount": 5000,
 "customerId": "002",
 "customerName": "Joe Swanson",
 "currencyType":"CAD"
}
```

1. currencyType can be "USD" or “MXN”

HTTP Response

If the API request is successful the service will respond with a JavaScript object (encoded as JSON) similar to the example below: But this message will be little change if currency is USD OR MXN

```
  {
  "message": "Joe Swanson withdraw $5000 CAD  from account with account# 5500."
  }
```

If the API request is unsuccessful the service will respond with a JavaScript object (encoded as JSON) similar to the example below:

```
{
  "message": "No Account exists with given account"
```

}

There could be different messages like if withdraw money is greater than balance.

```
{
  "message": "Requested withdraw amount is greater than balance"
}
```

SERVICE #4 - transfer

This service lets users transfer money from one account to another. But customer Id or customer name should be equal to the account in the database for 'From Account'

To Use The Service
To deposit, make an HTTP POST request to this endpoint:

http://localhost:8080/transfer

HTTP Request

Include the following JavaScript object (encoded as JSON) in the body of your POST request

```
{
 "toAccountNum": 5500,
 "fromAccountnum": 1010,
 "amount": 7300,
 "customerId": "002",
 "customerName": "Joe Swanson",
 "currencyType":"CAD"
}
```

currencyType can be "USD" or "MXN"

HTTP Response

If the API request is successful the service will respond with a JavaScript object (encoded as JSON) similar to the example below: But this message will be little change if currency is USD OR MXN

```
{
  "message": "From account# 1010 $7300 CAD  successfully transfer to account# 5500"
}
```

If the API request is unsuccessful the service will respond with a JavaScript object (encoded as JSON) similar to the example below:

```
{
  "message": "To account with account# 55000 does not exist"
}
```

There could be other errors like transfer amount is greater than balance or something else

```
{
  "message": "Requested Transfer amount is greater than balance"
}
```

SERVICE #5 all-accounts

This service fetches all the accounts from the database and returns.

To Use The Service
To deposit, make an HTTP GET request to this endpoint:

http://localhost:8080/all-accounts

HTTP Response

If the API request is successful the service will respond with a JavaScript object (encoded as JSON) similar to the example below:

```
  "AllAccounts": {
     "13": {
        "AccountNumber": "13",
        "Balance": "20",
        "CustomerId": "13",
        "CustomerName": "simer"
     },
     "1313": {
        "AccountNumber": "1313",
        "Balance": 3300,
        "CustomerId": "10",
        "CustomerName": "for jest",
        "depositHistory": "Simer deposited $20 CAD  in this account at 2020-7-27 10:9:15"
     },
      "1314": {
        "AccountNumber": "1314",
        "Balance": 2200,
        "CustomerId": "11",
        "CustomerName": "simer"
      }
    }
}
```

SERVICE #6 - checking balance or account

This service fetches a single account from the database.

To Use The Service
To deposit, make an HTTP POST request to this endpoint:

http://localhost:8080/check-balance

HTTP Request

Include the following JavaScript object (encoded as JSON) in the body of your POST request

```
{
  "accountNumber":"185"
}
```

HTTP Response

If the API request is successful the service will respond with a JavaScript object (encoded as JSON) similar to the example below:

```
{
  "account": {
    "AccountNumber": "185",
    "Balance": 3310,
    "CustomerId": "23",
    "CustomerName": "simer",
            "depositHistory": "simer deposited $undefined CAD  in this account at 2020-7-26 14:42:15”


  }
```

}

If the API request is unsuccessful the service will respond with a JavaScript object (encoded as JSON) similar to the example below:

```
{
  "message": "Validation failed, entered data is incorrect."
}
```