UNIVERSIDAD PERUANA DE CIENCIAS APLICADAS
APLICACIONES OPEN SOURCE (SI729)
2023-1

Se le encarga el desarrollo de una API REST que permita las siguientes funcionalidades.

Requisitos funcionales

- Registro de información
  - Registro de cuenta bancaria (POST)
  - Registro de transacción bancaria. (POST)
- Listado de información
  - Listado de todas las cuentas bancarias (GET)
  - Listado de transacciones bancarias por nombre de cliente (GET)
  - Listado de transacciones bancarias por rango de fechas. (GET)

Atributos de las clases / Columnas de las tablas

La relación es de uno a muchos desde Account (Cuenta bancaria) a Transaction (Transacción bancaria)

Clases	Tablas
Account ======= private Long id private String nameCustomer private String numberAccount Transaction ========= private Long id private String type private LocalDate createDate private Double amount private Double balance private Account account	accounts ======== id name_customer number_account transactions ========= id type create_date amount balance account_id

Base de datos

- El nombre de la base de datos seria db_backend
- El gestor de base de datos debe ser **PostgreSQL**. El nombre de las tablas debe ser “accounts” y “transactions”

Reglas de negocio.

Account

======

- Validar que los atributos del Account cumplan las siguientes restricciones
    - Cuando no se ingrese un nameCustomer se debe mostrar el mensaje “El nombre del cliente debe ser obligatorio”
    - Cuando se ingrese un nameCustomer que excede los 30 caracteres se debe mostrar el mensaje “El nombre del cliente no debe exceder los 30 caracteres”
    - Cuando no se ingrese un valor para numberAccount se debe mostrar el mensaje “El número de cuenta debe ser obligatorio”
    - Cuando se ingrese un numberAccouunt que no tiene 13 caracteres se debe mostrar el mensaje “El número de cuenta debe tener una longitud de 13 caracteres”.

- No se debe permitir el registro de un Account con el mismo nameCustomer y numberAccount
    - Cuando se trate de registrar un Account con un nameCustomer y numberAccount que ya existe se debe mostrar el siguiente mensaje “No se puede registrar la cuenta porque ya existe uno con estos datos”.

Transaction

========

- Para realizar el registro de una transacción bancaria se debe enviar el type, amount, balance y account (id). El valor de createDate se obtiene del sistema. En el caso del valor para type debe ser “Retiro” o “Deposito”
- Validar que los atributos del Transaction cumplan las siguientes restricciones
    - Cuando no se ingrese un type se debe mostrar el mensaje “El tipo de transacción bancaria debe ser obligatorio”
    - Cuando se ingrese un amount menor igual a 0.0 debe mostrar el mensaje “El monto en una transacción bancaria debe ser mayor de S/.0.0”. Esto debe cumplirse tanto para una transacción bancaria tipo “Deposito” o “Retiro”
    - Cuando se ingrese un valor de amount mayor al valor del balance se debe mostrar el mensaje “En una transacción bancaria tipo retiro el monto no puede ser mayor al saldo”.
    - Cuando el tipo de transacción bancaria (type) sea “Deposito” el amount se debe sumar al balance.
    - Cuando el tipo de transacción bancaria (type) sea “Retiro” el amount se debe restar del balance. Siempre y cuando el amount sea menor al balance.

Requisitos no funcionales

- Puede utilizar Postman o Swagger para realizar la prueba del API REST.
- Debe crear clases personalizadas de excepciones para los siguientes casos

- Recurso no encontrado: ResourceNotFoundException
- Validación de atributos: ValidationException
- Gestor de exceptiones (Handler): ControllerExceptionHandler
- La información del error debe ser representada mediante una clase de nombre ErrorMessage, el cual debe tener la siguiente información:

  private int statusCode;
  private String message;

  private String description;
  private LocalDateTime timestamp;

- El nombre de los endpoints (url) a utilizar en los controllers debe ser
  - La ruta base de los endpoints debe ser /api/bank/v1
  - La ruta para el registro de cuenta bancaria debe ser /api/bank/v1/accounts
  - La ruta para el listado de cuentas bancarias debe ser /api/bank/v1/accounts
  - La ruta para el registro de transacciones bancarias debe ser /api/bank/v1/accounts/{id}/transactions
  - La ruta para el listado de transacciones por nombre de cliente /api/bank/v1/transactions/filterByNameCustomer
  - La ruta para el listado de transacciones bancarias por rango de fechas debe ser /api/bank/v1/transactions/filterByCreateDateRange

1. Registro de cuenta bancaria (POST)

Escenario 1: registro de cuenta bancaria exitoso

Request: http://localhost:8080/api/bank/v1/accounts

{

"nameCustomer":"Luis Nicolas Ramirez Pocohuanca ",

"numberAccount":"6003067672367"

}

Response:

{

"id": 1,

"nameCustomer": "Luis Nicolas Ramirez Pocohuanca ",

"numberAccount": "6003067672367"

}

Escenario 2: registro de una cuenta bancaria con un nameCustomer y numberAccount que ya existe

Request: http://localhost:8080/api/banl/v1/accounts

{

"nameCustomer":"Luis Nicolas Ramirez Pocohuanca ",

"numberAccount":"6003067672367"

}

Response:

{

"statusCode": 400,

"message": "No se puede registrar la cuenta porque ya existe uno con estos datos",

"description": "uri=/api/bank/v1/accounts",

"timestamp": "29-11-2022 06:56:13"

}

Escenario 3: registro de una cuenta bancaria sin enviar un valor de nameCustomer o numberAccount

Request: http://localhost:8080/api/bank/v1/accounts

{

"nameCustomer":"",

"numberAccount":"6003067672364"

}

{

"nameCustomer":"Patricia Plasencia Burgos",

"numberAccount":""

}

Response


```
{
   "statusCode": 400,
   "message": "El nombre del cliente debe ser obligatorio",
   "description": "uri=/api/bank/v1/accounts",
   "timestamp": "29-11-2022 06:57:58"
}
```

```
{
   "statusCode": 400,
   "message": "El número de cuenta debe ser obligatorio",
   "description": "uri=/api/bank/v1/accounts",
   "timestamp": "29-11-2022 06:59:20"
}
```


2. Registro de transacción bancaria (POST)

Escenario 1: registro de transacción bancaria exitoso

Request: http://localhost:8080/api/bank/v1/accounts/1/transactions


```
{
   "type":"Retiro",
   "amount":500,
   "balance":700,
   "account":{
      "id":1
   }
}
```


Response:


```
{
   "id": 5,
   "type": "Retiro",
   "createDate": "2022-11-29",
```

```
    "amount": 500.0,
    "balance": 200.0
}
```

Escenario 2: registro de transacción bancaria sin enviar tipo (type)

Request: http://localhost:8080/api/bank/v1/accounts/1/transactions

```
{
    "type":"",
    "amount":500,
    "balance":700,
    "account":{
        "id":1
    }
}
```

Response

```
{
    "statusCode": 400,
    "message": "El tipo de transacción bancaria debe ser obligatorio",
    "description": "uri=/api/bank/v1/accounts/1/transactions",
    "timestamp": "29-11-2022 07:03:42"
}
```

Escenario 3: registro de transacción bancaria con valor de amount mayor al balance y tipo de transacción es Retiro

Request: http://localhost:8080/api/bank/v1/accounts/1/transactions

```
{
    "type":"Retiro",
    "amount":800,
    "balance":700,
    "account":{
```

"id":1

}

}

Response

{

"statusCode": 400,

"message": "En una transacción bancaria tipo retiro el monto a retirar no puede ser mayor al saldo ",

"description": "uri=/api/bank/v1/accounts/1/transactions",

"timestamp": "29-11-2022 07:05:16"

}

Escenario 4: registro de transacción bancaria con valor de 0.0 en el amount

Request: http://localhost:8080/api/bank/v1/accounts/1/transactions

{

"type":"Deposito",

"amount":0.0,

"balance":700,

"account":{

"id":1

}

}

{

"type":"Retiro",

"amount":0.0,

"balance":700,

"account":{

"id":1

}

}

Response

{

"statusCode": 400,

"message": "El monto en una transacción bancaria debe ser mayor de S/0.0",

"description": "uri=/api/bank/v1/accounts/1/transactions",

"timestamp": "29-11-2022 07:10:59"

}

3. Listado de todas las cuentas bancarias (GET)
   Request: http://localhost:8080/api/bank/v1/accounts

   Response


```
[
  {
    "id": 1,
    "nameCustomer": "Pedro Perez Rodriguez",
    "numberAccount": "6003067678567"
  },
  {
    "id": 2,
    "nameCustomer": "Luis Nicolas Ramirez Pocohuanca ",
    "numberAccount": "6003067672367"
  },
  {
    "id": 3,
    "nameCustomer": "Tania Chicana Messi",
    "numberAccount": "6003067672364"
  }
]
```


4. Listado de transacciones bancarias por nombre de cliente (GET)
   Request:
   http://localhost:8080/api/bank/v1/transactions/filterByNameCustomer?nameCustomer=Luis Nicolas Ramirez Pocohuanca

   Response


```
[
  {
    "id": 6,
    "type": "Retiro",
    "createDate": "2022-11-29",
```

```
    "amount": 100.0,
    "balance": 600.0
  },
  {
    "id": 7,
    "type": "Deposito",
    "createDate": "2022-11-29",
    "amount": 100.0,
    "balance": 800.0
  }
]
```


5. Listado de transacciones bancarias por rango de fechas. (GET)

Request

http://localhost:8080/api/bank/v1/transactions/filterByCreateDateRange?startDate=2022-11-28&endDate=2022-11-29

Response


```
[
  {
    "id": 5,
    "type": "Retiro",
    "createDate": "2022-11-28",
    "amount": 100.0,
    "balance": 600.0
  },
  {
    "id": 6,
    "type": "Retiro",
    "createDate": "2022-11-29",
    "amount": 100.0,
    "balance": 600.0
  },
  {
    "id": 7,
    "type": "Deposito",
    "createDate": "2022-11-29",
    "amount": 100.0,
    "balance": 800.0
  }
]
```