JBDL-53

E-Wallet

6th August, 2023

OVERVIEW

GOALS

1. You have to create an Entity Relationship Diagram based on a database schema.
2. Create API/endpoints as per Controllers.
3. Implement Security.

SPECIFICATIONS

Entity Involved:

1. User
2. Profile
3. Transaction
4. Account

Database Schema [Include CreatedAt and UpdatedAt on all the Schema]

User

1. UserName
2. Email
3. Name
4. Phone Number
5. Password
6. Age
7. Country

Profile

1. Name

2. Wallet Balance
3. Bank Account Details
4. Created On

Account

1. Id
2. Transaction ID
3. Transaction Type
4. Desc
5. Amount
6. Balance
7. Status

Transaction

1. Id
2. Source UserId
3. Source Profile ID
4. Source Account Number
5. To Account Number
6. To UserId
7. To Profile ID
8. Status
9. Amount
10. Desc
11. CreatedOn
12. UpdatedOn

Relations

Source	Destination	Relation
User	Profile	1-1
Profile	Account	1-1
Account	Transaction	M-N

Controller/Endpoints/API

User Controller

Request Mapping: /user/<Endpoint>

1. CRUD API for User
2. Allow User to update Password

Profile Controller

Request Mapping: /profile/<Endpoint>

1. CRUD API for Profile

Account Controller

Request Mapping: /account/<Endpoint>

1. CRUD API for Account

Transactions Controller

Request Mapping: /transact/<Endpoint>

1. Send Money (Parameters: Source Account ID, Destination Phone Number, Amount, Desc)
    a. If the both accounts are active
    b. The Sender should have balance after money is sent
    c. There should be a limit on the number of transactions allowed only on the sender side and on the amount the validation should be done on both ends.
    d. If all of the above are ok, process the transaction
        i. Create a new transaction
        ii. Insert both the entries in the Account table
        iii. Update the balance in the profile
        iv. Kafka: Drop a kafka message for an email to be sent to the user, email Id has to be passed with the details of transaction.
    e. If any of the above return an error, insert a transaction as failure.
    f. Add logic to commit the transaction on both accounts in case of success otherwise, you have to handle the reversal part.(How this can be done)
        i. Approach 1: Dummy Table
        ii. Approach 2: Using Commit and rollback

iii. Approach 3: Using Kafka as a broker to store the transaction and remove only when processed.

Reports Controller

Request Mapping: /report/<Endpoint>

1. All Transaction for a account (Parameters: StartDate, EndDate)
2. List of all active Accounts
3. Total Number of transactions processed with Amount