**Wallet Topup and Deduct System**

API Overview

This API manages user accounts, allowing users to top up and deduct balances. It also records transactions and provides endpoints for querying user balances and transaction details.

Base URL

http://localhost:8000/

Authentication

No authentication is currently required for accessing these endpoints. However, you can implement authentication (e.g., API keys, OAuth) if needed.

Endpoints

1. Top Up User Balance

- Endpoint: `/topup`

- Method: `POST`

- Description: This endpoint allows a user to top up their account balance. If the user doesn’t exist, a new user account is created.

Request Body (`TopupRequest`)

{

"user\_id": "string",

"amount": "float"

}

- `user\_id`: The unique ID of the user.

- `amount`: The amount to be added to the user's balance.

Response (`TransactionResponse`)

{

"status": true,

"new\_balance": "float",

"transaction\_id": "string"

}

- `status`: Whether the top-up operation was successful.

- `new\_balance`: The updated balance of the user.

- `transaction\_id`: The unique ID of the top-up transaction (in the format: `user\_id\_timestamp`).

Example Request

POST /topup

{

"user\_id": "12345",

"amount": 100.00

}

Example Response

{

"status": true,

"new\_balance": 200.00,

"transaction\_id": "12345\_20230914123456"

}

2. Deduct User Balance

- Endpoint: `/deduct`

- Method: `POST`

- Description: This endpoint allows a user to deduct an amount from their balance. The deduction will only succeed if the user has sufficient balance.

Request Body (`TopupRequest`)

{

"user\_id": "string",

"amount": "float"

}

Response (`TransactionResponse`)

{

"status": true,

"new\_balance": "float",

"transaction\_id": "string"

}

Example Request

POST /deduct

{

"user\_id": "12345",

"amount": 50.00

}

Example Response

{

"status": true,

"new\_balance": 150.00,

"transaction\_id": "12345\_20230914123600"

}

Error Responses

- `404`: User not found.

- `400`: Insufficient balance.

3. Get User Balance

- Endpoint: `/balance/{user\_id}`

- Method: `GET`

- Description: This endpoint retrieves the balance of a specific user.

Path Parameter

- `user\_id`: The unique ID of the user.

Response (`UserResponse`)

{

"user\_id": "string",

"amount": "float"

}

Example Request

GET /balance/12345

Example Response

{

"user\_id": "12345",

"amount": 150.00

}

Error Responses

- `404`: User not found.

4. Get All Users

- Endpoint: `/users`

- Method: `GET`

- Description: This endpoint retrieves the details (user ID and balance) of all users in the system.

Response (`List[UserResponse]`)

[

{

"user\_id": "string",

"amount": "float"

}

]

Example Request

GET /users

Example Response

[

{

"user\_id": "12345",

"amount": 150.00

},

{

"user\_id": "67890",

"amount": 200.00

}

]

5. Get All Transactions

- Endpoint: `/transactions`

- Method: `GET`

- Description: This endpoint retrieves all transaction details recorded in the system.

Response (`List[TransactionResponseDetail]`)

[

{

"transaction\_id": "string",

"user\_id": "string",

"amount": "float"

}

]

Example Request

GET /transactions

Example Response

[

{

"transaction\_id": "12345\_20230914123456",

"user\_id": "12345",

"amount": 100.00

},

{

"transaction\_id": "12345\_20230914123600",

"user\_id": "12345",

"amount": -50.00

}

]

Database Models

User

- user\_id (`String`): Unique identifier for the user.

- amount (`Float`): The balance of the user.

Transaction

- transaction\_id (`String`): Unique identifier for each transaction.

- user\_id (`String`): The user associated with the transaction.

- amount (`Float`): The transaction amount. Positive for top-ups, negative for deductions.

Error Codes

- `404 Not Found`: The requested resource (user or transaction) was not found.

- `400 Bad Request`: Insufficient balance for deduction.

- `500 Internal Server Error`: There was a problem with the database.

Rate Limiting

As the objective was of handling at least 100 requests per second, this api is capable of handling it.