Payments_dodo

This API documentation provides a backend service for managing user accounts, authentication, transactions, and account balances. Built using Rust with Actix Web, it follows RESTful principles to ensure scalability, reliability, and security. The service integrates PostgreSQL for data persistence and JWT for authentication.EndFragment

User

User API contains endpoints that handle user management operations like, register user, get token, get user details and update user details.

POST 0.0.0.0:8080/user/register_user

0.0.0.0:8080/user/register_user

User Register User

This endpoint is used to retrieve a user.

Request Body

The request body should be in JSON format and include the following parameters:

- `username` (string): The username of the new user.
- `email` (string): The email address of the new user.
- `password` (string): The password for the new user.

Response

The response will be in JSON format and may include the following fields:

- `detailed_Message` (string): A detailed message regarding the registration process.
- `message` (string): A general message related to the registration status.
- `status` (string): The status of the registration process.

JSON Schema

json

```json
{
    "type": "object",
    "properties": {
      "properties": {
        "detailed_Message": {
          "type": "string"
        },
        "message": {
          "type": "string"
        },
        "status": {
          "type": "string"
```

The request body should be in JSON format and include the following parameters:

- `username` (string): The username of the new user.
- `email` (string): The email address of the new user.
- `password` (string): The password for the new user.

Response

The response will be in JSON format and includes the following fields:

- `detailed_Message` (string): A detailed message regarding the registration process.
- `message` (string): A general message related to the registration status.
- `status` (string): The status of the registration process.

JSON Schema

json

```json
{
  "type": "object",
  "properties": {
    "detailed_Message": {
      "type": "string"
    },
    "message": {
      "type": "string"
    },
    "status": {
      "type": "string"
```

Body raw (json)

json

```json
{
    "username":"test_receiver",
    "email":"test2@test.com",
    "password":"test"
}
```

GET 0.0.0.0:8080/user/get_token

0.0.0.0:8080/user/get_token

User Get Token

This endpoint is used to retrieve a token for a user.

Request

- Method: GET
- URL: `:8080/user/get_token`
- Body:
  - email (text, required): The email of the user.
  - password (text, required): The password of the user.

Response

The response will be a JSON object with the following properties:

- `message` (string): A message related to the request.
- `status` (string): The status of the request.
- `token` (string): The token retrieved for the user.

Body raw (json)

json

```json
{
    "email":"test@test.com",
    "password":"test"
}
```

POST 0.0.0.0:8080/user/update_user

0.0.0.0:8080/user/update_user

Update User Information

This endpoint allows the client to update user information.

Request Body

- email (string, required): The email address of the user.
- username (string, required): The updated username for the user.

Response

The response will be in JSON format and will include the following fields:

- message (string): A message indicating the result of the update operation.
- status (string): The status of the update operation.

AUTHORIZATION Bearer Token

Token	<token>

Body raw (json)

json

```json
{
    "username":"test_updated"
}
```

GET 0.0.0.0:8080/user/get_user

0.0.0.0:8080/user/get_user

This endpoint makes an HTTP GET request to retrieve user information. The request should include the user's ID in the raw request body.

Request

- user_id (text, required): The ID of the user for whom the information is requested.

Response

The response will include the user's ID, username, email, created_at, and updated_at. Here is an example response:

json

```json
{
    "created_at":"",
    "email":"",
    "id":0,
    "updated_at":"",
    "user_id":"",
    "username":""
}
```

AUTHORIZATION Bearer Token

Token	<token>

Body raw (json)

json

```json
{
    "user_id":"be296e10-7c91-485d-a5fa-4cb8a949d4f7"
}
```

Balance

Balance API contains endpoints that helps to fetch the balance of the respective user.

GET 0.0.0.0:8080/balance/fetch_balance

0.0.0.0:8080/balance/fetch_balance

Balance Fetch

The `fetch_balance` endpoint retrieves the balance for a specific user.

Response

The response for this request follows the JSON schema below:

json

```json
{
    "type": "object",
    "properties": {
        "user_id": {
            "type": "number"
        },
        "balance": {
            "type": "string"
        }
    }
}
```

AUTHORIZATION Bearer Token

Token	<token>

Transactions

Transaction API contains endpoints that handles the transaction related operation like perform a transaction, fetch all the transaction of the user and fetch the details of a transaction.

POST 0.0.0.0:8080/transaction/operations

0.0.0.0:8080/transaction/operations

Transaction Operations

This endpoint is used to perform transaction operations.

Request Body

- Parameters required for the request body.
    - receiver:
        - Type string
        - It is the user_id of the receiver in case of transfer transaction type and is null in other case.
    - amount:
        - Type float
    - transaction_type:
        - withdrawl
        - deposit
        - transfer

```json
{  "receiver": null/"<user_id>",
  "amount" : 100.00,
  "transaction_type" : "deposit"
}
```

Response

```json
{
  "message": "",
  "status": ""
}
```

AUTHORIZATION Bearer Token

Token	<token>

Body raw (json)

```json
{
    "receiver":null,
    "amount":100.00,
    "transaction_type":"deposit"
}
```

GET 0.0.0.0:8080/transaction/list_trans

0.0.0.0:8080/transaction/list_trans

List transactions

This endpoint retrieves a list of transactions. The request does not require any parameters. The response will include an array of transaction objects, each containing a UUID, sender ID, receiver ID, amount, transaction type, and status.

AUTHORIZATION Bearer Token

Token	<token>

GET 0.0.0.0:8080/transaction/fetch_transaction

0.0.0.0:8080/transaction/fetch_transaction

Fetch Transaction

This endpoint retrieves a transaction based on the provided UUID.

Request

The request should be sent via an HTTP GET method to the following URL: `0.0.0.0:8080/transaction/fetch_transaction`

The request body should contain the following payload in raw format:

```json
{
    "uuid": "f123fcd5-6bd2-4051-b30e-39227c ..."
}
```

Response

.

The response will be a JSON object with the following schema:

```json
{
    "uuid": "",
    "sender": 0,
    "receiver": 0,
    "amount": "",
    "transaction_type": "",
    "status": ""
}
```

The `uuid` field represents the unique identifier of the transaction. The `sender` and `receiver` fields denote the sender and receiver IDs respectively. The `amount` field signifies the transaction amount. The `transaction_type` field specifies the type of transaction, and the `status` field indicates the current status of the transaction.

AUTHORIZATION Bearer Token

Token	<token>

Body raw (json)

```json
{
    "transaction_id":"21fb8729-a50d-4d96-aec1-6f346e721d59"
}
```