Frontend : https://personal-budget-liart.vercel.app/login
Backend : https://budget-tracker-be-1.onrender.com/users/login/

Live Login Credentials :

Email : test@gmail.com
Password : Test@123

API Documentation

Credentials to Access the Platform

To access the API, you need valid credentials. You can follow these steps to get access:

1. **Register** an account using the `POST /users/register/` endpoint.
2. Once registered, use the `POST /users/login/` endpoint to obtain a **JWT Token**.

After logging in, you will receive a JWT token that must be included in the `Authorization` header for all protected routes.

Request:

```
{
    "email": "test@gmail.com",
    "password": "Test@123"
}
```

Response:

```
{
    "token": "your_jwt_token"
}
```

Use the token in the header for subsequent requests:

```
Authorization: Bearer your_jwt_token
```

User Authentication

1. Register User

- **Method**: `POST`
- **Endpoint**: `/users/register/`
- **Description**: Register a new user with the system.

Request Body:

```
{
    "email": "string",
    "password": "string"
}
```

2. Login User

- **Method**: `POST`
- **Endpoint**: `/users/login/`
- **Description**: Login and obtain a JWT token for the user.

Request Body:

```
{
    "username": "string",
    "password": "string"
}
```

3. Logout User

- **Method**: `POST`
- **Endpoint**: `/users/logout/`
- **Description**: Logout the user and blacklist the JWT token.
- **Authorization**: Requires a valid JWT token.

Budget Management

1. Create Budget

- **Method**: `POST`
- **Endpoint**: `/budget/budget/`
- **Description**: Create a new monthly budget.

Request Body:

```
{
    "month": "int",
    "year": "int",
    "amount": "float"
}
```

2. List All Budgets

- **Method**: `GET`
- **Endpoint**: `/budget/budget/`
- **Description**: Retrieve all budgets for the authenticated user.

3. Get Budget by ID

- **Method**: `GET`
- **Endpoint**: `/budget/budget/<id>/`
- **Description**: Retrieve a specific budget by its ID.

4. Update Budget

- **Method**: `PUT`
- **Endpoint**: `/budget/budget/<id>/`
- **Description**: Update an existing budget by its ID.

Request Body:

```
{
    "amount": "float"
}
```

5. Delete Budget

- **Method**: `DELETE`
- **Endpoint**: `/budget/budget/<id>/`
- **Description**: Delete a specific budget by its ID.

6. View Budget Summary

- **Method**: `GET`
- **Endpoint**: `/budget/budget-summary/`
- **Description**: Retrieve a summary of the monthly budget including income, expenses, and remaining balance.

7. View Total Expenses Over Time

- **Method**: `GET`
- **Endpoint**: `/budget/total-expenses-over-time/`
- **Description**: Retrieve the total expenses over a period of time (e.g., by month or year).

Transaction Management

1. Create Transaction

- **Method**: `POST`
- **Endpoint**: `/budget/transactions/`
- **Description**: Create a new transaction entry.

Request Body:

```
{
    "category": "string",
    "amount": "float",
    "description": "string",
    "date": "string (YYYY-MM-DD)"
}
```

2. List All Transactions

- **Method**: `GET`
- **Endpoint**: `/budget/transactions/`
- **Description**: List all transactions.

3. Get Transaction by ID

- **Method**: `GET`
- **Endpoint**: `/budget/transactions/<id>/`
- **Description**: Retrieve a specific transaction by its ID.

4. Update Transaction

- **Method**: `PUT`
- **Endpoint**: `/budget/transactions/<id>/`
- **Description**: Update an existing transaction by its ID.

Request Body:

```
{
    "amount": "float",
    "description": "string"
}
```

5. Delete Transaction

- **Method**: `DELETE`
- **Endpoint**: `/budget/transactions/<id>/`
- **Description**: Delete a specific transaction by its ID.

6. Spending by Category

- **Method**: `GET`
- **Endpoint**: `/budget/spending-by-category/`
- **Description**: View spending statistics grouped by category.

7. Category Choices

- **Method**: `GET`
- **Endpoint**: `/budget/categories/`
- **Description**: List available categories for transactions.

Assumptions Made for Features

While developing the application, the following reasonable assumptions were made:

1. **User Roles**: The application assumes that only authenticated users can access protected endpoints. Role-based access (e.g., admin vs. regular users) was not explicitly mentioned, so it is assumed that all authenticated users have similar access unless explicitly mentioned.

2. **Categories for Transactions**: A predefined set of categories is used for transactions (e.g., "Food", "Transport", "Entertainment"). The assumption is made that these categories are enough to cover general expenses.

3. **Budget and Transaction Integrity**: The application assumes that users cannot set a budget that exceeds their expected income. There is no automated check for income vs. budget limits, so the user must ensure the budget is within their financial limits.

4. **JWT Token Lifespan**: The JWT token is assumed to expire after a certain period (e.g., 24 hours). The user needs to log in again to obtain a new token.

Project Set Up

Backend Setup

📚 Project Setup Instructions

1. **Clone the Project Repository**

To get started, clone the project repository to your local machine:

```bash

https://github.com/rahul-312/Personal-Budget.git

cd budget_tracker_be

python -m venv venv

# Activate the virtual environment

source venv/bin/activate   # For Linux/MacOS

venv\Scripts\activate      # For Windows

cd budget_tracker

pip install -r requirements.txt

python manage.py migrate

python manage.py createsuperuser

python manage.py runserver
```

For React Js

git clone https://github.com/rahul-312/Personal-Budget.git

cd budget_tracker_fe

npm install

npm start