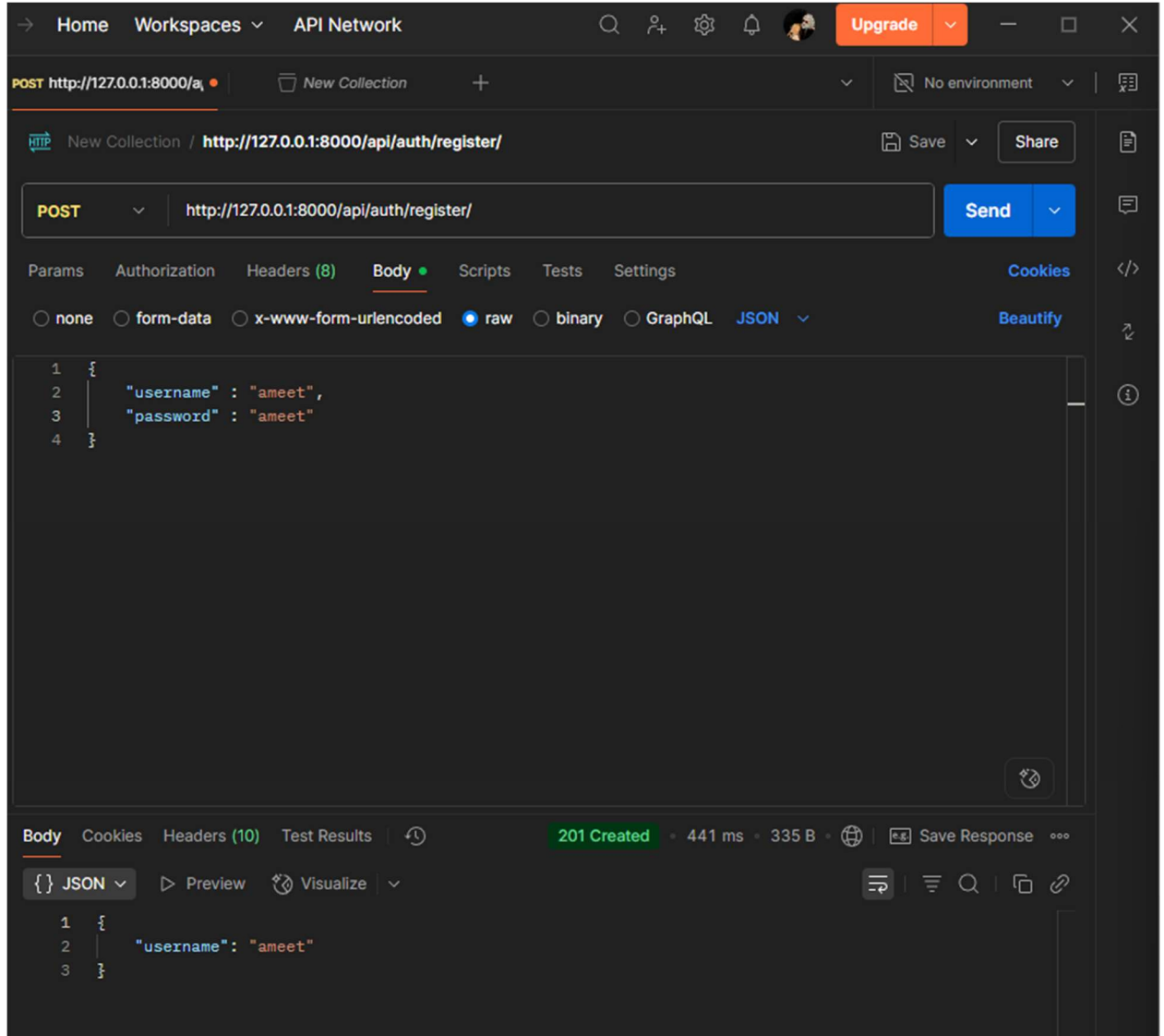


## Testing Checklist

### Authentication Tests

#### 1. User registration with valid data



The screenshot displays the Postman interface for a REST client. The top navigation bar includes 'Home', 'Workspaces', and 'API Network'. The main workspace shows a 'POST' request to the endpoint 'http://127.0.0.1:8000/api/auth/register/'. The request body is set to 'raw' and contains a JSON object with 'username' and 'password' both set to 'ameet'. The bottom panel shows the response, which is a '201 Created' status with a response time of 441 ms and a body size of 335 B. The response body is displayed in JSON format, showing only the 'username' field.

Home Workspaces API Network

POST http://127.0.0.1:8000/api/auth/register/

New Collection / http://127.0.0.1:8000/api/auth/register/ Save Share

POST http://127.0.0.1:8000/api/auth/register/ Send

Params Authorization Headers (8) Body Scripts Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
1 {
2   "username" : "ameet",
3   "password" : "ameet"
4 }
```

Body Cookies Headers (10) Test Results 201 Created • 441 ms • 335 B Save Response

{ } JSON Preview Visualize

```
1 {
2   "username": "ameet"
3 }
```

## 2. User registration with duplicate email/username

The screenshot displays the Postman interface for a REST client. At the top, the navigation bar includes 'Home', 'Workspaces', and 'API Network'. The main header shows the request method 'POST' and the URL 'http://127.0.0.1:8000/api/auth/register/'. Below this, the 'Body' tab is selected, showing a JSON payload: 

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

. The status bar at the bottom indicates a '400 Bad Request' response with a 6 ms duration and 377 B of data. The response body is shown in JSON format: 

```
{  "username": [    "A user with that username already exists."  ]}
```

Home Workspaces API Network

POST http://127.0.0.1:8000/api/auth/register/

New Collection / http://127.0.0.1:8000/api/auth/register/ Save Share

POST http://127.0.0.1:8000/api/auth/register/ Send

Params Authorization Headers (8) Body Scripts Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
1 {
2   "username": "ameet",
3   "password": "ameet"
4 }
```

Body Cookies Headers (10) Test Results 400 Bad Request 6 ms 377 B Save Response

{ JSON Preview Visualize

```
1 {
2   "username": [
3     "A user with that username already exists."
4   ]
5 }
```

### 3. User login with valid credentials

The screenshot displays the Postman interface for an API request. The top navigation bar includes 'Home', 'Workspaces', and 'API Network'. The main header shows the request method 'POST' and the URL 'http://127.0.0.1:8000/api/auth/login/'. The 'Body' tab is selected, showing a JSON payload with 'username' and 'password' both set to 'ameet'. The response section at the bottom indicates a '200 OK' status with a response time of 357 ms and a body size of 794 B. The response body is shown in JSON format, containing 'refresh' and 'access' tokens.

**Request:**

```
POST http://127.0.0.1:8000/api/auth/login/

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

**Response:**

```
{
  "refresh": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2t1b190eXB1IjoicmVmcVzaCI6ImV4cCI6MTc0MTc0Mzg2MiwiYWFWIjoxNzU3NDYyLCJqdGkiOiI3OGJhMDZhYjRkMDI0OWM0YjU1NGI0YzdkZjFmYjJmOSIsInVzZXJfaWQiOiJvV9.KNC_F79cfQF1ed2QExX-TFy06khIJbeFP8B6214-W6g",
  "access": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2t1b190eXB1IjoicmVmcVzaCI6ImV4cCI6MTc0MTc0Mzg2MiwiYWFWIjoxNzU3NDYyLCJqdGkiOiI3OGJhMDZhYjRkMDI0OWM0YjU1NGI0YzdkZjFmYjJmOSIsInVzZXJfaWQiOiJvV9.KNC_F79cfQF1ed2QExX-TFy06khIJbeFP8B6214-W6g"
}
```

#### 4. User login with invalid credentials

The screenshot shows the Postman interface for a POST request to `http://127.0.0.1:8000/api/auth/login/`. The request body is a JSON object with `username: "ameet"` and `password: "ameet1234567"`. The response is a 401 Unauthorized status with a message: `"detail": "No active account found with the given credentials"`.

**Request Details:**

- Method: POST
- URL: `http://127.0.0.1:8000/api/auth/login/`
- Body (raw):

```
1 {  
2   "username": "ameet",  
3   "password": "ameet1234567"  
4 }
```

**Response Details:**

- Status: 401 Unauthorized
- Time: 349 ms
- Size: 421 B
- Body (JSON):

```
1 {  
2   "detail": "No active account found with the given credentials"  
3 }
```

## 5. Token refresh functionality

The screenshot displays a REST client interface with a dark theme. At the top, navigation tabs include 'Home', 'Workspaces', and 'API Network'. The main header shows a 'POST' request to 'http://127.0.0.1:8000/api/auth/register/'. Below this, a dropdown menu shows the selected method 'POST' and the URL 'http://127.0.0.1:8000/api/auth/refresh/'. A 'Send' button is visible. The 'Body' tab is active, showing a JSON payload with a 'refresh' token. The response section at the bottom shows a '200 OK' status with a JSON body containing an 'access' token. The interface includes various tabs like 'Params', 'Authorization', 'Headers (8)', 'Scripts', 'Tests', 'Settings', 'Cookies', and 'Test Results'. The response status bar indicates '200 OK', '3 ms', and '552 B'.

```
POST http://127.0.0.1:8000/api/auth/register/

POST http://127.0.0.1:8000/api/auth/refresh/

{
  "refresh": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2t1b190eXB1IjoicmVmcmVzaCI6ImV4cCI6MTc0MTc0Mzg2MiwiaWF0IjoxNzU3NDYyLCJqdGkiOiI3OGJhMDZhYjRkMDI0OWM0YjU1NGI0YzdkZjFmYjJmOSIsInVzZXJfaWQiOiJvV9.KNC_F79cfQF1ed2QExX-TFy06khIJbeFP8B6214-W6g"
}
```

```
{
  "access": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2t1b190eXB1IjoiiYWNjZXRzIiwiaXhwIjoxNzU3NDYyLCJpYXQiOiJlE3NTE2NTc0NjIsImp0aSI6ImQ3Mjk5M2I2Y2IwZTQ0NDU5ZmExMTc0NDY3NDY1NWlwiIiwidXNlc19pZCI6NX0.w-75ekLNPxqYcCEpJc-UCBeI-Xf056uS5BY-q4qlBr8"
}
```

## 6. Access protected endpoint with valid token

The screenshot displays the Postman API client interface. At the top, the navigation bar includes 'Home', 'Workspaces', and 'API Network'. The main header shows the request method 'GET' and the URL 'http://127.0.0.1:8000/api/'. Below this, the 'Headers' tab is selected, showing a table with one header: 'Authorization' with a Bearer token. The 'Body' tab is also visible, showing a JSON response. The status bar at the bottom indicates a '200 OK' response with a 6 ms latency and 892 B of data.

GET <http://127.0.0.1:8000/api/>

New Collection / <http://127.0.0.1:8000/api/auth/register/>

GET <http://127.0.0.1:8000/api/expenses/> Send

Params Authorization Headers (9) Body Scripts Tests Settings Cookies

Headers 8 hidden

Key	Value	Description
Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6I...	

Body Cookies Headers (10) Test Results 200 OK 6 ms 892 B Save Response

```
{
  "count": 2,
  "next": null,
  "previous": null,
  "results": [
    {
      "id": 9,
      "total": 1050.0,
      "title": "Fuel",
      "description": "Petrol for bike",
      "amount": "1000.00",
      "transaction_type": "debit",
      "tax": "5.00",
      "tax_type": "percentage",
      "created_at": "2025-07-04T19:41:06.944484Z",
      "updated_at": "2025-07-04T19:41:06.944501Z",
      "user": 5
    }
  ]
}
```

## 7. Access protected endpoint without token

The screenshot shows the Postman interface with a GET request to `http://127.0.0.1:8000/api/expenses/`. The request is in the 'Headers' tab, and the response is in the 'Body' tab. The response status is **401 Unauthorized**, with a message: `"detail": "Authentication credentials were not provided."`

**Request Details:**

- Method: GET
- URL: `http://127.0.0.1:8000/api/expenses/`
- Headers (8 hidden):

Key	Value	Description
Key	Value	Description

**Response Details:**

- Status: 401 Unauthorized
- Time: 4 ms
- Size: 427 B
- Body (JSON):

```
1 {
2   "detail": "Authentication credentials were not provided."
3 }
```

## CRUD Operations Tests

## 1. Create expense/income record

The screenshot displays the Postman interface for an API request. The top navigation bar includes 'Home', 'Workspaces', and 'API Network'. The main header shows the request method 'POST' and the URL 'http://127.0.0.1:8000/api/expenses/'. The 'Body' tab is selected, showing a JSON payload. The status bar at the bottom indicates a '201 Created' response with a 115 ms duration and 608 B size. The response body is also shown in JSON format.

**Request:**

```
POST http://127.0.0.1:8000/api/expenses/

{
  "title": "Chillax with friends",
  "description": "Ktaharule daruma paisa udaye",
  "amount": 25000.0,
  "transaction_type": "debit",
  "tax": 25.0,
  "tax_type": "percentage"
}
```

**Response:**

```
{
  "id": 11,
  "total": 31250.0,
  "title": "Chillax with friends",
  "description": "Ktaharule daruma paisa udaye",
  "amount": "25000.00",
  "transaction_type": "debit",
  "tax": "25.00",
  "tax_type": "percentage",
  "created_at": "2025-07-04T19:48:33.889753Z",
  "updated_at": "2025-07-04T19:48:33.889767Z",
  "user": 5
}
```



## 2. List user's own records only

The screenshot shows the Postman API client interface. At the top, the navigation bar includes 'Home', 'Workspaces', and 'API Network'. The main header displays the request method 'GET' and the URL 'http://127.0.0.1:8000/api/'. Below this, a collection name 'New Collection / http://127.0.0.1:8000/api/auth/register/' is shown. The request body is set to 'raw' and contains a JSON object:

```
{
  "title": "Chillax with friends",
  "description": "Ktaharule daruma paisa udaye",
  "amount": 25000.0,
  "transaction_type": "debit",
  "tax": 25.0,
  "tax_type": "percentage"
}
```

The response status is '200 OK' with a response time of '6 ms' and a size of '1.15 KB'. The response body is displayed in JSON format, showing a list of expense records:

```
[
  {
    "id": 10,
    "total": 187500.0,
    "title": "Computer Parts",
    "description": "Bought GPU, CPU, RAM",
    "amount": "150000.00",
    "transaction_type": "debit",
    "tax": "25.00",
    "tax_type": "percentage",
    "created_at": "2025-07-04T19:41:51.255399Z",
    "updated_at": "2025-07-04T19:41:51.255413Z",
    "user": 5
  },
  {
    "id": 11,
    "total": 31250.0,
    "title": "Chillax with friends",
    "description": "Ktaharule daruma paisa udaye",
    "amount": 25000.0,
    "transaction_type": "debit",
    "tax": 25.0,
    "tax_type": "percentage",
    "created_at": "2025-07-04T19:41:51.255399Z",
    "updated_at": "2025-07-04T19:41:51.255413Z",
    "user": 5
  }
]
```

### 3. Retrieve specific record (own only)

The screenshot shows a REST client interface with a GET request to `http://127.0.0.1:8000/api/expenses/9/`. The request is sent, and the response is displayed in the 'Body' tab. The response is a JSON object representing an expense record with the following details:

- `id`: 9
- `total`: 1050.0
- `title`: "Fuel"
- `description`: "Petrol for bike"
- `amount`: "1000.00"
- `transaction_type`: "debit"
- `tax`: "5.00"
- `tax_type`: "percentage"
- `created_at`: "2025-07-04T19:41:06.944484Z"
- `updated_at`: "2025-07-04T19:41:06.944501Z"
- `user`: 5

```
1 {
2   "id": 9,
3   "total": 1050.0,
4   "title": "Fuel",
5   "description": "Petrol for bike",
6   "amount": "1000.00",
7   "transaction_type": "debit",
8   "tax": "5.00",
9   "tax_type": "percentage",
10  "created_at": "2025-07-04T19:41:06.944484Z",
11  "updated_at": "2025-07-04T19:41:06.944501Z",
12  "user": 5
13 }
```

#### 4. Update existing record (own only)

```
{
  "id": 11,
  "total": 31250.0,
  "title": "Chillax with friends",
  "description": "Ktaharule daruma paisa udaye",
  "amount": "25000.00",
  "transaction_type": "debit",
  "tax": "25.00",
  "tax_type": "percentage",
  "created_at": "2025-07-04T19:48:33.889753Z",
  "updated_at": "2025-07-04T19:48:33.889767Z",
  "user": 5
}
```

The screenshot displays a REST client interface with the following details:

- Request Method:** PUT
- URL:** http://127.0.0.1:8000/api/expenses/11/
- Body:** A JSON object representing an expense record:

```
1 {
2   "title": "Jindagi sarai ramailo",
3   "description": "Ktaharu",
4   "amount": 5000.0,
5   "transaction_type": "debit",
6   "tax": 25.0,
7   "tax_type": "percentage"
8 }
```
- Response:** 200 OK, 95 ms, 595 B. The response body is a JSON object:

```
1 {
2   "id": 11,
3   "total": 6250.0,
4   "title": "Jindagi sarai ramailo",
5   "description": "Ktaharu",
6   "amount": "5000.00",
7   "transaction_type": "debit",
8   "tax": "25.00",
9   "tax_type": "percentage",
10  "created_at": "2025-07-04T19:48:33.889753Z",
11  "updated_at": "2025-07-04T19:56:48.893736Z",
12  "user": 5
13 }
```

5. Delete record (own only)

The screenshot shows the Postman application interface. At the top, the request method is set to **DELETE** and the URL is `http://127.0.0.1:8000/api/expenses/11/`. The environment is set to **No environment**. Below the URL bar, there are tabs for **Params**, **Authorization**, **Headers (7)**, **Body** (selected), **Scripts**, **Tests**, and **Settings**. The **Body** tab shows the content type set to **raw**. The main area contains a single request labeled **1** with the instruction `Ctrl+Alt+P for Postbot`. At the bottom, the response status is **204 No Content**, with a response time of **101 ms** and a size of **310 B**. The response body is empty, and the **Raw** tab is selected.

6. Verify superuser can access all records

The screenshot displays a REST client interface with the following details:

- Request:** A POST request to `http://127.0.0.1:8000/api/auth/login/` with a JSON body: `{ "username": "admin", "password": "admin" }`.
- Response:** A 200 OK status with a response time of 345 ms and a body size of 794 B. The response is in JSON format and contains two tokens: `"refresh"` and `"access"`, each followed by a long alphanumeric string.

```
{
  "refresh": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2t1b190eXB1IjoicmVmcVzaCIsImV4cCI6MTc1MTc0NTYxMCwiaWF0IjoxNzU5MjEwLCJqdGkiOiI2OGVjMjlmNDU0YmI0N2FhOTA4NDZlOTY3MSIsInVzZXJfaWQiOiJ0b2t1b190eXB1Ij09.eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2t1b190eXB1IjoicmVmcVzaCIsImV4cCI6MTc1MTc0NTYxMCwiaWF0IjoxNzU5MjEwLCJqdGkiOiI2OGVjMjlmNDU0YmI0N2FhOTA4NDZlOTY3MSIsInVzZXJfaWQiOiJ0b2t1b190eXB1Ij09",
  "access": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2t1b190eXB1IjoicmVmcVzaCIsImV4cCI6MTc1MTc0NTYxMCwiaWF0IjoxNzU5MjEwLCJqdGkiOiI2OGVjMjlmNDU0YmI0N2FhOTA4NDZlOTY3MSIsInVzZXJfaWQiOiJ0b2t1b190eXB1Ij09"
}
```

GET http://127.0.0.1:8000/api/

New Collection / http://127.0.0.1:8000/api/auth/register/ Save Share

GET http://127.0.0.1:8000/api/expenses/ Send

Params Authorization Headers (9) Body Scripts Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

Body Cookies Headers (10) Test Results 200 OK 5 ms 892 B Save Response

{ } JSON Preview Visualize

```
1 {
2   "count": 2,
3   "next": null,
4   "previous": null,
5   "results": [
6     {
7       "id": 9,
8       "total": 1050.0,
9       "title": "Fuel",
10      "description": "Petrol for bike",
11      "amount": "1000.00",
12      "transaction_type": "debit",
13      "tax": "5.00",
14      "tax_type": "percentage",
15      "created_at": "2025-07-04T19:41:06.944484Z",
16      "updated_at": "2025-07-04T19:41:06.944501Z",
17      "user": 5
18    },
19    {
20      "id": 10,
21      "total": 187500.0,
22      "title": "Computer Parts",
23      "description": "Bought GPU, CPU, RAM",
24      "amount": "150000.00",
25      "transaction_type": "debit",
26      "tax": "25.00",
27      "tax_type": "percentage",
28      "created_at": "2025-07-04T19:41:51.255399Z",
29      "updated_at": "2025-07-04T19:41:51.255413Z",
30      "user": 5
31    }
32  ]
33 }
```

## Permission Tests

# 1. Regular user cannot access other users' records

The screenshot shows a REST client interface with the following details:

- Request:**
  - Method: **POST**
  - URL: `http://127.0.0.1:8000/api/auth/login/`
  - Body (raw):

```
1 {
2   "username": "ameet",
3   "password": "ameet"
4 }
```
- Response:**
  - Status: **200 OK**
  - Time: 349 ms
  - Size: 794 B
  - Body (JSON):

```
1 {
2   "refresh": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2t1b190eXB1IjoicmVmcVzaCIsImV4cCI6MTc1MTc0ODU2MywiaWF0IjoxNzUxNjYyMTYzLCJqdGkiOiIwNDczNzY4Y2Q3OTY0NzRhYWNiOTA1NDJmM2ViYWQzNiIsInVzZXJfaWQiOiJvV9.fIo-GLPNdLH-UhIkRZoHuN-mH0Cvpg4_cW9HFVF9rrk",
3   "access": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2t1b190eXB1IjoiiYWNjZXNzIiwiaXhwIjoxNzUxNjYyNDYzLCJpYXQiOiJlE3NTE2NjIxNjMsImp0aSI6ImIzNTA1YjYzNzUzZTQzMzc4YWE2YTliYjc4MjEyYTQ1IiwidXNlc19pZCI6NX0.v8joBRbVjHe-CmuRwXZbuJwqX6mLCb5Fi1zGooKz7Wo"
4 }
```



GET http://127.0.0.1:8000/api/ • + No environment

New Collection / http://127.0.0.1:8000/api/auth/register/ Save Share

GET http://127.0.0.1:8000/api/expenses/ Send

Params Authorization Headers (9) Body Scripts Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
1 {
2   "username": "ameet",
3   "password": "ameet"
4 }
```

Body Cookies Headers (10) Test Results 200 OK • 6 ms • 892 B Save Response

{ } JSON Preview Visualize

```
1 {
2   "count": 2,
3   "next": null,
4   "previous": null,
5   "results": [
6     {
7       "id": 9,
8       "total": 1050.0,
9       "title": "Fuel",
10      "description": "Petrol for bike",
11      "amount": "1000.00",
12      "transaction_type": "debit",
13      "tax": "5.00",
14      "tax_type": "percentage",
15      "created_at": "2025-07-04T19:41:06.944484Z",
16      "updated_at": "2025-07-04T19:41:06.944501Z",
17      "user": 5
18     },
19     {
20       "id": 10,
21       "total": 187500.0,
22       "title": "Computer Parts",
23       "description": "Bought GPU, CPU, RAM",
24       "amount": "150000.00",
25       "transaction_type": "debit",
26       "tax": "25.00",
27       "tax_type": "percentage",
28       "created_at": "2025-07-04T19:41:51.255399Z",
29       "updated_at": "2025-07-04T19:41:51.255413Z",
30       "user": 5
31     }
32   ]
33 }
```



GET http://127.0.0.1:8000/ap. + No environment

New Collection / http://127.0.0.1:8000/api/auth/register/ Save Share

GET http://127.0.0.1:8000/api/expenses/10/ Send

Params Authorization Headers (9) Body Scripts Tests Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON Beautify

```
1 {
2   "username": "ashish",
3   "password": "ashish"
4 }
```

Body Cookies Headers (10) Test Results 403 Forbidden 5 ms 405 B Save Response

{ } JSON Preview Visualize

```
1 {
2   "detail": "You do not have permission to perform this action."
3 }
```

## 2. Superuser can access all records

The screenshot shows a REST client interface with the following details:

- Request:** GET `http://127.0.0.1:8000/api/expenses/`
- Body:**

```
{
  "username": "admin",
  "password": "admin"
}
```
- Response:** 200 OK, 9 ms, 2.83 KB
- JSON Response:**

```
{
  "count": 10,
  "next": null,
  "previous": null,
  "results": [
    {
      "id": 1,
      "total": 6010.0,
      "title": "Dadama mastii",
      "description": "Bought rice and vegetables",
      "amount": "6000.00",
      "transaction_type": "credit",
      "tax": "10.00",
      "tax_type": "flat",
      "created_at": "2025-07-04T17:30:49.451915Z",
      "updated_at": "2025-07-04T18:16:32.154687Z",
      "user": 1
    },
    {
      "id": 2,
      "total": 210.0,
      "title": "Fuel",
      "description": "Petrol for bike",
      "amount": "200.00",
      "transaction_type": "debit",
      "tax": "5.00",
      "tax_type": "percentage",
      "created_at": "2025-07-04T17:44:54.835554Z",
      "updated_at": "2025-07-04T17:44:54.835567Z",
      "user": 1
    },
    {
      "id": 3,
      "total": 500.0,

```

GET http://127.0.0.1:8000/api/

New Collection / http://127.0.0.1:8000/api/auth/register/

GET http://127.0.0.1:8000/api/expenses/ Send

Params Authorization Headers (9) Body Scripts Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
1 {
2   "username": "admin",
3   "password": "admin"
4 }
```

Body Cookies Headers (10) Test Results 200 OK 9 ms 2.83 KB Save Response

{ JSON Preview Visualize

```
32 {
33   "id": 3,
34   "total": 500.0,
35   "title": "Freelance Payment",
36   "description": "Client paid",
37   "amount": "500.00",
38   "transaction_type": "credit",
39   "tax": "0.00",
40   "tax_type": "flat",
41   "created_at": "2025-07-04T17:45:19.200315Z",
42   "updated_at": "2025-07-04T17:45:19.200331Z",
43   "user": 1
44 },
45 {
46   "id": 4,
47   "total": 140.0,
48   "title": "Haircut",
49   "description": "Kapal kateko",
50   "amount": "120.00",
51   "transaction_type": "debit",
52   "tax": "20.00",
53   "tax_type": "flat",
54   "created_at": "2025-07-04T18:02:22.172046Z",
55   "updated_at": "2025-07-04T18:02:22.172060Z",
56   "user": 1
57 },
58 {
59   "id": 5,
60   "total": 200.0,
61   "title": "Futsal",
62   "description": "Game hariyo paisa tiriyo",
63   "amount": "200.00",
64   "transaction_type": "debit",
65   "tax": "0.00"
```

### 3. Unauthenticated requests are rejected

The screenshot shows a REST client interface with the following details:

- Request:** GET `http://127.0.0.1:8000/api/expenses/`
- Headers:** 8 hidden
- Response:** 401 Unauthorized, 4 ms, 427 B
- Body:** JSON format showing the error message: `{"detail": "Authentication credentials were not provided."}`

Key	Value	Description
Key	Value	Description

```
1 {
2   "detail": "Authentication credentials were not provided."
3 }
```