

# AIM:Personal Expense Tracker

## Create Expense

### POST /api/expenses

This API is used to add a new expense to the system. It accepts expense details like title, amount, category, and user ID, then saves the record in the database and returns the created expense.

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

- Method:** POST
- URL:** http://localhost:8080/api/expenses
- Body Type:** JSON
- Request Body (JSON):**

```
1 {
2   "title": "Lunch",
3   "amount": 250,
4   "category": "Food",
5   "user_id": 5
6 }
7
```
- Response Body (JSON):**

```
1 {
2   "id": 4,
3   "amount": 250,
4   "category": "Food",
5   "description": "",
6   "date": "",
7   "user_id": 5
8 }
```
- Status:** 200

# Get All Expenses

## GET /api/expenses

This API returns a list of all expenses available in the system. It is mainly used to display expenses in dashboards or reports.

GET

http://localhost:8080/api/expenses

S

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

Query Params

Key	Value
Key	Value

Body

Cookies

Headers (3)

Test Results

Status: 200 OK

Time: 4 ms

Size: 414 B

Save R

Pretty

Raw

Preview

Visualize

JSON

```
1 {
2   {
3     "id": 4,
4     "amount": 250,
5     "category": "Food",
6     "description": "",
7     "date": "",
8     "user_id": 5
9   },
10  {
11    "id": 3,
12    "amount": 300.75,
13    "category": "food",
14    "description": "Dinner at restaurant",
15    "date": "2026-01-12",
16    "user_id": 1
17  },
18  {
19    "id": 2,
20    "amount": 350.75,
21    "category": "food",
22    "description": "Dinner at restaurant",
23    "date": "2026-01-12",
24    "user_id": 1
25  }
26 }
```

# Get Expense by ID

## GET /api/expenses/{id}

This API fetches a single expense using its unique ID. It is useful when viewing the details of a specific expense.

GET

http://localhost:8080/api/expenses/2

S

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

Query Params

	Key	Value
	Key	Value

Body

Cookies

Headers (3)

Test Results

Status: 200 OKTime: 3 msSize: 221 BSave R

Pretty

Raw

Preview

Visualize

JSON

1

2

3

4

5

6

7

8

1

2

3

4

5

6

7

8

"id": 2,

"amount": 350.75,

"category": "food",

"description": "Dinner at restaurant",

"date": "2026-01-12",

"user\_id": 1

# Update Expense

## PUT /api/expenses/{id}

This API updates an existing expense. The expense ID is passed in the URL, and the updated values are sent in the request body.

The screenshot displays a REST client interface for a PUT request. The URL bar shows `http://localhost:8080/api/expenses/2`. The 'Body' tab is selected, showing a JSON payload with the following fields: `"title": "Dinner", "amount": 400, "category": "Food", "user_id": 1`. The status bar at the bottom indicates a successful response with `Status: 200 OK`, `Time: 12 ms`, and `Size: 187 B`. The 'Body' tab is also selected in the response section, showing the updated JSON object: `"id": 2, "amount": 400, "category": "Food", "description": "", "date": "", "user_id": 1`.

```
PUT http://localhost:8080/api/expenses/2

{"title": "Dinner",
 "amount": 400,
 "category": "Food",
 "user_id": 1}

Status: 200 OK Time: 12 ms Size: 187 B Save R

Pretty Raw Preview Visualize JSON {
  "id": 2,
  "amount": 400,
  "category": "Food",
  "description": "",
  "date": "",
  "user_id": 1
}
```

# Delete Expense

## DELETE /api/expenses/{id}

This API deletes an expense based on its ID. Once deleted, the expense is removed from the system and a confirmation message is returned.

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

- Method:** DELETE
- URL:** http://localhost:8080/api/expenses/5
- Body:** A JSON object with the following fields:

```
1 {  
2   "title": "Dinner",  
3   "amount": 400,  
4   "category": "Food",  
5   "user_id": 1  
6 }
```
- Status:** 200 OK
- Time:** 10 ms
- Size:** 151 B
- Response Body:** A JSON object with the following field:

```
1 {  
2   "message": "Expense deleted successfully"  
3 }
```

# Get Expenses by User

## GET /api/expenses/user/{user\_id}

This API retrieves all expenses created by a specific user. It helps in showing user-wise expense history.

GET

http://localhost:8080/api/expenses/user/1

S

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

Query Params

Key	Value
Key	Value

BodyCookiesHeaders (3)Test Results

Status: 200 OKTime: 3 msSize: 223 BSave R

PrettyRawPreviewVisualizeJSON

1

{

2

"id": 3,

3

"amount": 300.75,

4

"category": "food",

5

"description": "Dinner at restaurant",

6

"date": "2026-01-12",

7

"user\_id": 1

8

}

9

10

# Get Expenses by Category

## GET /api/expenses/category/{category}

This API returns all expenses that belong to a particular category, such as Food or Travel. It is commonly used for filtering and analysis.

GET

http://localhost:8080/api/expenses/category/food

S

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

Query Params

Key	Value
Key	Value

Body

Cookies

Headers (3)

Test Results

Status: 200 OK

Time: 11 ms

Size: 223 B

Save R

Pretty

Raw

Preview

Visualize

JSON

```
1 {
2   "id": 3,
3   "amount": 300.75,
4   "category": "food",
5   "description": "Dinner at restaurant",
6   "date": "2026-01-12",
7   "user_id": 1
8 }
9
10
```