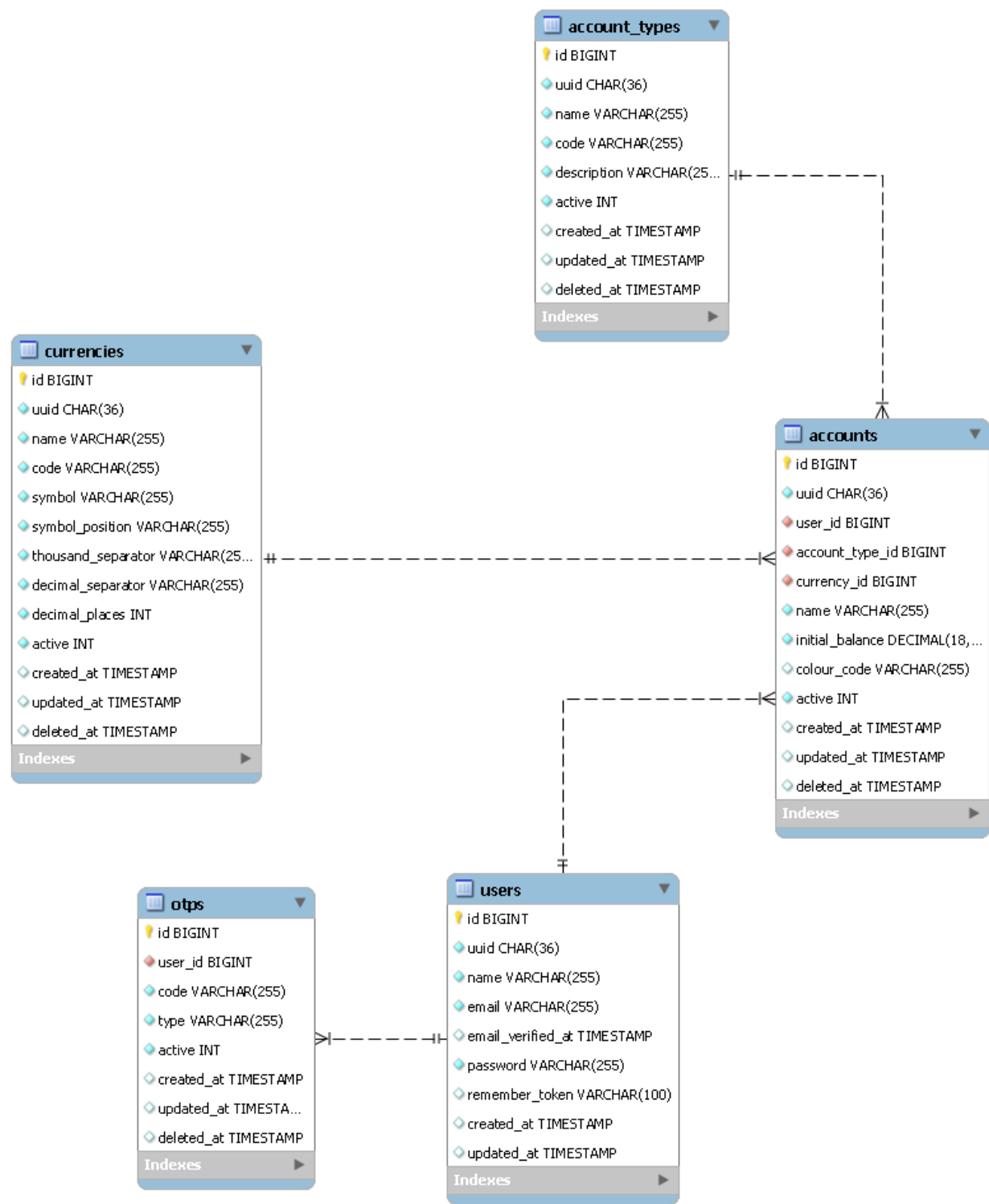


CRUD UI FRAMEWORK – we were working for it

DATABASE SCHEMA DIAGRAM



CREATE ACCOUNT

The screenshot displays the Postman API client interface. The left sidebar shows the 'My Workspace' with a collection named 'Monet API' containing several endpoints. The main panel shows a POST request to 'Monet API / Get Account / Create Accounts' with a 'form-data' body. The response is a 200 OK status with a JSON body indicating successful account creation. The right sidebar shows a PHP code snippet for testing the endpoint.

Request Details:

- Method: POST
- URL: {{API_URL}}/account
- Body: form-data
- Parameters: account_type (Text), 103f9fdd-1dfb-4521-a170...

Response Details:

- Status: 200 OK
- Time: 197 ms
- Size: 670 B

Response Body (JSON):

```
1 {
2   "message": "Account Created Successfully!",
3   "results": {
4     "account": {
5       "id": "ed905dd9-862a-4a50-9900-cc64851e4213",
6       "account_type": {
7         "id": "103f9fdd-1dfb-4521-a170-6d5cuffed174",
8         "name": "Cash",
9         "code": "cash",
10        "description": "Cash Accounts"
11      },
12      "currency": {
13        "id": "0e3f6798-e5ed-4cd4-b9a0-3976562712e7",
14        "name": "Philippine Peso",
15        "code": "PHP",
16        "symbol": "P",
17        "symbol_position": "before",
18        "thousand_separator": ",",
19        "decimal_separator": ".",
20        "decimal_places": 2,
21        "sample": "P 1,000.00"
22      },
23      "name": "Marjovic",
24      "initial_balance": 0,
25      "initial_balance_text": "P 0.00",
26      "current_balance": 0,
27      "current_balance_text": "P 0.00"
28    }
29  }
30 }
```

Code snippet (PHP):

```
1 <?php
2 require_once 'HTTP/Request2.php';
3 $request = new HTTP_Request2();
4 $request->setUri('https://monet.test/api/account');
5 $request->setMethod(HTTP_Request2::METHOD_POST);
6 $request->setConfig(array(
7   'follow_redirects' => TRUE
8 ));
9 $request->setHeader(array(
10  'Accept' => 'application/json',
11  'Authorization' => '*****'
12 ));
13 $request->addPostParameter(array(
14  'account_type' =>
15    '103f9fdd-1dfb-4521-a170-6d5cuffed174',
16  'currency' =>
17    '0e3f6798-e5ed-4cd4-b9a0-3976562712e7',
18  'name' => 'Marjovic',
19  'initial_balance' => '0',
20  'colour_code' => ''
21 ));
22 try {
23   $response = $request->send();
24   if ($response->getStatus() == 200) {
25     echo $response->getBody();
26   }
27   else {
28     echo 'Unexpected HTTP status: ' .
29       $response->getStatus() . ' .
30       $response->getReasonPhrase();
31   }
32 }
33 catch(HTTP_Request2_Exception $e) {
34   echo 'Error: ' . $e->getMessage();
35 }
```

READ ACCOUNT

The screenshot displays the Postman API client interface. The left sidebar shows a collection named 'Monet API' with several endpoints. The main panel shows a GET request to `{{API_URL}}/account` that has been successfully executed, returning a 200 OK status. The response body is displayed in JSON format, showing account details for a 'Savings Account' and currency information for the 'Philippine Peso'.

Request Details:

- Method: GET
- URL: `{{API_URL}}/account`
- Status: 200 OK
- Time: 188 ms
- Size: 677 B

Response Body (JSON):

```
{
  "message": "Accounts Loaded Successfully!",
  "results": {
    "accounts": [
      {
        "id": "ed085dd9-802a-4a50-9908-cc64851e4213",
        "account_type": {
          "id": "49126a8f-34b5-4928-9ede-c873a1325467",
          "name": "Savings Account",
          "code": "savings-account",
          "description": "Savings Accounts"
        },
        "currency": {
          "id": "0e3f6798-e5ed-4cd4-b9a0-3976562712e7",
          "name": "Philippine Peso",
          "code": "PHP",
          "symbol": "P",
          "symbol_position": "before",
          "thousand_separator": ",",
          "decimal_separator": ".",
          "decimal_places": 2,
          "sample": "P 1,000.00"
        },
        "name": "Savings Account",
        "initial_balance": 1500,
        "initial_balance_text": "P 1,500.00",
        "current_balance": 1500,
        "current_balance_text": "P 1,500.00",
        "colour_code": null,
        "active": 1
      }
    ]
  }
}
```

Code Snippet:

```
1 <?php
2 require_once 'HTTP/Request2.php';
3 $request = new HTTP_Request2();
4 $request->setUri('https://monet.test/api/account');
5 $request->setMethod(
6   (HTTP_Request2::METHOD_GET);
7   $request->setConfig(array(
8     'follow_redirects' => TRUE
9   ));
10 $request->setHeader(array(
11   'Accept' => 'application/json',
12   'Authorization' => '.....'
13 ));
14 try {
15   $response = $request->send();
16   if ($response->getStatus() == 200) {
17     echo $response->getBody();
18   }
19   else {
20     echo 'Unexpected HTTP status: ' .
21       $response->getStatus() . ' ' .
22       $response->getReasonPhrase();
23   }
24 } catch (HTTP_Request2_Exception $e) {
25   echo 'Error: ' . $e->getMessage();
26 }
```

UPDATE ACCOUNT

The image shows the Postman application interface. The top bar includes navigation icons, a search bar, and an 'Upgrade' button. The left sidebar shows the 'My Workspace' with a collection named 'Monet API' containing several endpoints. The main panel displays a PATCH request to the URL `{{API_URL}}/account/ed005dd9-862a-4a50-9900-cc64851e4213`. The request body is in 'x-www-form-urlencoded' format. The response is a 200 OK with a JSON body. The right sidebar shows a 'Code snippet' for a PHP script that uses the `HTTP_Request2` library to send the request and handle the response.

Request:

```
PATCH {{API_URL}}/account/ed005dd9-862a-4a50-9900-cc64851e4213
```

Response:

```
200 OK • 165 ms • 674 B
```

```
{
  "message": "Account Updated Successfully!",
  "results": {
    "account": {
      "id": "ed005dd9-862a-4a50-9900-cc64851e4213",
      "account_type": {
        "id": "49126a8f-34b5-4928-9ede-c873a1325467",
        "name": "Savings Account",
        "code": "savings-account",
        "description": "Savings Accounts"
      },
      "currency": {
        "id": "0e3f6798-e5ed-4cd4-b9a0-3976562712e7",
        "name": "Philippine Peso",
        "code": "PHP",
        "symbol": "P",
        "symbol_position": "before",
        "thousand_separator": ",",
        "decimal_separator": ".",
        "decimal_places": 2,
        "sample": "P 1,000.00"
      },
      "name": "Savings Account",
      "initial_balance": 1500,
      "initial_balance_text": "P 1,500.00",
      "current_balance": 1500,
      "current_balance_text": "P 1,500.00",
      "colour_code": null,
      "active": "1"
    }
  }
}
```

Code snippet:

```
PHP - HTTP_Request2
1 <?php
2 require_once 'HTTP/Request2.php';
3 $request = new HTTP_Request2();
4 $request->setUri('https://monet.test/api/
  account/
  ed005dd9-862a-4a50-9900-cc64851e4213');
5 $request->setMethod('PATCH');
6 $request->setConfig(array(
7   'follow_redirects' => TRUE
8 ));
9 $request->setHeader(array(
10  'Accept' => 'application/json',
11  'Content-Type' => 'application/
    x-www-form-urlencoded',
12  'Authorization' => '*****'
13 ));
14 $request->addPostParameter(array(
15  'account_type' =>
16    '49126a8f-34b5-4928-9ede-c873a1325467',
17  'name' => 'Savings Account',
18  'initial_balance' => '1500',
19  'colour_code' => '',
20  'active' => '1'
21 ));
22 $response = $request->send();
23 if ($response->getStatus() == 200) {
24   echo $response->getBody();
25 }
26 else {
27   echo 'Unexpected HTTP status: ' .
    $response->getStatus() . ' .
    $response->getReasonPhrase();
28 }
29 }
30 }
31 catch(HTTP_Request2_Exception $e) {
32   echo 'Error: ' . $e->getMessage();
33 }
```

DELETE ACCOUNT

The screenshot displays the Postman API client interface. The left sidebar shows a collection named 'Monet API' with a sub-collection 'Account Type'. The 'Delete Account by UUID' endpoint is selected. The main panel shows the request details for a DELETE method with the URL `{{API_URL}}/account/ed005dd9-862a-4a50-9900-ccb4851e4213`. The response is a 200 OK status with a body containing a success message.

Request Details:

- Method: DELETE
- URL: `{{API_URL}}/account/ed005dd9-862a-4a50-9900-ccb4851e4213`
- Headers: (11)
- Body: (Scripts, Settings)
- Query Params: (Key, Value, Description)

Response Details:

- Status: 200 OK
- Time: 152 ms
- Size: 362 B
- Body:

```
1 {
2   "message": "Account Deleted Successfully!"
3 }
```

Code Snippet:

```
1 <?php
2 require_once 'HTTP/Request2.php';
3 $request = new HTTP_Request2();
4 $request->setUri('https://monet.test/api/
5   account/
6   ed005dd9-862a-4a50-9900-ccb4851e4213');
7 $request->setMethod(
8   HTTP_Request2::METHOD_DELETE);
9 $request->setConfig(array(
10   'follow_redirects' => TRUE
11 ));
12 $request->setHeader(array(
13   'Accept' => 'application/json',
14   'Content-Type' => 'application/
15   x-www-form-urlencoded',
16   'Authorization' => '.....'
17 ));
18 $request->addPostParameter(array(
19   'account_type' =>
20     '49126a8f-34b5-4928-9ede-c873a1325467',
21   'name' => 'Savings',
22   'initial_balance' => '15000',
23   'colour_code' => ''
24 ));
25 try {
26   $response = $request->send();
27   if ($response->getStatus() == 200) {
28     echo $response->getBody();
29   }
30   else {
31     echo 'Unexpected HTTP status: ' .
32       $response->getStatus() . ' ' .
33       $response->getReasonPhrase();
34   }
35 }
36 catch(HTTP_Request2_Exception $e) {
37   echo 'Error: ' . $e->getMessage();
38 }
```

Database

MySQL Workbench

Local instance MySQL80 x MySQL Model* x EER Diagram x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

monet

monetdb

Tables

- account_types
- accounts
- admins
- cache
- cache_jobs
- currencies
- failed_jobs
- job_batches
- jobs
- migrations
- otps
- password_reset_tokens
- password_resets
- personal_access_tokens
- sessions
- users

Views

Stored Procedures

Functions

Administration Schemas

Information

Schema: monetdb

SQL File 1*

```
1 • use monetdb;  
2 • select * from accounts;  
3  
4
```

Limit to 1000 rows

Result Grid

#	id	uuid	user_id	account_type_id	currency_id	name	initial_balance	colour_code	active	created_at	updated_at	deleted_at
1	ed005d49-862a-4a50-9900-cc64851e4213	24	4	105	Savings Account	1500.00	NULL	1	2025-03-12 04:23:45	2025-03-12 04:46:31	2025-03-12 04:46:31	

accounts 1 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	12:52:54	use monetdb	0 row(s) affected	0.000 sec
2	12:52:54	select * from accounts LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Query Completed

Windows taskbar: 33°C Haze 12:53 PM 3/12/2025

CHALLENGES AND SOLUTIONS

1. Issue: Incorrect API Endpoints

Challenge:

- You may enter the wrong API URL, leading to errors like 404 Not Found.
- Example: Sending requests to /user instead of /api/user.

Solution:

- Double-check the API documentation.
 - Ensure the correct base URL and endpoints are used.
-

2. Issue: Incorrect HTTP Methods

Challenge:

- Using the wrong method (e.g., GET instead of POST).
- Sending a PUT request for creating instead of updating.

Solution:

- Follow CRUD principles:
 - Create → POST
 - Read → GET
 - Update → PUT / PATCH
 - Delete → DELETE
-

3. Issue: Missing or Incorrect Headers

Challenge:

- API requires headers like Content-Type: application/json, but they are missing.
- Authorization headers are required but not included.

Solution:

- Add required headers in Postman under the Headers tab.
 - For authentication, include tokens using Authorization settings.
-

4. Issue: Incorrect JSON Data Format

Challenge:

- Sending malformed JSON causes 400 Bad Request errors.
- Example of incorrect JSON:
- `{ "name": "John, "age": 25 }`

Solution:

- Always validate JSON before sending.
- Use Postman's Prettify feature to format JSON properly.

5. Issue: Database Not Updating

Challenge:

- POST or PUT requests succeed but do not update the database.
- API might return a success response, but changes don't reflect.

Solution:

- Check server logs for errors.
- Ensure API is correctly handling data and committing changes to the database.

6. Issue: CORS Policy Errors

Challenge:

- If testing with a frontend, you might get CORS (Cross-Origin Resource Sharing) issues.

Solution:

- Enable CORS on the backend.
- Use a CORS extension in the browser for local testing.

7. Issue: Authentication & Authorization Errors

Challenge:

- API requires authentication, but token is missing or expired.

Solution:

- Obtain a fresh authentication token.

- Use Postman's Authorization tab to add API keys, JWT tokens, or OAuth credentials.

8. Issue: Unexpected API Response

Challenge:

- The API returns an unexpected response (wrong data or errors).

Solution:

- Debug using Postman's Console (Ctrl + Alt + C) to inspect request details.
- Compare request and response data.

