Creating a Database-Driven API with PHP and MySQL

This tutorial creates a read-only JSON endpoint using PHP, PDO and fictional database records.

An API should return only the data needed for its stated purpose. Do not expose password hashes, session data, personal information or unnecessary database fields.

Create the endpoint

Create `api/events.php`:

```
<?php
declare(strict_types=1);

require __DIR__ . '/../includes/database.php';

header('Content-Type: application/json; charset=utf-8');
header('Cache-Control: no-store');

try {
    $statement = $pdo->query(
        'SELECT
            event_id,
            event_timestamp,
            device_id,
            room,
            event_type,
            severity,
            access_result
         FROM security_events
         ORDER BY event_timestamp DESC
         LIMIT 100'
    );
```

```
    $events = $statement->fetchAll();

    echo json_encode(
        [
            'ok' => true,
            'count' => count($events),
            'events' => $events
        ],
        JSON_THROW_ON_ERROR | JSON_UNESCAPED_SLASHES
    );
} catch (Throwable $error) {
    http_response_code(500);

    echo json_encode([
        'ok' => false,
        'error' => 'The fictional event data is unavailable.'
    ]);
}
```

The endpoint:

- uses the shared PDO connection;
- selects only the required fields;
- limits the response size;
- returns a predictable object; and
- avoids exposing database error details.

Test the response

Open the endpoint through XAMPP:

```
http://localhost/project/api/events.php
```

A successful response should have this shape:

```
{
  "ok": true,
  "count": 2,
  "events": [
```

```
    {
      "event_id": 14,
      "event_timestamp": "2026-06-14 20:57:06",
      "device_id": "CAM-01",
      "room": "Science Lab",
      "event_type": "failed_login",
      "severity": "medium",
      "access_result": "denied"
    }
  ]
}
```


The values are fictional. Your live response will reflect the current database.

Important boundary

This example is a public, read-only classroom endpoint. A production API may also require authentication, authorisation, pagination, rate limiting, audit logging and a documented privacy purpose.

Check

- The endpoint returns valid JSON.
- The response uses an appropriate HTTP status when processing fails.
- No credentials, password hashes or unnecessary fields are exposed.
- The response is limited to a reasonable number of records.
- All records are fictional.

Revision #3
Created 2026-06-10 02:02:11 UTC by Mr Napper
Updated 2026-08-19 23:03:40 UTC by Mr Napper