Repository Project

Lightweight File Data Source Management for PHP

Anonymous · 24/08/2026 · #php

Lightweight File Data Source Management for PHP

A lightweight, flexible PHP library for managing data repositories with multiple data sources and seamless integration with PHP frameworks.

Why Derafu\Repository?

Simplified Data Management

Traditional PHP data repositories often:

- Are tightly coupled to specific storage mechanisms.
- Require complex configuration.
- Lack flexibility across different data sources.

What Makes Derafu\Repository Unique?

Feature	Derafu\Repository	Traditional Repositories
File-Based Data Sources	Yes	No
Lightweight Design	Yes	No
Simple Configuration	Yes	No
Framework Agnostic	Yes	⚠ Varies

Features

- **Multiple Data Source Support** – Load data from PHP arrays, JSON, YAML files.

- **Generic Entity Handling** – Work with any type of data structure.
- **Flexible Querying** – Find, filter, and order data with ease.
- **Doctrine Collections Compatible** – Use Doctrine Criteria for advanced filtering.
- **Zero Heavy Dependencies** – Lightweight and performance-focused.
- **PHP 8+ Ready** – Leverages modern PHP features.

Installation

Install via Composer:

```
composer require derafu/repository
```

Basic Usage

```
use Derafu\Repository\Repository;

// Load data from a PHP file or array.
$data = [
    'products' => [
        'prod-001' => [
            'name' => 'Laptop XPS',
            'category' => 'computers',
            'price' => 1299.99,
        ],
        // More products...
    ],
];

// Create a repository.
$repository = new Repository($data, idAttribute: 'id');

// Find all products.
$allProducts = $repository->findAll();

// Find products by criteria.
$computerProducts = $repository->findBy([
    'category' => 'computers'
]);

// Find a single product.
$laptop = $repository->findOneBy([
    'name' => 'Laptop XPS',
]);
```

Advanced Usage with Doctrine Criteria

```
use Doctrine\Common\Collections\Criteria;
use Doctrine\Common\Collections\Order;

// Advanced filtering with Doctrine Criteria.
$expensiveProducts = $repository->findByCriteria(
    Criteria::create()
        ->where(Criteria::expr()->gt('price', 1000))
        ->orderBy(['price' => Order::Descending])
);
```

Supported Data Sources

- PHP Arrays.
- PHP Files returning arrays.
- JSON Files.
- YAML Files.

Performance Considerations

- Optimized for small to medium-sized datasets.
- In-memory data management.
- Recommended for configuration, lookup tables, and static data.

Last updated on 24/08/2026