



Rapport Sprint 5 :

PHP Mastery : From Fundamentals to Frameworks

Réalisé par :

Sara Akharraz

Encadrant :

M.Abdelmajid Bendrif

M.Hamza Bahlaouane

Entreprise :

Void

Syntaxe, configuration et outils PHP :

Objectifs :

- Acquérir de la pratique concrète en **PHP** et se familiariser avec sa syntaxe et ses particularités.
- Apprendre à créer, configurer et publier un package PHP factice pour se familiariser avec la gestion de projets et **Packagist**.

Ce module m'a permis d'apprendre les principes de base de PHP, de me familiariser avec sa syntaxe, ses fonctions et la structure d'un projet.

La création d'un dummy-package a été pratiquée en parallèle, et ensuite je l'ai publié sur Packagist.

The screenshot shows the Packagist.org interface. At the top, there's a dark header with the text "Packagist.org The PHP Package Repository", "Browse", "Submit", and a user account "saraakharraz". Below the header is a search bar with the placeholder "Search packages...". The main content area displays a package entry for "sara-akharraz/dummy-package". The package name is at the top, followed by a brief description: "Dummy package for testing purposes. Do not use in production.". Below the description are four buttons: "Abandon" (orange), "Delete" (red), "Update" (green), and "Edit" (blue). To the right of the package details, there are sections for "Maintainers" (with a green icon) and "Maintainer actions" (with a plus sign and arrows icon). Further down are sections for "Details", "github.com/saraakharraz/dummy-pac...", "Source", and "Issues". Under "Details", there are metrics: Installs: 0, Dependents: 0, Suggesters: 0, Security: 0, Stars: 0, Watchers: 0, Forks: 0, and Open Issues: 0. A link "pkg:composer/sara-akharraz/dummy-package" is also present.

Le lien vers le dépôt : GitHub Repository

Créer votre propre framework :

Objectifs :

- Apprendre à concevoir et structurer un **framework PHP**, en commençant par un modèle MVC simple.
- Se familiariser avec **les composants Symfony** et vérifier la compréhension en installant et utilisant un package publié sur Packagist.

Construire un framework PHP MVC simple :

Pour mettre en pratique les connaissances acquises lors du premier module, j'ai implémenté un framework PHP from scratch en suivant le tutoriel Write Your Own PHP MVC Framework. L'objectif était de créer une application TODO en respectant l'architecture MVC.

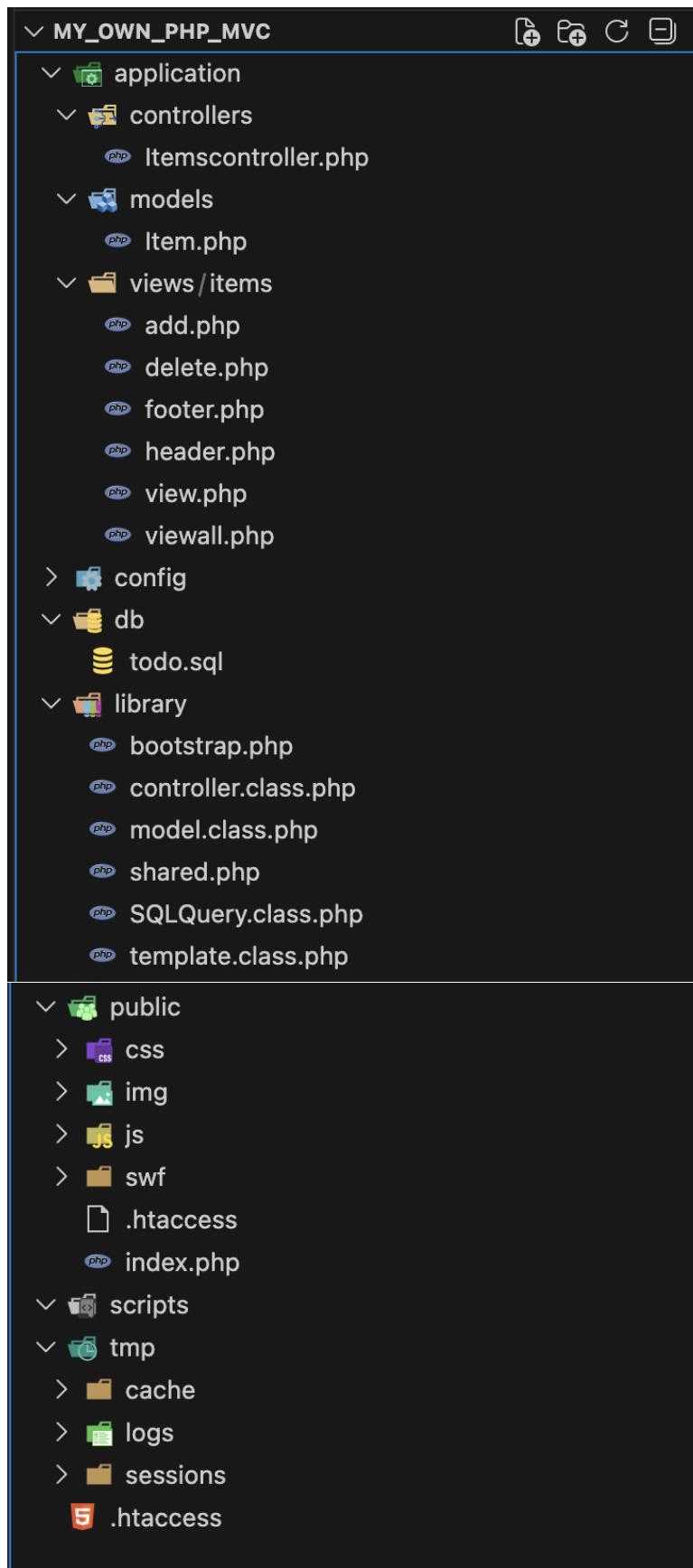


FIGURE 1 – Architecture du projet

Le mode **mod_rewrite** d'Apache permet de réécrire les URL pour les rendre plus lisibles et dirigées vers le bon fichier.

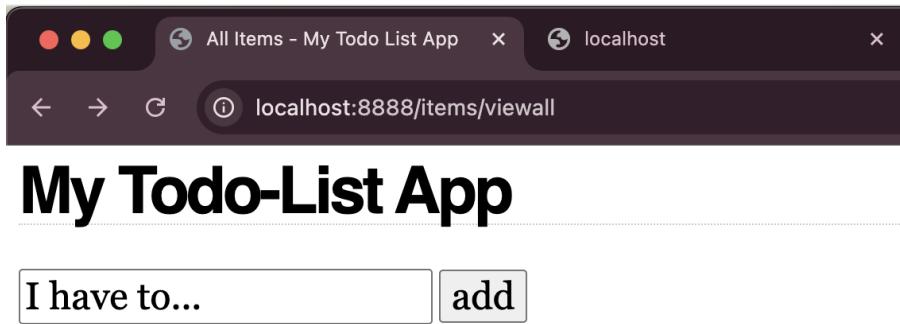
Voici la configuration utilisée dans le fichier **.htaccess** :

```
<IfModule mod_rewrite.c>
```

```
RewriteEngine on  
RewriteRule ^$ public/ [L]  
RewriteRule (.*) public/$1 [L]  
</IfModule>
```

- <IfModule mod_rewrite.c> : Vérifie si le module `mod_rewrite` est activé.
- `RewriteEngine on` : Active le moteur de réécriture d'URL.
- `RewriteRule public/ [L]` : Redirige la racine du site vers le dossier `public/`.
- `RewriteRule (*) public/$1 [L]` : Redirige toutes les autres requêtes vers le dossier `public/`, en conservant le chemin demandé.

Résultat :



- 1 Write your own PHP MVC Framework
- 2 Build a Framework Using Symfony Components
- 3 test
- 4 it works

Le lien vers le dépôt : [GitHub Repository](#)

Construire un framework en utilisant les composants Symfony :

Pour approfondir mes connaissances, j'ai suivi le guide Symfony : Create a PHP Framework Guide. L'objectif était de construire un framework PHP en utilisant les composants Symfony et de comprendre comment organiser un projet complexe tout en réutilisant des bibliothèques existantes.

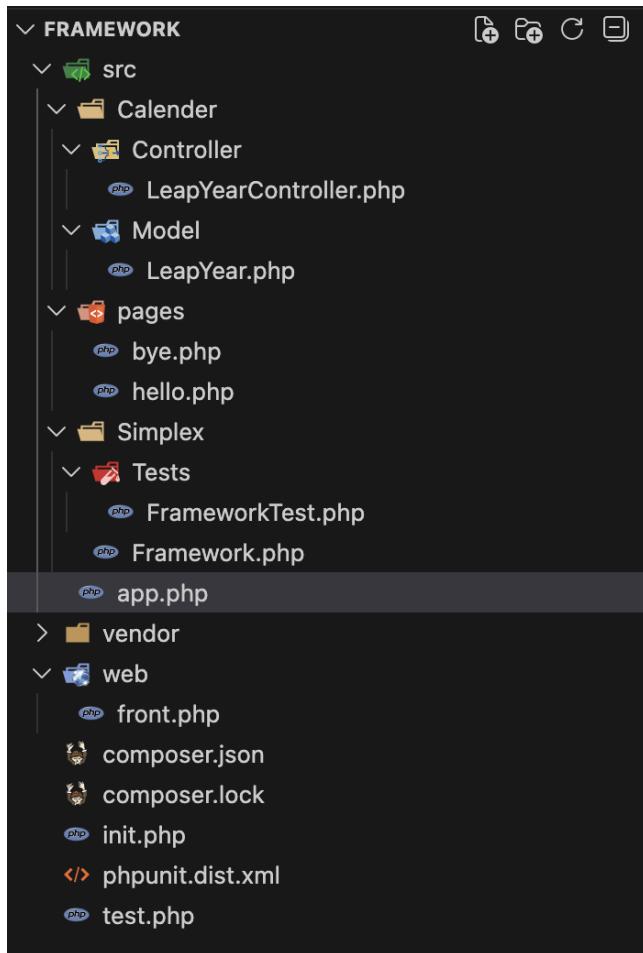


FIGURE 2 – Architecture du projet Symfony

Travailler avec les bases de données et la ligne de commande :

Objectifs :

- Développer une application CLI en PHP pour importer efficacement des fichiers CSV dans une base MySQL.

Pour mettre en pratique mes connaissances en PHP et bases de données, j'ai développé une application CLI pour importer des fichiers CSV dans une base MySQL.

L'application est construite avec le composant Symfony Console et utilise le package `league/csv` pour parser les fichiers.

Elle repose sur PDO et des requêtes SQL brutes, sans ORM, et suit une architecture claire :

- Point d'entrée CLI : `ImportCsvCommand`
- Lecteur de CSV : `CsvReader`
- Validateur de données : `CsvValidator`
- Importeur qui gère les insertions en lots : `CsvImporter`

- **Transactions** : Chaque batch d'insertion s'exécute dans une transaction, garantissant l'atomicité (tout ou rien).
En cas d'erreur, la transaction est annulée avec `rollback`.
- **Insertions en lots** : Les données sont insérées par groupes de lignes (batchs) pour améliorer les performances.
- **Gestion des doublons** : Les doublons sont ignorés avec `INSERT IGNORE`, ce qui évite les interruptions de l'import.
- **Performance** : Optimisation grâce aux colonnes indexées, aux insertions en batch et aux transactions.
- **Export et transfert** : La base peut être exportée avec `mysqldump`, compressée avec `gzip` et transférée via `scp`.

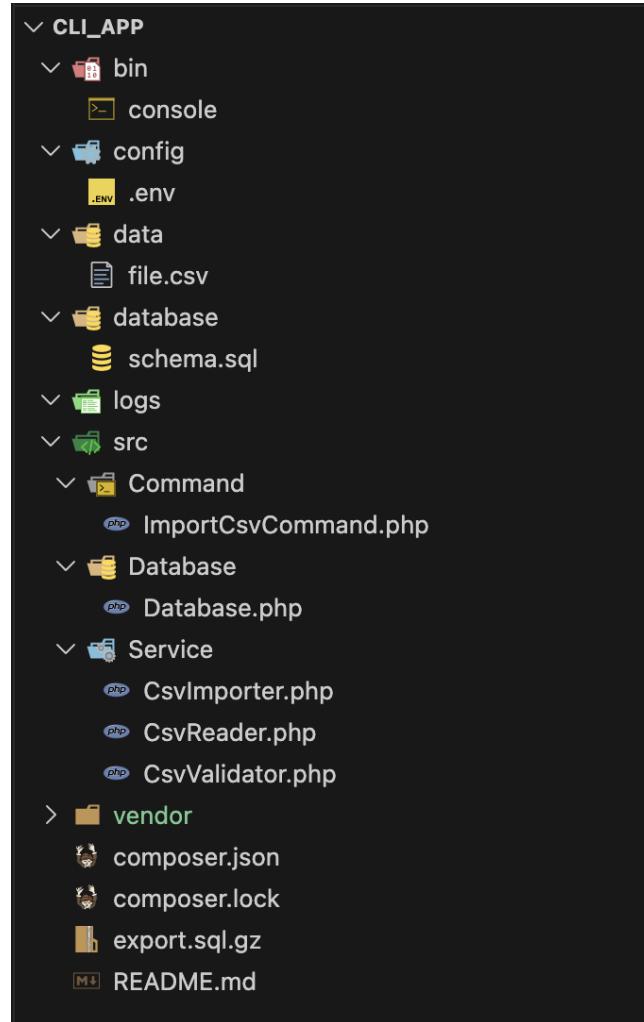


FIGURE 3 – Architecture de l'application CLI d'import CSV

On run via la commande :

```
./bin/console csv:import data/file.csv customers
```

```
Import Results
=====
Total rows: 10000
Imported: 10000
Skipped: 0
Execution time: 1.07 seconds
→ cli_app git:(main) ✘
```

Le lien vers le dépôt : [GitHub Repository](#)

API REST avec Laravel :

Objectifs :

- Mettre en pratique l'utilisation de Laravel pour créer une API REST sécurisée et fonctionnelle, capable de filtrer, trier et paginer des données.

Pour approfondir l'utilisation de PHP et des frameworks modernes, une application Laravel a été mise en place pour exposer une API REST sur la base de données `customers`.

Un schéma JSON d'exemple (*Example Schema*) a été adapté afin de faciliter l'implémentation des filtres.

Plusieurs fonctionnalités ont été mises en oeuvre pour l'API :

- Ajout d'un en-tête personnalisé dans la réponse (`x-api-version: v1`) pour indiquer la version de l'API.
- Mise en place d'une authentification basique pour sécuriser l'accès.
- Interaction avec l'API via Postman et Curl.
- Possibilité de filtrer, trier et paginer les données.

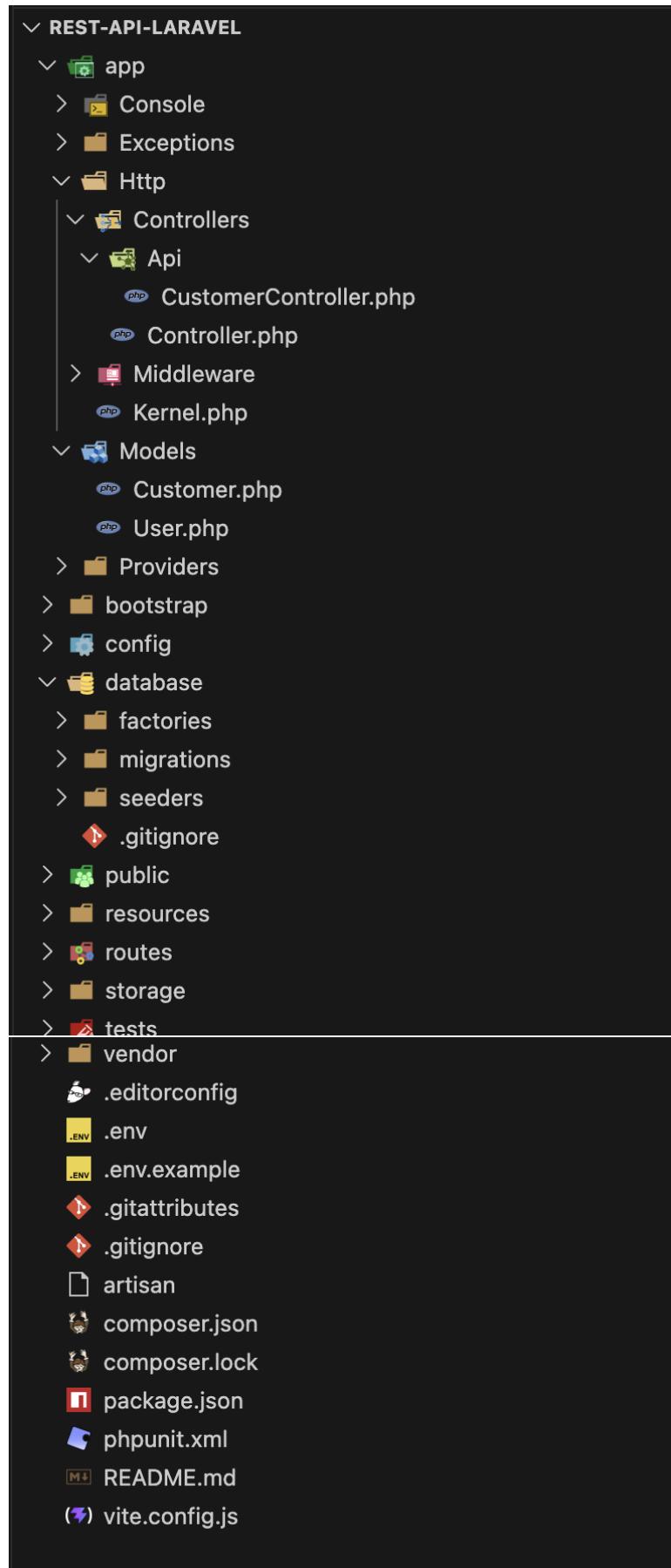


FIGURE 4 – Architecture de l'api rest Laravel

Les Tests : Get All Customers (No Auth) :

The screenshot shows the Postman interface with the following details:

- Method:** GET
- URL:** http://localhost:8000/api/customers
- Status:** 200 OK
- Body (JSON):**

```

1 {
2   "success": true,
3   "message": "Customers retrieved successfully",
4   "data": [
5     {
6       "id": 1,
7       "csv_index": null,
8       "customer_id": "EB54EF1154C3A78",
9       "first_name": "Heather",
10      "last_name": "Callahan",
11      "company": "Mosley-David",
12      "city": "Lake Jeffborough",
13      "country": "Norway",
14      "phone_1": "043-797-5229",
15      "phone_2": "915.112.1727",
16      "email": "urangel@espinoza-francis.net",
17      "subscription_date": "2020-08-26T00:00:00.000000Z",
18      "website": "http://www.escobar.org/",
19      "created_at": "2026-02-20T14:46:08.000000Z",
    ]
  }

```

Filter by First name (Heather) :

```
rest-api-laravel git:(main) ✘ curl -X GET 'http://localhost:8000/api/customers?filter%5Bfirst_name%5D=Heather' \
-H "Accept: application/json"
{"success":true,"message":"Customers retrieved successfully","data":[{"id":1,"cs
v_index":null,"customer_id":"EB54EF1154C3A78","first_name":"Heather","last_name"
:"Callahan","company":"Mosley-David","city":"Lake Jeffborough","country":"Norway
","phone_1":"043-797-5229","phone_2":"915.112.1727","email":"urangel@espinoza-fr
ancis.net","subscription_date":"2020-08-26T00:00:00.000000Z","website":"http://www.esc
obar.org/","created_at":"2026-02-20T14:46:08.000000Z","updated_at":"202
6-02-20T14:46:08.000000Z"}, {"id":1281,"csv_index":null,"customer_id":"3B6FEB4DC7
FA6B2","first_name":"Heather","last_name":"Higgins","company":"Luna-Hampton","ci
ty":"Anthonyview","country":"Sudan","phone_1":"914.966.3356x726","phone_2":"426-
```

Filter by Country (Andorra) :

HTTP My Collection / Get data

GET http://localhost:8000/api/customers?filter%5Bcountry%5D=Andorra

Send

Docs Params Authorization Headers (7) Body Scripts Settings Cookies

Query Params

<input checked="" type="checkbox"/> Key	Value	Description	... Bulk Edit
<input checked="" type="checkbox"/> filter%5Bcountry%5D	Andorra		
Key	Value	Description	

Body Cookies Headers (12) Test Results (1/1) ⏪ 200 OK • 28 ms • 4.89 KB • ⌂ | ⌂ Save Response ⌂

{ } JSON ▾ ▶ Preview ⌂ Visualize | ▾

```

1 {
2   "success": true,
3   "message": "Customers retrieved successfully",
4   "data": [
5     {
6       "id": 2,
7       "csv_index": null,
8       "customer_id": "10dAcafEBbA5FcA",
9       "first_name": "Kristina",
10      "last_name": "Ferrell",
11      "company": "Horn, Shepard and Watson",
12      "city": "Aaronville",
13      "country": "Andorra",
14      "phone_1": "932-062-1802",
15      "phone_2": "(209)172-7124x3651",
16      "email": "xreese@hall-donovan.com",
17      "subscription_date": "2020-04-27T00:00:00.000000Z",
18      "website": "https://tyler-pugh.info/",
19      "created_at": "2026-02-20T14:46:08.000000Z",

```

SORT BY EMAIL (ascending A→Z)

HTTP My Collection / Get data

GET http://localhost:8000/api/customers?sort=email

Send

Docs Params Authorization Headers (7) Body Scripts Settings Cookies

Query Params

<input checked="" type="checkbox"/> Key	Value	Description	... Bulk Edit
<input checked="" type="checkbox"/> sort	email		
Key	Value	Description	

Body Cookies Headers (12) Test Results (1/1) ⏪ 200 OK • 67 ms • 4.85 KB • ⌂ | ⌂ Save Response ⌂

{ } JSON ▾ ▶ Preview ⌂ Visualize | ▾

```

1 {
2   "success": true,
3   "message": "Customers retrieved successfully",
4   "data": [
5     {
6       "id": 3413,
7       "csv_index": null,
8       "customer_id": "3cf87E40a418E42",
9       "first_name": "Connor",
10      "last_name": "Harper",
11      "company": "Payne, Lloyd and Randall",
12      "city": "North Staciemouth",
13      "country": "Swaziland",
14      "phone_1": "+1-361-025-4372x5094",
15      "phone_2": "352-182-2376x8387",
16      "email": "aalvarez@mcclain-mcpherson.com",
17      "subscription_date": "2022-04-23T00:00:00.000000Z",
18      "website": "http://dean.com/",
19      "created_at": "2026-02-20T14:46:09.000000Z",

```

PAGE 2 with 10 items per page :

The screenshot shows a REST API testing interface with the following details:

- HTTP Method:** GET
- URL:** http://localhost:8000/api/customers?page%5Bnumber%5D=2&page%5Bsize%5D=10
- Headers:** (7)
- Body:** (1/1)
- Cookies:** (1/1)
- Test Results:** (1/1) - 200 OK (62 ms, 4.87 KB)
- Response Preview:**

```
166     "country": "Singapore",
167     "phone_1": "(822)795-8754x29384",
168     "phone_2": "(769)638-7026x967",
169     "email": "amy99@booker.com",
170     "subscription_date": "2020-12-30T00:00:00.000000Z",
171     "website": "http://www.larsen-floyd.biz/",
172     "created_at": "2026-02-20T14:46:08.000000Z",
173     "updated_at": "2026-02-20T14:46:08.000000Z"
174   }
175 ],
176   "pagination": [
177     "total": 10000,
178     "per_page": 10,
179     "current_page": 2,
180     "last_page": 1000,
181     "from": 11,
182     "to": 20
183   ]
184 }
```

Filter by country + Sort by email + Pagination :

HTTP My Collection / Get data

GET http://localhost:8000/api/customers?filter%5Bcountry%5D=Norway&sort=-email&page%5Bnumber%5D=1&page%5Bsize%5D=10

Send

Params

Key	Value	Description
sort	-email	
page%5Bnumber%5D	1	
page%5Bsize%5D	10	

Body Cookies Headers (12) Test Results (1/1)

200 OK • 72 ms • 4.8 KB • Save Response

{ } JSON ▾ Preview Visualize | ↻

```
166   [
167     {
168       "country": "Norway",
169       "phone_1": "(937)866-9688x17423",
170       "phone_2": "001-778-967-6161x67221",
171       "email": "regina78@sampson.net",
172       "subscription_date": "2021-11-13T00:00:00.000000Z",
173       "website": "https://www.harding.com/",
174       "created_at": "2026-02-20T14:46:08.000000Z",
175       "updated_at": "2026-02-20T14:46:08.000000Z"
176     },
177     "pagination": {
178       "total": 51,
179       "per_page": 10,
180       "current_page": 1,
181       "last_page": 6,
182       "from": 1,
183       "to": 10
184     }
]
```

Le lien vers le dépôt : GitHub Repository