**Laravel Random User List**

**Overview**

This Laravel project provides a user listing page that fetches data from the Random User API, displays it in a paginated and filterable table, and allows exporting the data to CSV. The application demonstrates best practices in API consumption, caching, pagination, error handling, and responsive UI design using Bootstrap.

PHP – 8.2.12  
after clone the project  
1 .composer install  
2. php artisan key:generate  
3. Php artisan serve.

**HERE I HAVE USED**  
LARAVEL 12 for api creation

Cache are used to minimize api call at a time

Pagination library are used for view 10 data of every page

**League/csv** library for export the data into a csv file.

No Table are used

No Database are used

**To handle the request create a api route file & controller**Route::get('/users', [UserController::class, 'index']);

Route::post('/users/export', [UserController::class, 'export'])->name('users.export');

**To create a controller**  
php artisan make:controller Api\UserController.

**Architecture & Flow**

1. Controller Logic:

- Reads `page`, `gender`, and `export` from the request

- Checks cache for a batch of 50 users (per gender and batch)

- If not cached, fetches from API and stores in cache

- Paginates users in memory (10 per page)

- If `export` is set, streams a CSV download of the current batch

- Passes paginated users, filter state, and error info to the view

2. View:

- Responsive table with Bootstrap styling

- Gender filter dropdown in the table header

- Export to CSV button

- Laravel pagination controls (Bootstrap 4 style)

- Error message display if API fails

**Future Enhancements**

- User Details Modal/Page: Allow clicking a user row to view more detailed information in a modal or separate page.

- Multiple Filters: Add more filters (e.g., nationality, age range, search by name/email).

- Bulk Export: Option to export all filtered users (not just current page) to CSV or other formats (Excel, PDF).

- User Image/Avatar: Display user profile pictures in the table.

- API Rate Limiting Handling: Show a specific message or retry logic if the Random User API rate limit is hit.

- Frontend Framework Integration: Use Vue.js or React for a more dynamic, SPA-like experience.

- User Authentication: Restrict access to the user list/export to authenticated users only.