1. function groupAnagrams(array $words)

{

$result = [];

$groupedWords = [];

foreach ($words as $word) {

$sortedWord = str\_split($word);

sort($sortedWord);

$sortedWord = implode('', $sortedWord);

if (!array\_key\_exists($sortedWord, $groupedWords)) {

$groupedWords[$sortedWord] = [];

}

$groupedWords[$sortedWord][] = $word;

}

foreach ($groupedWords as $group) {

$result[] = $group;

}

return $result;

}

$words = ['kita', 'atik', 'tika', 'aku', 'kia', 'makan', 'kua'];

$result = groupAnagrams($words);

print\_r($result);

1. a.Pagination

We can use the built-in pagination functionality of Laravel to achieve this.

public function index(Request $request)

{

$perPage = $request->query('per\_page', 10);

$page = $request->query('page', 1);

$characters = Character::paginate($perPage, ['\*'], 'page', $page);

return response()->json($characters);

}

b.Sorting:

We can sort the data by adding the 'orderBy' clause in the query builder

public function index(Request $request)

{

$perPage = $request->query('per\_page', 10);

$page = $request->query('page', 1);

$sort = $request->query('sort', 'name');

$order = $request->query('order', 'asc');

$characters = Character::orderBy($sort, $order)->paginate($perPage, ['\*'], 'page', $page);

return response()->json($characters);

}

C. Searching by keyword:

We can search the data by adding the 'where' clause in the query builder.

public function index(Request $request)

{

$perPage = $request->query('per\_page', 10);

$page = $request->query('page', 1);

$sort = $request->query('sort', 'name');

$order = $request->query('order', 'asc');

$keyword = $request->query('keyword');

$characters = Character::when($keyword, function ($query) use ($keyword) {

return $query->where('name', 'like', "%$keyword%");

})->orderBy($sort, $order)->paginate($perPage, ['\*'], 'page', $page);

return response()->json($characters);

}

D. Implement Headers:

We can add headers for security to add a level of security to our RESTful API. For example, you can add an 'Authorization' header to verify the API key.

public function index(Request $request)

{

$perPage = $request->query('per\_page', 10);

$page = $request->query('page', 1);

$sort = $request->query('sort', 'name');

$order = $request->query('order', 'asc');

$keyword = $request->query('keyword');

$apiKey = $request->header('Authorization');

if ($apiKey !== env('API\_KEY')) {

return response()->json(['error' => 'Unauthorized'], 401);

}

$characters = Character::when($keyword, function ($query) use ($keyword) {

return $query->where('name', 'like', "%$keyword%");

})->orderBy($sort, $order)->paginate($perPage, ['\*'], 'page', $page);

return response()->json($characters);

}

``

1. <?php

function display\_pattern($N) {

for ($i = 1; $i <= $N; $i++) {

for ($j = $N; $j >= $i; $j--) {

echo "&nbsp;";

}

for ($k = 1; $k <= $i; $k++) {

echo $k;

}

echo "<br>";

}

}

$N = 3;

display\_pattern($N);

Output:

1

2 1

3 2 1