

First install Laravel Command Using Command: `composer global require laravel/installer`

```
C:\Users\jaffar>composer global require laravel/installer
Changed current directory to C:/Users/jaffar/AppData/Roaming/Composer
./composer.json has been updated
Running composer update laravel/installer
Loading composer repositories with package information
Updating dependencies
Lock file operations: 0 installs, 1 update, 0 removals
  - Upgrading laravel/installer (v4.0.7 => v4.5.1)
Writing lock file
Installing dependencies from lock file (including require-dev)
Package operations: 0 installs, 1 update, 0 removals
  - Downloading laravel/installer (v4.5.1)
  - Upgrading laravel/installer (v4.0.7 => v4.5.1): Extracting archive
Generating autoload files
10 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
No security vulnerability advisories found
Using version ^4.5 for laravel/installer
```

To Make New Project: `laravel new project-name-here:`

- Ex: `laravel new realtime-laravel-basics`

```
G:\Web\Laravel>laravel new realtime-laravel-basics

Laravel

Creating a "laravel/laravel" project at "./realtime-laravel-basics"
Installing laravel/laravel (v11.1.4)
  - Installing laravel/laravel (v11.1.4): Extracting archive
Created project in G:\Web\Laravel\realtime-laravel-basics
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
Updating dependencies
Lock file operations: 107 installs, 0 updates, 0 removals
  - Locking brick/math (0.12.1)
  - Locking carbonphp/carbon-doctrine-types (3.2.0)
  - Locking dflydev/dot-access-data (v3.0.3)
  - Locking doctrine/inflector (2.0.10)
  - Locking doctrine/lexer (3.0.1)
  - Locking dragonmantank/cron-expression (v3.3.3)
  - Locking egulias/email-validator (4.0.2)
  - Locking fakerphp/faker (v1.23.1)
  - Locking filp/whoops (2.15.4)
  - Locking fruitcake/php-cors (v1.3.0)
  - Locking graham-campbell/result-type (v1.1.3)
  - Locking guzzlehttp/guzzle (7.9.1)
  - Locking guzzlehttp/promises (2.0.3)
  - Locking guzzlehttp/psr7 (2.7.0)
  - Locking guzzlehttp/uri-template (v1.0.3)
  - Locking hamcrest/hamcrest-php (v2.0.1)
  - Locking laravel/framework (v11.16.0)
  - Locking laravel/pint (v1.16.2)
  - Locking laravel/prompts (v0.1.24)
  - Locking laravel/sail (v1.30.2)
  - Locking laravel/serializable-closure (v1.3.3)
  - Locking laravel/tinker (v2.9.0)
  - Locking league/commonmark (2.4.2)
  - Locking league/config (v1.2.0)
  - Locking league/flysystem (3.28.0)
  - Locking league/flysystem-local (3.28.0)
  - Locking league/mime-type-detection (1.15.0)
  - Locking mockery/mockery (1.6.12)
  - Locking monolog/monolog (3.7.0)
  - Locking myclabs/deep-copy (1.12.0)
  - Locking nesbot/carbon (3.7.0)
```

```

90/97 [=====-->] 92%
97/97 [=====] 100%
59 package suggestions were added by new dependencies, use `composer suggest` to see details.
Generating optimized autoload files
> Illuminate\Foundation\ComposerScripts::postAutoloadDump
> @php artisan package:discover --ansi

[INFO] Discovering packages.

laravel/sail ..... DONE
laravel/tinker ..... DONE
nesbot/carbon ..... DONE
nunomaduro/collision ..... DONE
nunomaduro/termwind ..... DONE

79 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
> @php artisan vendor:publish --tag=laravel-assets --ansi --force

[INFO] No publishable resources for tag [laravel-assets].

No security vulnerability advisories found
> @php artisan key:generate --ansi

[INFO] Application key set successfully.

> @php -r "file_exists('database/database.sqlite') || touch('database/database.sqlite');"
> @php artisan migrate --graceful --ansi

[INFO] Preparing database.

Creating migration table ..... 111.85ms DONE

[INFO] Running migrations.

0001_01_01_000000_create_users_table ..... 655.07ms DONE
0001_01_01_000001_create_cache_table ..... 399.19ms DONE
0001_01_01_000002_create_jobs_table ..... 874.25ms DONE

[INFO] Application ready! Build something amazing.

```

Then We Can Install Any Starter Kit, like breeze: `composer require laravel/breeze --dev`

```

PS G:\Web\Laravel\realtime-laravel-basics> composer require laravel/breeze --dev
./composer.json has been updated
Running composer update laravel/breeze
Loading composer repositories with package information
Updating dependencies
Lock file operations: 1 install, 0 updates, 0 removals
- Locking laravel/breeze (v2.1.2)
Writing lock file
Installing dependencies from lock file (including require-dev)
Package operations: 1 install, 0 updates, 0 removals
- Downloading laravel/breeze (v2.1.2)
- Installing laravel/breeze (v2.1.2): Extracting archive
Generating optimized autoload files
> Illuminate\Foundation\ComposerScripts::postAutoloadDump
> @php artisan package:discover --ansi

[INFO] Discovering packages.

laravel/breeze ..... DONE
laravel/sail ..... DONE
laravel/tinker ..... DONE
nesbot/carbon ..... DONE
nunomaduro/collision ..... DONE
nunomaduro/termwind ..... DONE

79 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
> @php artisan vendor:publish --tag=laravel-assets --ansi --force

[INFO] No publishable resources for tag [laravel-assets].

No security vulnerability advisories found
Using version ^2.1 for laravel/breeze
PS G:\Web\Laravel\realtime-laravel-basics>

```

Then, From CMD Of The Project, We Run: **php artisan breeze:install**

- We Choose: **Blade With Alpine.**
- Then We Set: **Pest As Unit Tester.**

```
PS G:\Web\Laravel\realtime-laravel-basics> php artisan breeze:install

Which Breeze stack would you like to install?
Blade with Alpine ..... blade
Livewire (Volt Class API) with Alpine ..... livewire
Livewire (Volt Functional API) with Alpine ..... livewire-functional
React with Inertia ..... react
Vue with Inertia ..... vue
API only ..... api
> blade

Would you like dark mode support? (yes/no) [no]
> no

Which testing framework do you prefer? [PHPUnit]
Pest ..... 0
PHPUnit ..... 1
> 0

./composer.json has been updated
Running composer update phpunit/phpunit
Loading composer repositories with package information
Updating dependencies
Lock file operations: 0 installs, 0 updates, 25 removals
```

To Install Broadcasting Capabilities For Laravel, We Run:

- Command: **php artisan install:broadcasting**
- Then We Choose: **Install Reverb** (Yes).
- We Choose Also, To Install NodeJS Libraries And Build Them.

```
PS G:\Web\Laravel\realtime-laravel-basics> php artisan install:broadcasting

[INFO] Published 'broadcasting' configuration file.
[INFO] Published 'channels' route file.

Would you like to install Laravel Reverb? (yes/no) [yes]
> yes

./composer.json has been updated
Running composer update laravel/reverb
Loading composer repositories with package information
Updating dependencies
Lock file operations: 15 installs, 0 updates, 0 removals
- Locking clue/redis-protocol (v0.3.1)
- Locking clue/redis-react (v2.7.0)
```

Then We Activate These Chooses For DB (MySQL):

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=realtime_laravel_basics
DB_USERNAME=root
DB_PASSWORD=
```

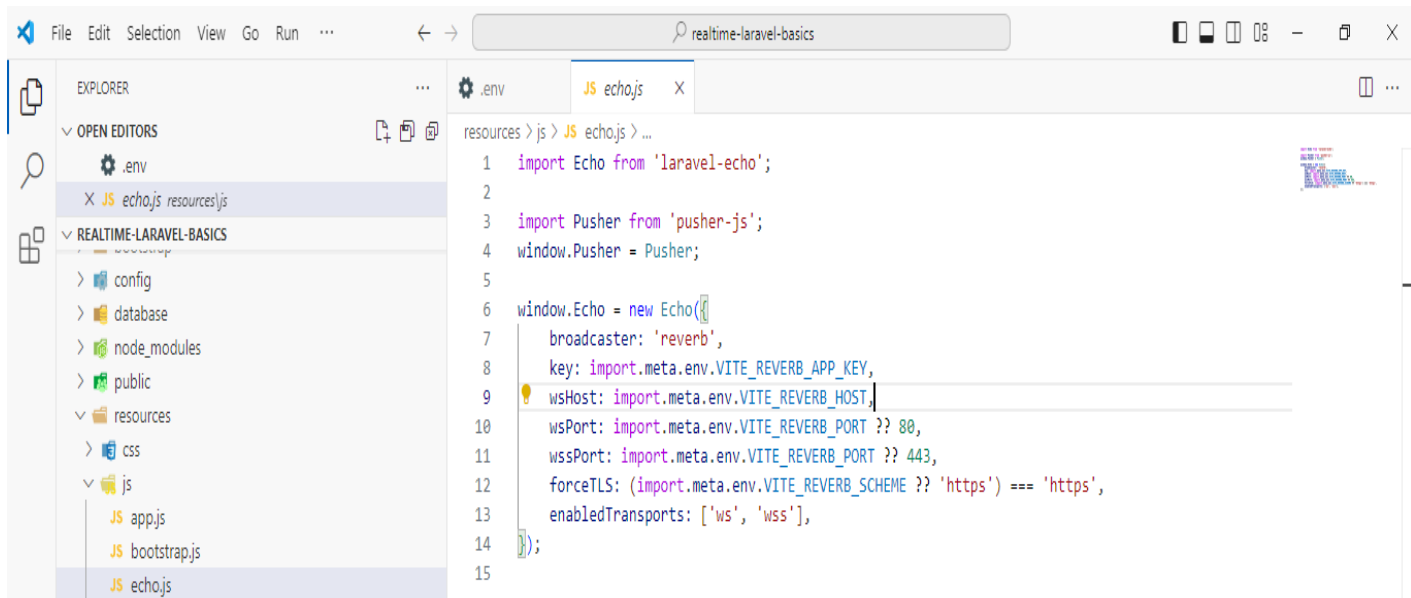
After Installing Reverb, These Options Will Be Changed Like That:

```
BROADCAST_CONNECTION=reverb
FILESYSTEM_DISK=local
QUEUE_CONNECTION=database

REVERB_APP_ID=856430
REVERB_APP_KEY=6boqtzhvibiv8gzltevh
REVERB_APP_SECRET=2vwyfawcw4rltyjcksty
REVERB_HOST="localhost"
REVERB_PORT=8080
REVERB_SCHEME=http
```

In The Production Environment, The Previous Options May Changed, especially, The **REVERB_HOST**.

Also, The JS Files, Will Be Changed, To Import Echo Server With Configuration:

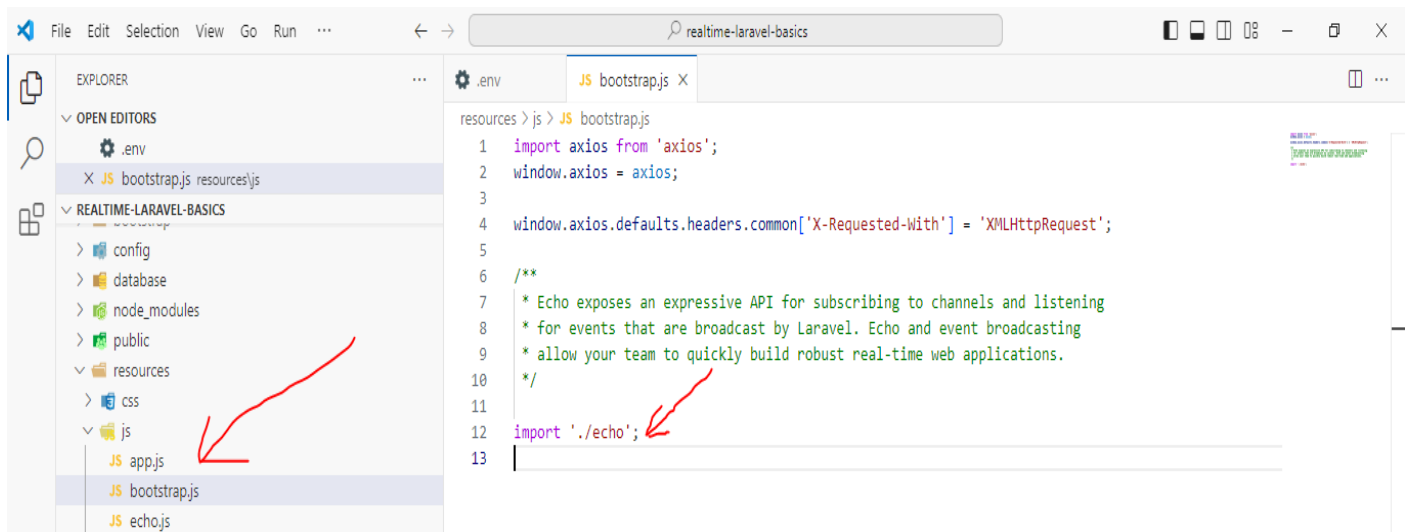


This screenshot shows the VS Code editor with the file explorer on the left and the editor window on the right. The file explorer shows the project structure with the following folders and files:

- resources
 - js
 - app.js
 - bootstrap.js
 - echo.js

The editor window shows the contents of `resources > js > echo.js`. The code is as follows:

```
1 import Echo from 'laravel-echo';
2
3 import Pusher from 'pusher-js';
4 window.Pusher = Pusher;
5
6 window.Echo = new Echo({
7     broadcaster: 'reverb',
8     key: import.meta.env.VITE_REVERB_APP_KEY,
9     wsHost: import.meta.env.VITE_REVERB_HOST,
10    wsPort: import.meta.env.VITE_REVERB_PORT ?? 80,
11    wssPort: import.meta.env.VITE_REVERB_PORT ?? 443,
12    forceTLS: (import.meta.env.VITE_REVERB_SCHEME ?? 'https') === 'https',
13    enabledTransports: ['ws', 'wss'],
14 });
15
```



This screenshot shows the VS Code editor with the file explorer on the left and the editor window on the right. The file explorer shows the project structure with the following folders and files:

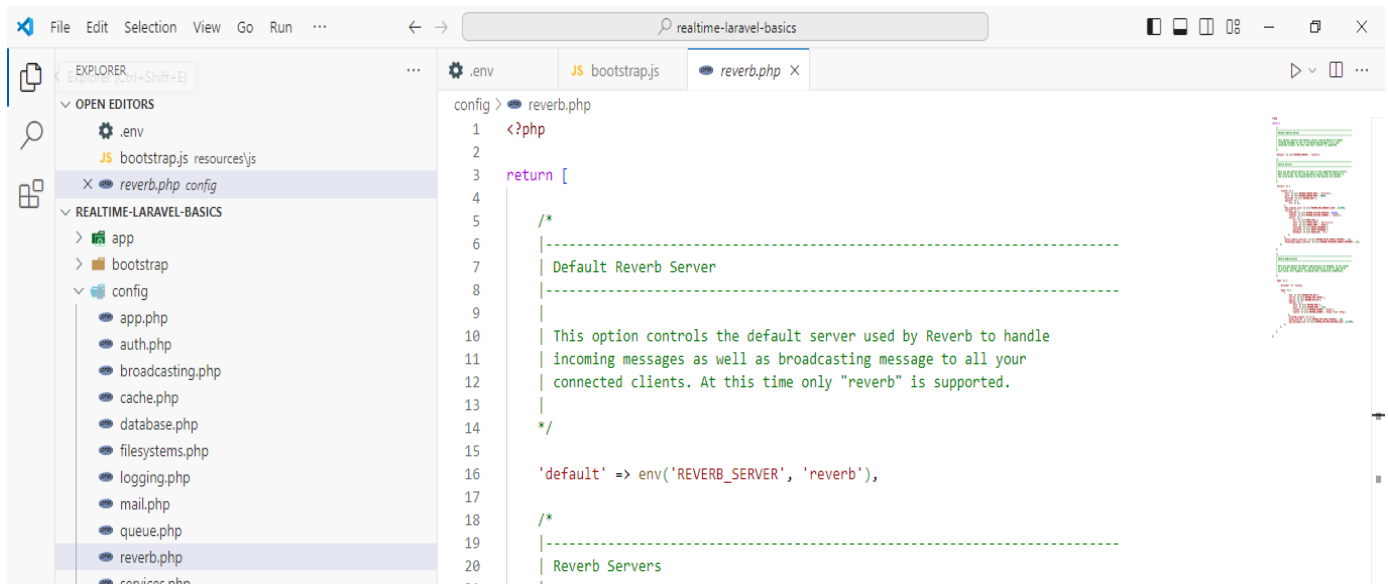
- resources
 - css
 - js
 - app.js
 - bootstrap.js
 - echo.js

The editor window shows the contents of `resources > js > bootstrap.js`. The code is as follows:

```
1 import axios from 'axios';
2 window.axios = axios;
3
4 window.axios.defaults.headers.common['X-Requested-With'] = 'XMLHttpRequest';
5
6 /**
7  * Echo exposes an expressive API for subscribing to channels and listening
8  * for events that are broadcast by Laravel. Echo and event broadcasting
9  * allow your team to quickly build robust real-time web applications.
10 */
11
12 import './echo';
13
```

Red arrows point to the `app.js` file in the file explorer and the `import './echo';` line in the code.

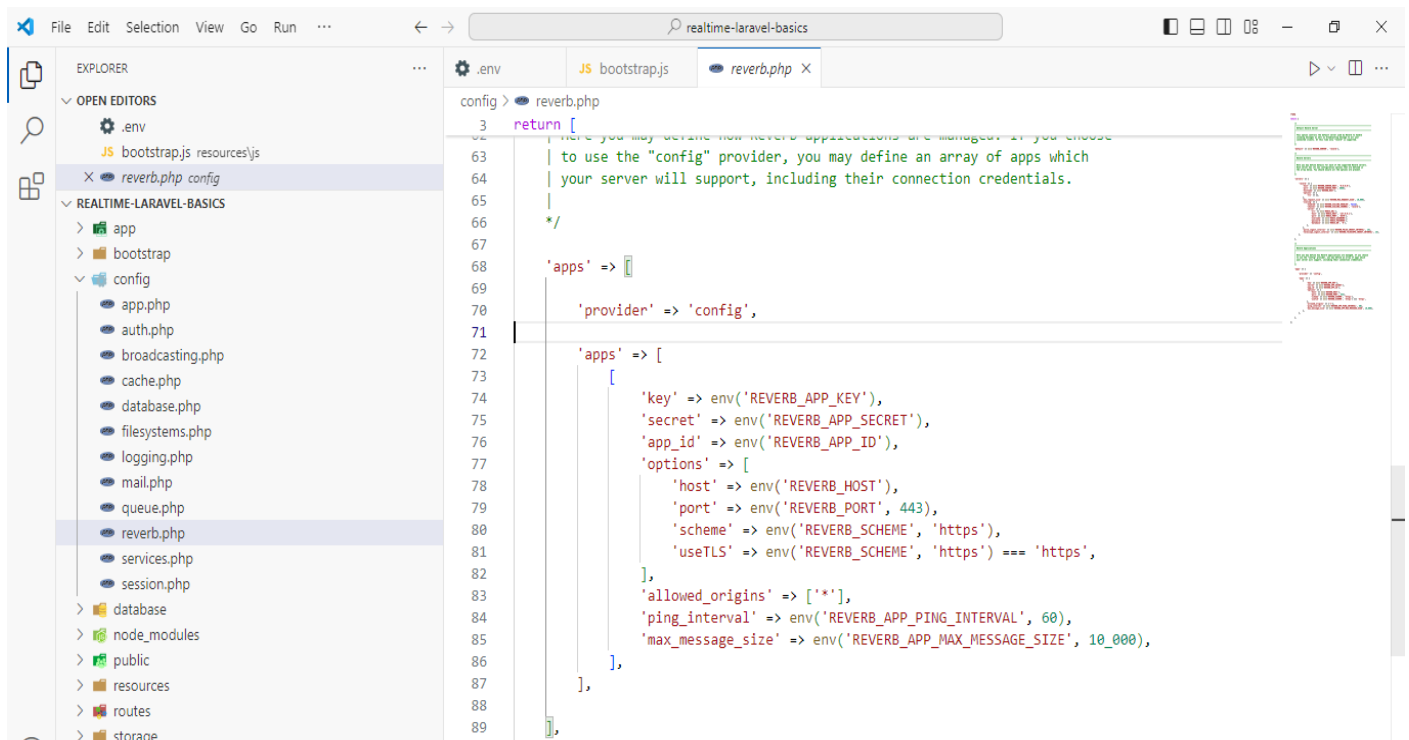
Also, After Installing Reverb, New Config File Will Be published:



The screenshot shows the Visual Studio Code editor with the file explorer on the left. The 'config' directory is expanded, showing various PHP files including 'reverb.php'. The 'reverb.php' file is open in the editor, displaying the following code:

```
1 <?php
2
3 return [
4
5     /*
6      |-----
7      | Default Reverb Server
8      |-----
9      |
10     | This option controls the default server used by Reverb to handle
11     | incoming messages as well as broadcasting message to all your
12     | connected clients. At this time only "reverb" is supported.
13     |
14     */
15
16     'default' => env('REVERB_SERVER', 'reverb'),
17
18     /*
19     |-----
20     | Reverb Servers
```

We Can Have Multiple Reverb Applications That Deal With Different Types Of Logics That Our Server Support:

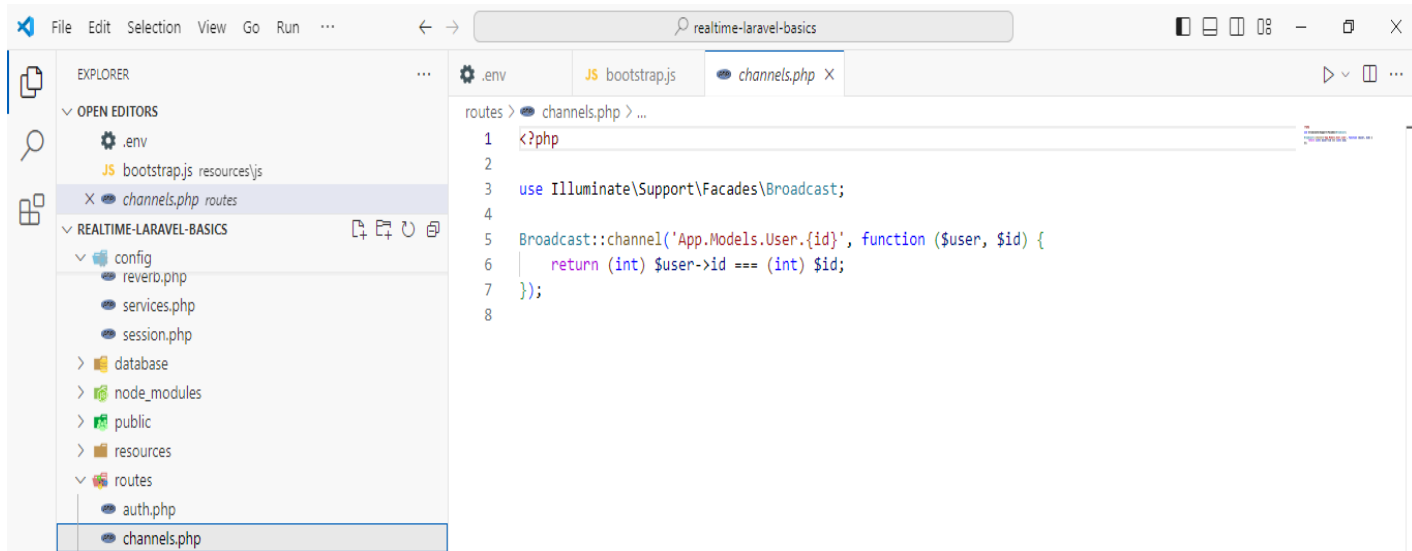


The screenshot shows the Visual Studio Code editor with the file explorer on the left. The 'config' directory is expanded, showing various PHP files including 'reverb.php'. The 'reverb.php' file is open in the editor, displaying the following code:

```
3 return [
4     /*
5      |-----
6      | Here you may define how Reverb applications are managed. If you choose
7      | to use the "config" provider, you may define an array of apps which
8      | your server will support, including their connection credentials.
9      |-----
10     */
11
12     'apps' => [
13
14         'provider' => 'config',
15
16         'apps' => [
17             [
18                 'key' => env('REVERB_APP_KEY'),
19                 'secret' => env('REVERB_APP_SECRET'),
20                 'app_id' => env('REVERB_APP_ID'),
21                 'options' => [
22                     'host' => env('REVERB_HOST'),
23                     'port' => env('REVERB_PORT', 443),
24                     'scheme' => env('REVERB_SCHEME', 'https'),
25                     'useTLS' => env('REVERB_SCHEME', 'https') === 'https',
26                 ],
27                 'allowed_origins' => ['*'],
28                 'ping_interval' => env('REVERB_APP_PING_INTERVAL', 60),
29                 'max_message_size' => env('REVERB_APP_MAX_MESSAGE_SIZE', 10_000),
30             ],
31         ],
32     ],
33 ];
```

Also, When We Have Reverb Installed, The Channels Folder Will Be Created:

Note 1: This is private Channel That Do Authentication Of The User.



Note 1: We must run: php artisan migrate

To Start The Reverb Server, We Write:

- From CMD: **php artisan reverb:start**

```
PS G:\Web\Laravel\realtime-laravel-basics> php artisan reverb:start

INFO Starting server on 0.0.0.0:8080 (localhost).
```

To Make Our Reverb Start With Debugging Capabilities, We Aet **--debug option**:

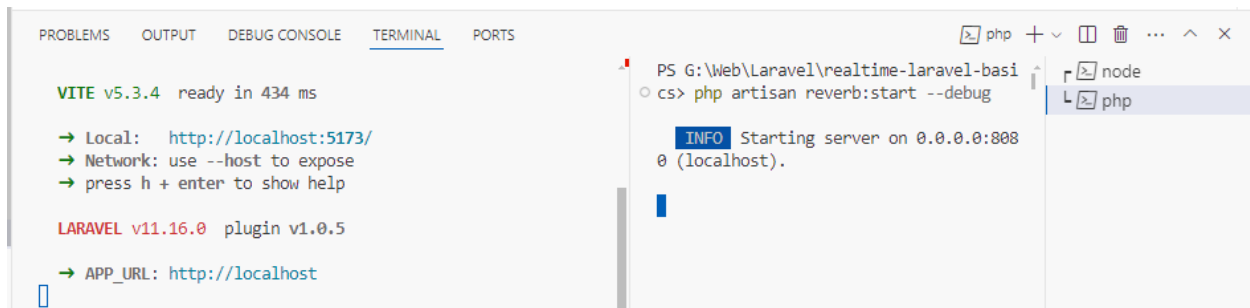
- The Command: **php artisan reverb:start --debug**

```
PS G:\Web\Laravel\realtime-laravel-basics> php artisan reverb:start --debug
```

```
INFO Starting server on 0.0.0.0:8080 (localhost).
```

To Start Our Project, We Run:

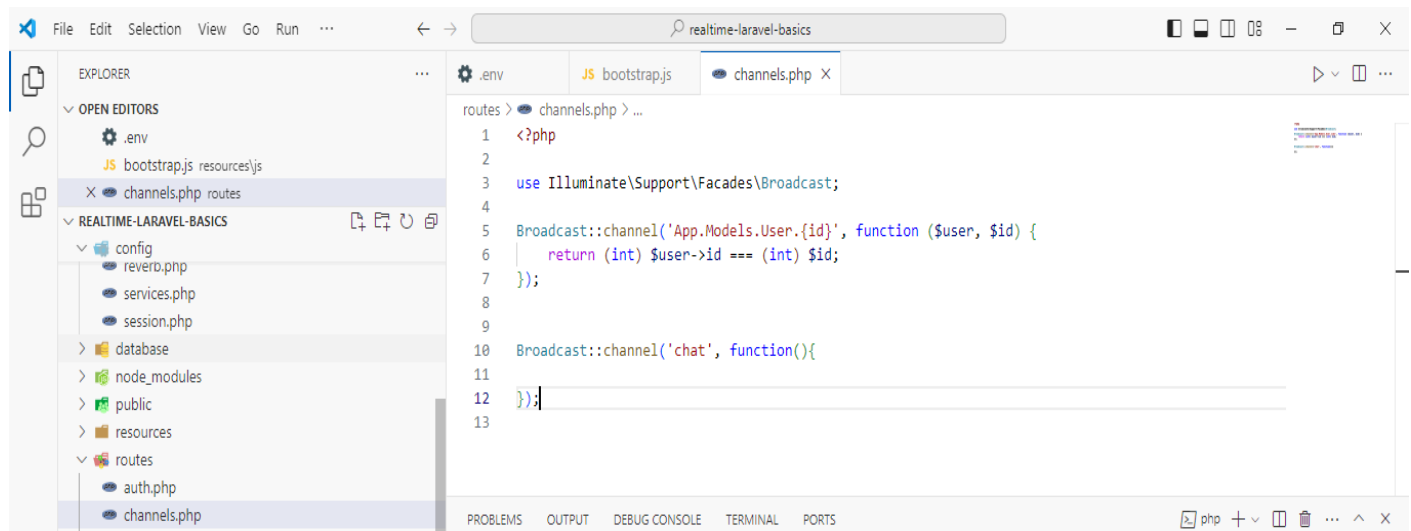
- Command 1: **npm run dev**
- Command 2: **php artisan reverb:start --debug** (Only For Local Development):



To Create New Public Channel, With Chat-As-Name, Inside **channels.php**:

- Note 1: The function accepts \$user as parameter:

```
Broadcast::channel('chat', function(){  
  
});
```



First, We Create New Example Event:

- Command: **php artisan make:event Example\ExampleEvent**

Inside **Event-Class-File** We Have Method Called broadcastOn-method, that broadcast our Event On **Private Channel**, But we Need Only **public Channel** With **chat-as-name**.

```
public function broadcastOn(): array  
{  
    return [  
        new Channel('chat'),  
    ];  
}
```

Then To make our Event able to broadcast we must:

- Extend the ShouldBroadcast-interface
 - This will Set The Broadcast On Queue And Make The Action
- OR: Extend ShouldBroadcastNow-interface
 - This Will Broadcast As It Can Be, For All Times (From The First Time The Event is Broadcasted).

```
use Illuminate\Contracts\Broadcasting\ShouldBroadcastNow;

class ExampleEvent implements ShouldBroadcastNow
{ ... }
*****
```

To Broadcast Event From Any Route, Controller, ...etc:

- Use command: **broadcast(new ExampleEvent()).**

```
Route::get('/broadcast-example-event', function(){
    broadcast(new ExampleEvent());
    return "Event is Broadcasted Successfully...";
});
*****
```