**Crear\_proyecto\_LARAVEL**

1. Instalar XAMPP
   1. APACHE
   2. PHP
   3. MariaDB/Mysql

<https://www.apachefriends.org/es/index.html>

Ir -> C:\xampp\php\php.ini

Descomentar la línea: extension=zip

Añadir la carpeta php de xampp al path.

1. Instalar Composer

<https://getcomposer.org/>

1. Instalar NodeJS

<https://nodejs.org/en>

1. Instalar y configurar Git
   1. <https://git-scm.com/>
   2. $ git config --global user.name “Tu\_nombre”
   3. $ git config --global user.email “Tu\_email”
2. Instalar Laravel

<https://laravel.com/docs/11.x/installation>

En la terminal

* 1. cd C:\xampp\htdocs\
  2. composer create-project laravel/laravel example-app
  3. php artisan serve

**Para proyecto Laravel 11 + Vue3 + InertiaJS + Vite (***Se puede instalar de frente con Breeze***)**

1. **Instala Laravel 11:**

composer create-project laravel/laravel nombre-del-proyecto

cd nombre-del-proyecto

1. **Instala las dependencias de frontend**

npm install

1. **Instala Inertia.js:**

composer require inertiajs/inertia-laravel

php artisan inertia:middleware

1. **Configura Vue 3 y Inertia.js:**

npm install @inertiajs/vue3

1. **Configura Vite (ya viene por defecto en Laravel 11): Asegúrate de que tu archivo vite.config.js esté configurado correctamente.**

npm install --save-dev @vitejs/plugin-vue

1. **Incluir el complemento en tu vite.config.js archivo de configuración. Hay algunas opciones adicionales que necesitarás al usar el complemento Vue con Laravel:**

import { defineConfig } from 'vite';

import laravel from 'laravel-vite-plugin';

import vue from '@vitejs/plugin-vue';

export default defineConfig({

plugins: [

laravel(['resources/js/app.js']),

vue({

template: {

transformAssetUrls: {

// The Vue plugin will re-write asset URLs, when referenced

// in Single File Components, to point to the Laravel web

// server. Setting this to `null` allows the Laravel plugin

// to instead re-write asset URLs to point to the Vite

// server instead.

base: null,

// The Vue plugin will parse absolute URLs and treat them

// as absolute paths to files on disk. Setting this to

// `false` will leave absolute URLs un-touched so they can

// reference assets in the public directory as expected.

includeAbsolute: false,

},

},

}),

],

});

1. **Actualiza tu archivo resources/js/app.js**

import { createApp, h } from 'vue'

import { createInertiaApp } from '@inertiajs/vue3'

createInertiaApp({

resolve: name => {

const pages = import.meta.glob('./Pages/\*\*/\*.vue', { eager: true })

return pages[`./Pages/${name}.vue`]

},

setup({ el, App, props, plugin }) {

createApp({ render: () => h(App, props) })

.use(plugin)

.mount(el)

},

})

1. **Configura tu layout principal en resources/views/app.blade.php**

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0" />

@vite('resources/js/app.js')

@inertiaHead

</head>

<body>

@inertia

</body>

</html>

1. **Crea tu primer componente Vue en resources/js/Pages/Welcome.vue**

<template>

<h1>Bienvenido a tu nuevo proyecto</h1>

</template>

<script>

export default {

// ...

}

</script>

1. **Configura una ruta en routes/web.php**

use Inertia\Inertia;

Route::get('/', function () {

return Inertia::render('Welcome');

});

1. **Ejecuta el servidor de desarrollo**

php artisan serve

1. **En otra terminal, compila los assets**

npm run dev

**Para proyecto Laravel 11: Autentificación (BREEZE)**

$ composer create-project laravel/laravel nombre-del-proyecto

$ composer require laravel/breeze --dev

$ php artisan breeze:install

Se debe elegir opciones para seguir con la instalación (Vue + InertiaJS)

$ php artisan migrate

$ php artisan serve

$ npm run dev

\*\* Breeze instala automáticamente Vue3 e InertiaJS por lo que no es necesario instalarlos antes.

Para poner Login como pagina principal (lo primero que se sea al ejecutar la app):

En /routes/web.php

Borrar:

Route::get('/', function () {

    return Inertia::render('Welcome', [

        'canLogin' => Route::has('login'),

        'canRegister' => Route::has('register'),

        'laravelVersion' => Application::VERSION,

        'phpVersion' => PHP\_VERSION,

    ]);

});

Añadir:  
  
Route::get('/', function () {

    if (auth()->check()) {

        return redirect()->route('dashboard');

    }

    return redirect()->route('login');

})->name('home');

\*\* Se debe configurar los permisos como RegisterUser ya que está activo para cualquier usuario.

**Para proyecto Laravel 11: Migración de base de datos**

* Crear solo migraciones:

$ php artisan make:migration create\_nombretabla\_table

* Crear solo modelo

$ php artisan make:model Nombremodelo

* Crear solo controller

$ php artisan make:controller NombreControlador

*Ejemplo: php artisan make:controller EstadoController*

* Crear modelo y migraciones

$ php artisan make:model Nombremodelo --migration

* Crear modelo, migraciones y controller (CRUD)

$ php artisan make:model Nombremodelo -m --resource

**Ejecutar comandos para crear models y migrations**

$ php artisan make:model Payroll --migration

$ php artisan make:model Employee --migration

$ php artisan make:model PayrollType --migration

$ php artisan make:migration create\_employee\_payroll\_table

Orden: (**Necesariamente crear migration en ese orden por tema de llave s foráneas**)

1. **User (ya está creado solo modificar)**
2. **payrollTypes**
3. **payroll**
4. **employee**
5. **employee\_payroll**

**Modificar archivos:**

*En database/migrations/xxxx\_xx\_xx\_create\_users\_table.php*

public function up()

{

Schema::create('users', function (Blueprint $table) {

$table->id();

$table->string('name', 150);

$table->string('email', 100)->unique();

$table->timestamp('email\_verified\_at')->nullable();

$table->string('password', 255);

        $table->rememberToken();

$table->timestamps();

});

}

*En database/migrations/xxxx\_xx\_xx\_create\_employees\_table.php*

public function up()

{

Schema::create('employees', function (Blueprint $table) {

$table->id();

$table->string('dni', 30)->unique();

$table->string('first\_name', 150);

$table->string('last\_name', 150);

$table->timestamps();

});

}

*En database/migrations/xxxx\_xx\_xx\_create\_payrolls\_table.php*

public function up()

{

Schema::create('payrolls', function (Blueprint $table) {

$table->id();

$table->string('tittle', 255)->nullable();

$table->string('year', 4);

$table->string('month', 50);

$table->string('file\_uri', 255);

$table->foreignId('payroll\_type\_id')->constrained();

$table->foreignId('user\_id')->constrained();

$table->timestamps();

});

}

*En database/migrations/xxxx\_xx\_xx\_create\_payroll\_types\_table.php*

public function up()

{

Schema::create('payroll\_types', function (Blueprint $table) {

$table->id();

$table->string('name\_type', 100);

$table->timestamps();

});

}

*En database/migrations/xxxx\_xx\_xx\_create\_employee\_payroll\_table.php*

public function up()

{

Schema::create('employee\_payroll', function (Blueprint $table) {

$table->foreignId('payroll\_id')->constrained()->onDelete('cascade');

$table->foreignId('employee\_id')->constrained()->onDelete('cascade');

$table->primary(['payroll\_id', 'employee\_id']);

$table->timestamps();

});

}

**Modificar los models**

*Employee.php*

    public function payrolls()

    {

        return $this->belongsToMany(Payroll::class);

    }

*Payroll.php*

    public function payrollType()

    {

        return $this->belongsTo(PayrollType::class);

    }

    public function user()

    {

        return $this->belongsTo(User::class);

    }

    public function employees()

    {

        return $this->belongToMany(Employee::class);

    }

*PayrollType.php*

    public function payrolls()

    {

        return $this->hasMany(Payroll::class);

    }

*User.php*

    public function payrolls ()

    {

        return $this->hasMany(Payroll::class);

    }

**Agregar Seeder de PayrollType**

$ php artisan make:seeder PayrollTypeSeeder

*En database/seeders/PayrollTypeSeeder.php*

namespace Database\Seeders;

use Illuminate\Database\Console\Seeds\WithoutModelEvents;

use Illuminate\Database\Seeder;

use App\Models\PayrollType;

class PayrollTypeSeeder extends Seeder

{

    public function run(): void

    {

        PayrollType::create([

            'name\_type' => 'Haberes',

        ]);

        PayrollType::create([

            'name\_type' => 'Jornales',

        ]);

    }

}

*Actualizar database/seeders/DatabaseSeeder.php*

    public function run()

    {

        $this->call([

            PayrollTypeSeeder::class,

        ]);

    }

Ejecutar comando:

*php artisan migrate*

*php artisan db:seed*

**Ayuda con git y github**

[**https://docs.github.com/es/authentication**](https://docs.github.com/es/authentication)

**CLONAR REPOSITORIO DE GITHUB**

* Al clonar un repositorio de Github hay archivos que no se clonan a causa de .gitignore como:

/.phpunit.cache

/node\_modules

/public/build

/public/hot

/public/storage

/storage/\*.key

**/vendor**

**.env**

.env.backup

.env.production

.phpactor.json

.phpunit.result.cache

Homestead.json

Homestead.yaml

auth.json

npm-debug.log

yarn-error.log

/.fleet

/.idea

/.vscode

* **Para solucionar .env**

$ cp .env.example .env

* **Para solucionar /vendor**

$ composer install

* **Para solucionar vite**

$ npm install

* **Actualizar los datos de .env**

DB\_CONNECTION=sqlite

# DB\_HOST=127.0.0.1

# DB\_PORT=3306

# DB\_DATABASE=laravel

# DB\_USERNAME=root

# DB\_PASSWORD=

$ php artisan key:generate

$ php artisan migrate

**AVANZAR EL PROYECTO DESDE UN COMMIT, PERO TENER ERRORES.**

**PASOS PARA VOLVER AL INICIO DEL COMMIT**

* **Primero, obtén el hash del primer commit. Puedes usar el comando git log para ver el historial de commits**

$ git log --oneline

* **Resetear el Repositorio al Commit**

$ git reset –hard [hash]

* **Volver a la Rama Principal**

$ git checkuot main

**Videos de ayuda:**

* **CRUD Full Stack con Laravel, Inertia y Vue**

<https://www.youtube.com/watch?v=esIlt5wREL8&t=323s>

* **Laravel - Tablar - Login, Dashboard y CRUD ¡Fácil y rápido!**

<https://www.youtube.com/watch?v=JbAtwtCN8PI>

* **APLICACIÓN REAL DE LARAVEL, VUE LARAVEL 9 - 10**

<https://www.youtube.com/watch?v=xgu6NbM06gA&list=PLDllzmccetSPjp-wGeCUnzoU_f0wxE-t_>

**#1 Desarrollando un Proyecto Real con Laravel y Vue: Paso a Paso | GOGODEV**

(creación de model, migration, controller – ejemplo de base de datos many-to-many : 38:00)

**#2 Desarrollando un Proyecto Real con Laravel y Vue: Paso a Paso | GOGODEV**

(roles y permisos)

Paginas\_de\_ayuda

# **Buenas prácticas para escribir commits en Git**

<https://midu.dev/buenas-practicas-escribir-commits-git/>

# **Cómo Instalar Laravel en Windows, macOS y Linux**

<https://kinsta.com/es/base-de-conocimiento/instalar-laravel/#windows-6>

<https://tablar.ebuz.xyz/>

**# Open-Source Admin Template for Laravel, and More**

<https://tablar.ebuz.xyz/>

**Cómo clonar un repositorio Laravel desde GitHub**

<https://www.youtube.com/watch?v=EdZ0hQtrfEU>

**Manual de instalación TABLAR**

<https://tablar.ebuz.xyz/docs/11.0/overview>

**Autentificación Laravel**

<https://laravel.com/docs/11.x/authentication>

<https://laravel.com/docs/11.x/starter-kits#laravel-breeze>

**Laravel 11 Many-to-Many Relationship**

<https://medium.com/@maitrikt1998/laravel-many-to-many-relationship-4bbe2d70dbec>

**Laravel 11 One-to-Many Relationship**

<https://medium.com/@maitrikt1998/laravel-one-to-many-relationship-2d165a8db6b5>

**Laravel 11 Multi Auth using Breeze with CRUD Admin**

<https://www.youtube.com/watch?v=b-qEj11h7as>

**Doble CRUD Laravel 11 Vue 3 con Inertia | Tailwind | ChartJs | Select2 | Subir imágenes | Dashboard**

<https://www.youtube.com/watch?v=aaR9JNkFcnc&t=1561s>