

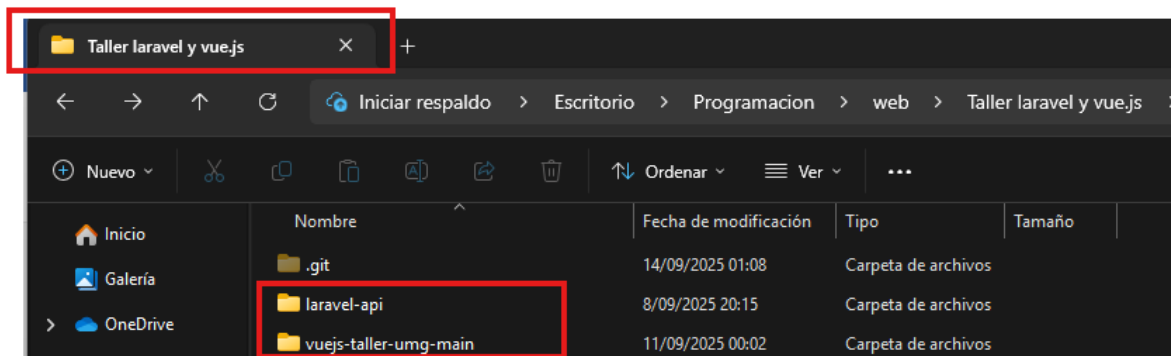
Nombre: Allan Guamuch

Carne:2990-19-22192

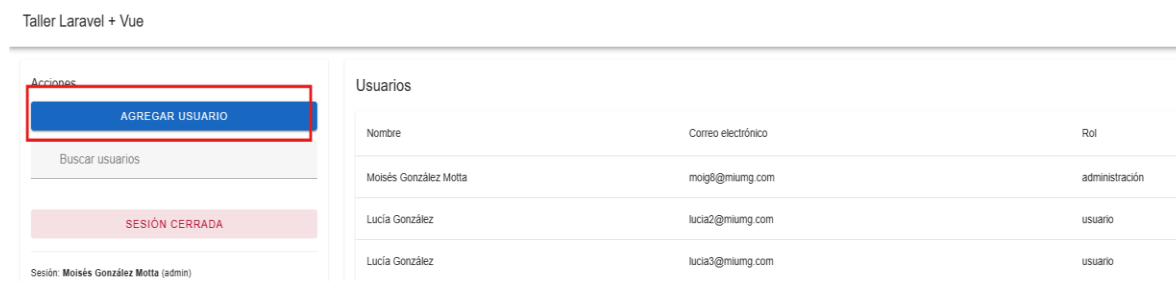
Curso: Desarrollo Web

Link: <https://github.com/ALLANROBER/Taller-laravel-Vuejs.git>

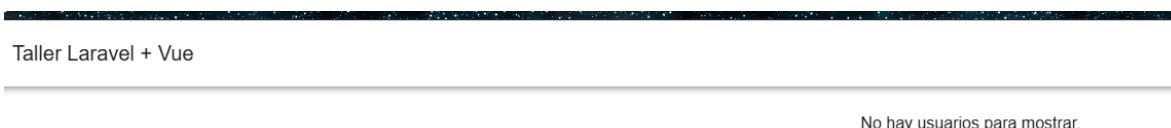
Para este taller cree una carpeta llamada Taller laravel y vue.js en la cual están las carpetas laravel-api y vuejs-taller-umg-main uno que es el proyecto **Laravel (backend)** y el otro **Vue.js (frontend)**



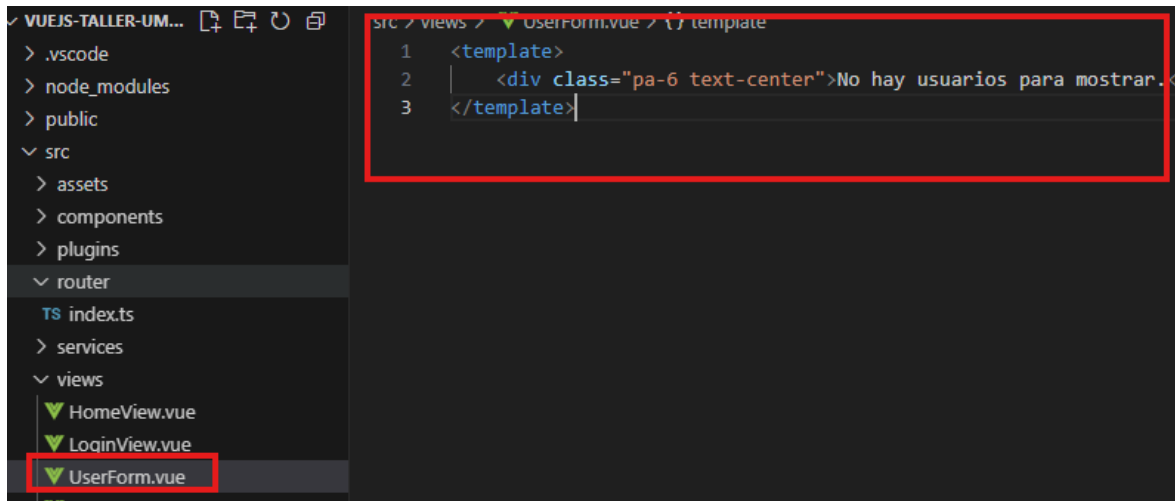
Al ejecutar el proyecto vue logramos ver los usuarios que ya tenemos en nuestra base de datos y también un botón llamado AGREGAR USUARIO



Al apretar el botón Agregar usuario nos redirecciona a otra pagina en la cual solo nos muestra un mensaje

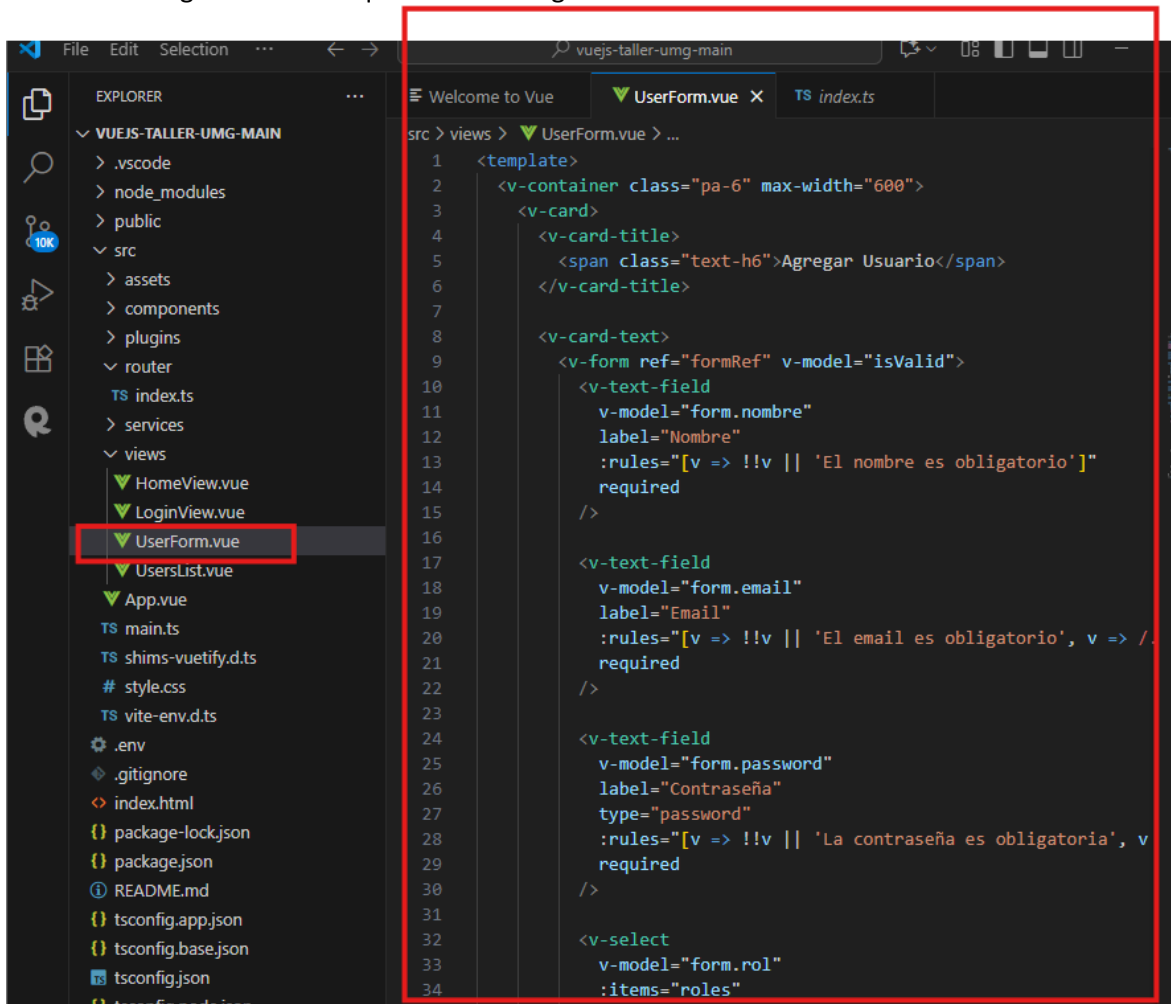


El mensaje proviene del proyecto vue de views/userform.vue, como primer tarea es modificar ese código para que poder agregar usuarios desde el navegador



```
src > views > UserForm.vue > {} template
1 <template>
2   <div class="pa-6 text-center">No hay usuarios para mostrar.</div>
3 </template>
```

El código modificado quedaría de la siguiente manera



```
vuejs-taller-umg-main
Welcome to Vue
UserForm.vue x TS index.ts
src > views > UserForm.vue > ...
1 <template>
2   <v-container class="pa-6 max-width=600">
3     <v-card>
4       <v-card-title>
5         <span class="text-h6">Agregar Usuario</span>
6       </v-card-title>
7
8       <v-card-text>
9         <v-form ref="formRef" v-model="isValid">
10          <v-text-field
11            v-model="form.nombre"
12            label="Nombre"
13            :rules="[v => !!v || 'El nombre es obligatorio']"
14            required
15          />
16
17          <v-text-field
18            v-model="form.email"
19            label="Email"
20            :rules="[v => !!v || 'El email es obligatorio', v => /.+@.+$/]"
21            required
22          />
23
24          <v-text-field
25            v-model="form.password"
26            label="Contraseña"
27            type="password"
28            :rules="[v => !!v || 'La contraseña es obligatoria', v => /.+@.+$/]"
29            required
30          />
31
32          <v-select
33            v-model="form.rol"
34            :items="roles"
```

Al guardar el código ya podríamos ver la ventana de agregar usuarios, como vemos en la siguiente imagen ya hemos agregado un usuario y ya podemos guardarlo

Taller Laravel + Vue

Agregar usuario

Nombre

Allan

Correo electrónico

Allan@miumg.com

Contraseña

Rol

administración

GUARDAR

Al dar clic al botón guardar podremos ver una pequeña notificación de que se ha agregado correctamente el usuario

Taller Laravel + Vue

Agregar usuario

Nombre

El nombre es obligatorio

Correo electrónico

El correo electrónico es obligatorio

Contraseña

La contraseña es obligatoria

Rol

Selecciona un rol

Usuario creado correctamente

Al validar la ventana de usuarios podemos ver que efectivamente se agregó el usuario añadido

Taller Laravel + Vue

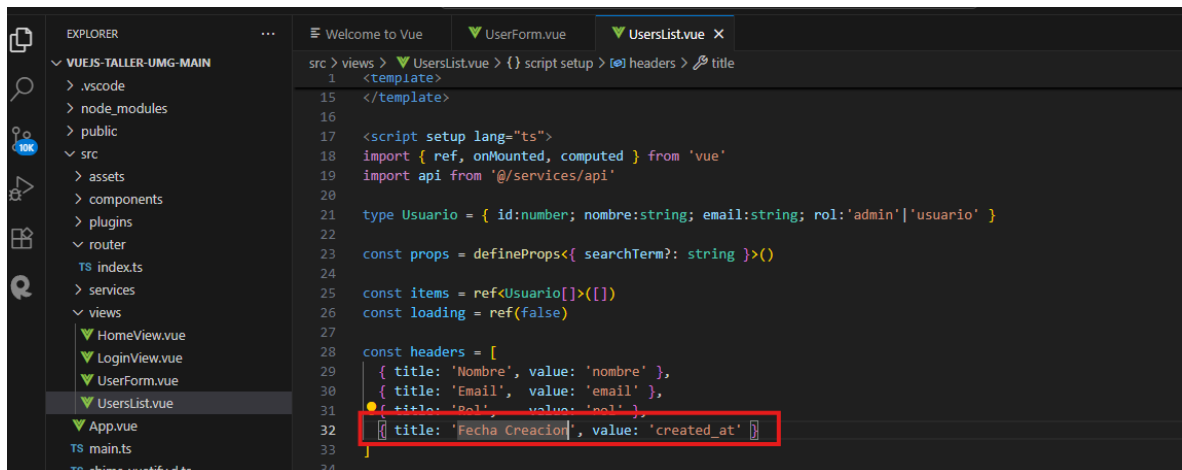
Acciones		
AGREGAR USUARIO		
Buscar usuarios		
SESIÓN CERRADA		
Sesión: Moisés González Motta (admin)		

Usuarios		
Nombre	Correo electrónico	Rol
Moisés González Motta	moig8@miung.com	administración
Lucía González	lucia2@miung.com	usuario
Lucía González	lucia3@miung.com	usuario
sss	jajaj@miung.com	administración
holabb	mijapontebonita@miung.com	administración
Alano	Alano@miung.com	administración

Como vemos en la vista solo podemos visualizar lo que es nombre, correo electrónico y rol, el siguiente reto es hacer que se visualice la fecha de creación del registro de usuario, para eso vamos a agregar en el archivo vue en views/userlist.vue la línea de código para poder visualizarlo, como vemos en la siguiente imagen en el código solo tenemos nombre, email y rol.

```
1 <template>
2   <v-data-table
3     :items="filtered"
4     :headers="headers"
5     :loading="loading"
6     class="elevation-1"
7   >
8     <template #no-data>
9       <div class="pa-6 text-center">No hay usuarios para mostrar.</div>
10    </template>
11  </v-data-table>
12 </template>
13
14 <script setup lang="ts">
15   import { ref, onMounted, computed } from 'vue'
16   import api from '@/services/api'
17
18   type Usuario = { id:number; nombre:string; email:string; rol:'admin'
19
20   const props = defineProps<{ searchTerm?: string }>()
21
22   const items = ref<Usuario[]>([])
23   const loading = ref(false)
24
25   const headers = [
26     { title: 'Nombre', value: 'nombre' },
27     { title: 'Email', value: 'email' },
28     { title: 'Rol', value: 'rol' },
29   ]
30
31   // carga desde la API
32   const fetchUsers = async () => {
33     loading.value = true
34     try {
```

Al agregar el código para visualizar la fecha de usuario quedaría de la siguiente manera



```
src > views > UsersList.vue > {} script setup > headers > title
1 <template>
15 </template>
16
17 <script setup lang="ts">
18 import { ref, onMounted, computed } from 'vue'
19 import api from '@services/api'
20
21 type Usuario = { id:number; nombre:string; email:string; rol:'admin'|'usuario' }
22
23 const props = defineProps<{ searchTerm?: string }>()
24
25 const items = ref<Usuario[]>([])
26 const loading = ref(false)
27
28 const headers = [
29   { title: 'Nombre', value: 'nombre' },
30   { title: 'Email', value: 'email' },
31   { title: 'Rol', value: 'rol' },
32   { title: 'Fecha Creación', value: 'created_at' }
33 ]
34
```

Como vemos ya tenemos la fecha de creación, solo que hay un detalle la fecha no está compuesta para poder componerla y que se mire bonita deberemos agregar otro pedazo de código

Taller Laravel + Vue

Acciones

AGREGAR USUARIO

Buscar usuarios

CERRAR SESIÓN

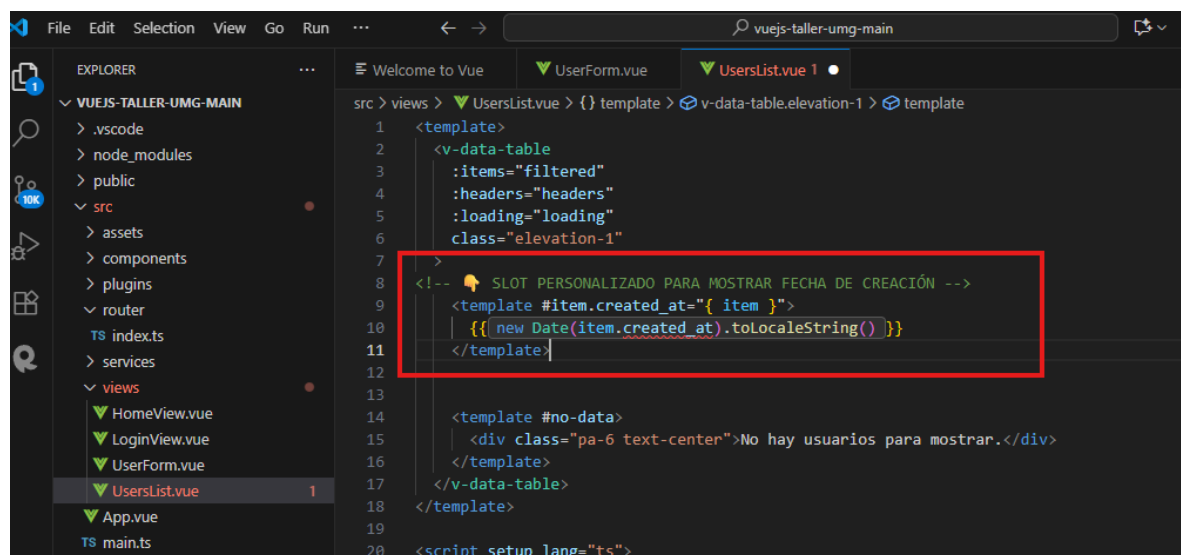
Sesión: Moises Gonzalez Motta (admin)

Usuarios

Nombre	Email	Rol	Fecha Creación
Moises Gonzalez Motta	moiga@miung.com	admin	2025-09-11T05:04:59.000000Z
lucia gonzalez	lucia2@miung.com	usuario	2025-09-11T05:17:41.000000Z
lucia gonzalez	lucia3@miung.com	usuario	2025-09-11T05:17:49.000000Z
sss	jajaj@miung.com	admin	2025-09-14T06:51:38.000000Z
holab	mijapontebonita@miung.com	admin	2025-09-14T06:52:37.000000Z
Allan	Allan@miung.com	admin	2025-09-14T07:35:24.000000Z

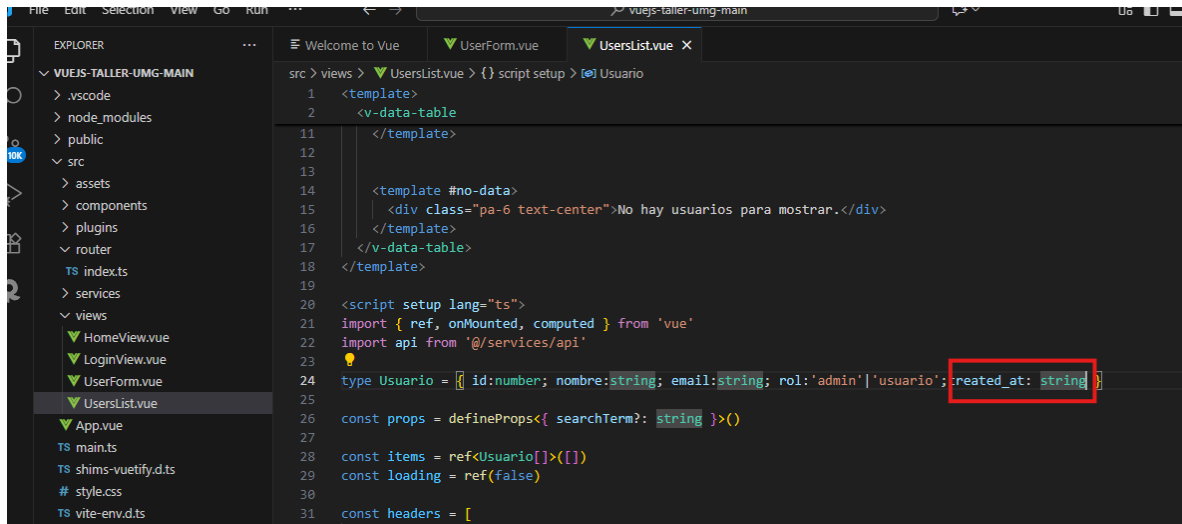
Items per page: 10 1-6 of 6

El código para que la fecha de mire bonita es la siguiente



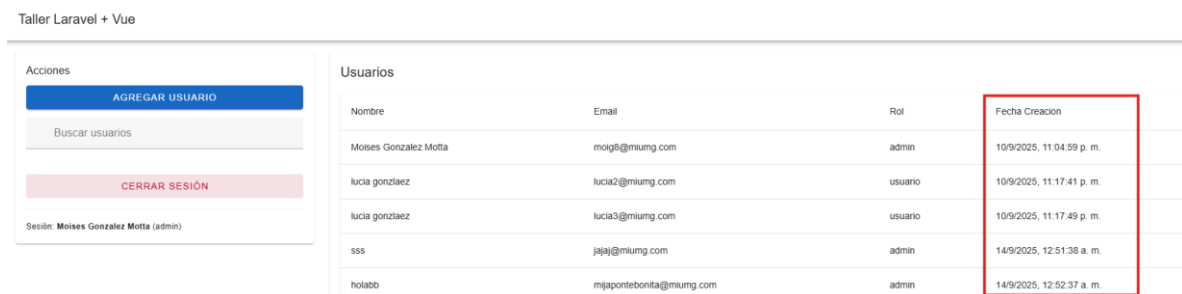
```
src > views > UsersList.vue > {} template > v-data-table.elevation-1 > template
1 <template>
2   <v-data-table
3     :items="filtered"
4     :headers="headers"
5     :loading="loading"
6     class="elevation-1"
7   >
8     <!-- SLOT PERSONALIZADO PARA MOSTRAR FECHA DE CREACIÓN -->
9     <template #item.created_at="{ item }">
10       {{ new Date(item.created_at).toLocaleString() }}
11     </template>
12
13
14   <template #no-data>
15     <div class="pa-6 text-center">No hay usuarios para mostrar.</div>
16   </template>
17 </v-data-table>
18 </template>
19
20 <script setup lang="ts">
```

Pero para que el código funcione y no tire error debemos agregar la propiedad `created_at` al tipo `Usuario` en tu `<script setup>` de `UsersList.vue`. y quedaría así



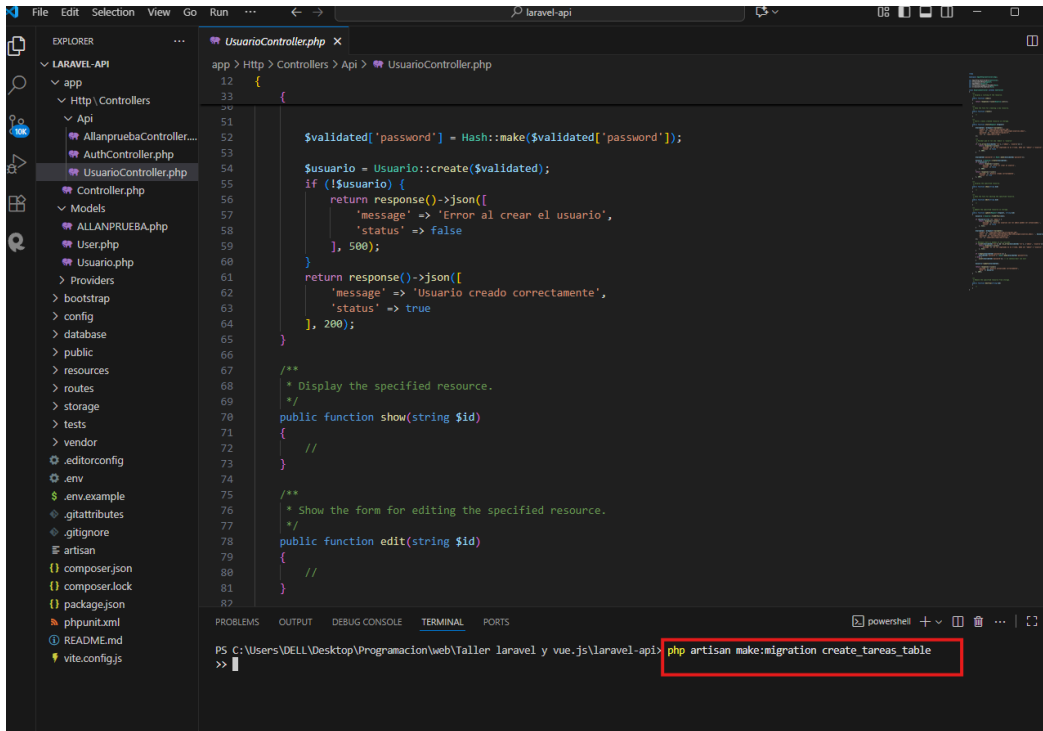
```
1 <template>
2   <v-data-table>
11 </template>
12
13
14   <template #no-data>
15     <div class="pa-6 text-center">No hay usuarios para mostrar.</div>
16   </template>
17 </v-data-table>
18 </template>
19
20 <script setup lang="ts">
21   import { ref, onMounted, computed } from 'vue'
22   import api from '@services/api'
23
24   type Usuario = { id:number; nombre:string; email:string; rol:'admin'|'usuario'; created_at: string }
25
26   const props = defineProps<{ searchTerm?: string }>()
27
28   const items = ref<Usuario[]>([])
29   const loading = ref(false)
30
31   const headers = [
```

Al ya guardar los cambios, podremos ver la fecha ya bien moldeada como se muestra en la siguiente imagen.



Nombre	Email	Rol	Fecha Creacion
Moises Gonzalez Motta	moig8@miiumg.com	admin	10/9/2025, 11:04:59 p. m.
lucia gonzlaez	lucia2@miiumg.com	usuario	10/9/2025, 11:17:41 p. m.
lucia gonzlaez	lucia3@miiumg.com	usuario	10/9/2025, 11:17:49 p. m.
sss	jajaj@miiumg.com	admin	14/9/2025, 12:51:38 a. m.
holiab	mijapontebonifa@miiumg.com	admin	14/9/2025, 12:52:37 a. m.

ejecutamos código en el proyecto de laravel para la creación de la tabla de migración de usuarios

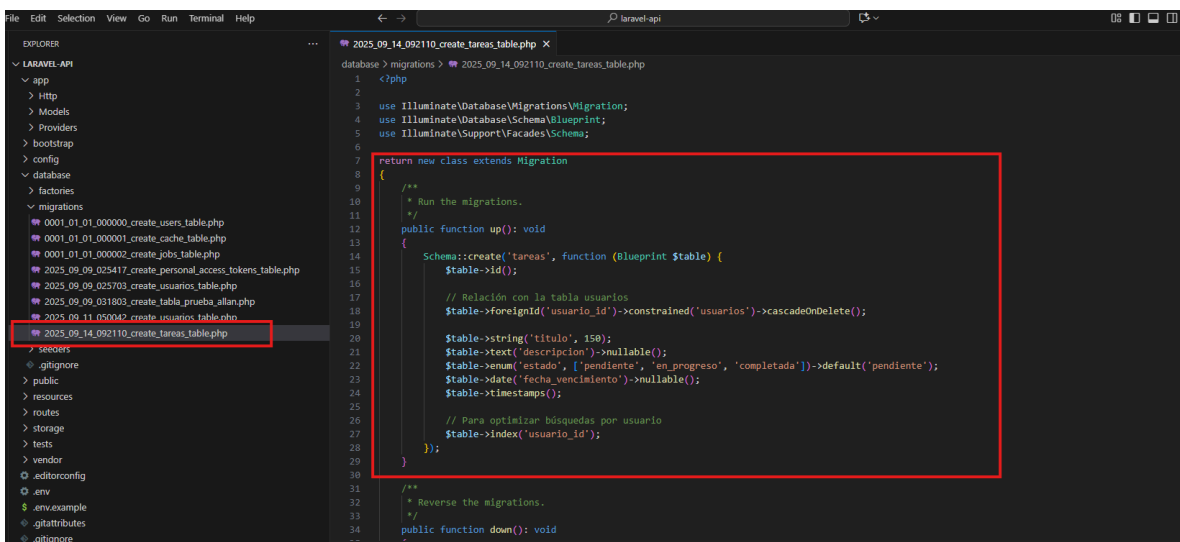


The screenshot shows the VS Code editor with the 'UsuarioController.php' file open. The file contains PHP code for a Laravel API controller. The terminal window at the bottom shows the command 'php artisan make:migration create_tareas_table' being executed.

```
app > Http > Controllers > Api > UsuarioController.php
12 {
33 {
51
52 $validated['password'] = Hash::make($validated['password']);
53
54 $usuario = Usuario::create($validated);
55 if (!$usuario) {
56     return response()->json([
57         'message' => 'Error al crear el usuario',
58         'status' => false
59     ], 500);
60 }
61 return response()->json([
62     'message' => 'Usuario creado correctamente',
63     'status' => true
64 ], 200);
65 }
66
67 /**
68  * Display the specified resource.
69  */
70 public function show(string $id)
71 {
72     //
73 }
74
75 /**
76  * Show the form for editing the specified resource.
77  */
78 public function edit(string $id)
79 {
80     //
81 }
82 }
```

PS C:\Users\DELL\Desktop\Programacion\Web\Taller laravel y vue.js\laravel-api> php artisan make:migration create_tareas_table

Agregamos el siguiente código



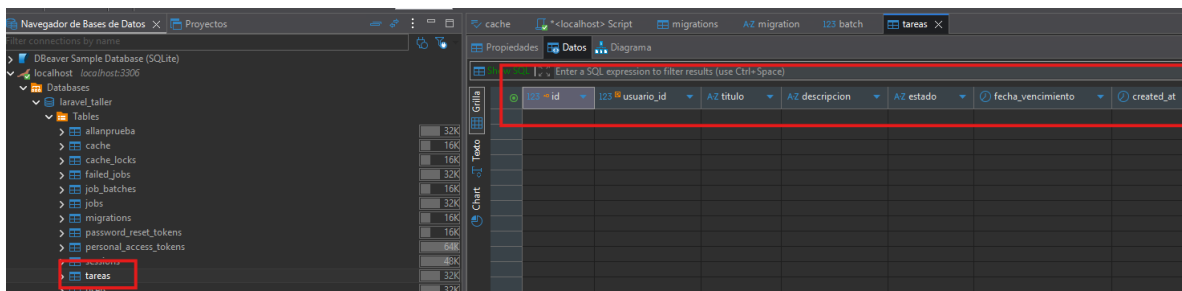
The screenshot shows the VS Code editor with the '2025_09_14_092110_create_tareas_table.php' migration file open. The file contains PHP code for a Laravel migration. The code is highlighted with a red box.

```
database > migrations > 2025_09_14_092110_create_tareas_table.php
1 <?php
2
3 use Illuminate\Database\Migrations\Migration;
4 use Illuminate\Database\Schema\Blueprint;
5 use Illuminate\Support\Facades\Schema;
6
7 return new class extends Migration
8 {
9     /**
10      * Run the migrations.
11      */
12     public function up(): void
13     {
14         Schema::create('tareas', function (Blueprint $table) {
15             $table->id();
16
17             // Relación con la tabla usuarios
18             $table->foreignId('usuario_id')->constrained('usuarios')->cascadeOnDelete();
19
20             $table->string('titulo', 150);
21             $table->text('descripcion')->nullable();
22             $table->enum('estado', ['pendiente', 'en progreso', 'completada'])->default('pendiente');
23             $table->date('fecha_vencimiento')->nullable();
24             $table->timestamps();
25
26             // Para optimizar búsquedas por usuario
27             $table->index('usuario_id');
28         });
29     }
30
31     /**
32      * Reverse the migrations.
33      */
34     public function down(): void
35     {
36     }
```

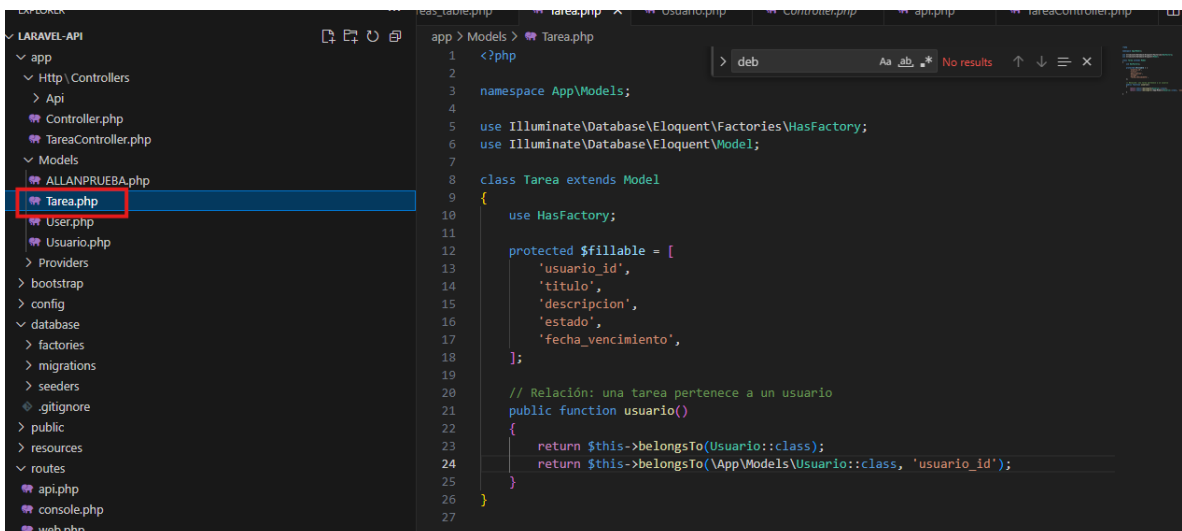
Luego de eso ejecutamos en la terminal php artisan migrate para crear la tabla en la base de datos

```
33 public function down(): void
34
-- -- --
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\DELL\Desktop\Programacion\web\Taller laravel y vue.js\laravel-api> php artisan make:migration create_tareas_table
>>
[INFO] Migration [C:\Users\DELL\Desktop\Programacion\web\Taller laravel y vue.js\laravel-api\database\migrations\2025_09_14_092110_create_tareas_table.php] created successfully.
PS C:\Users\DELL\Desktop\Programacion\web\Taller laravel y vue.js\laravel-api> php artisan migrate
>>
```

Se crea la tabla de base de datos



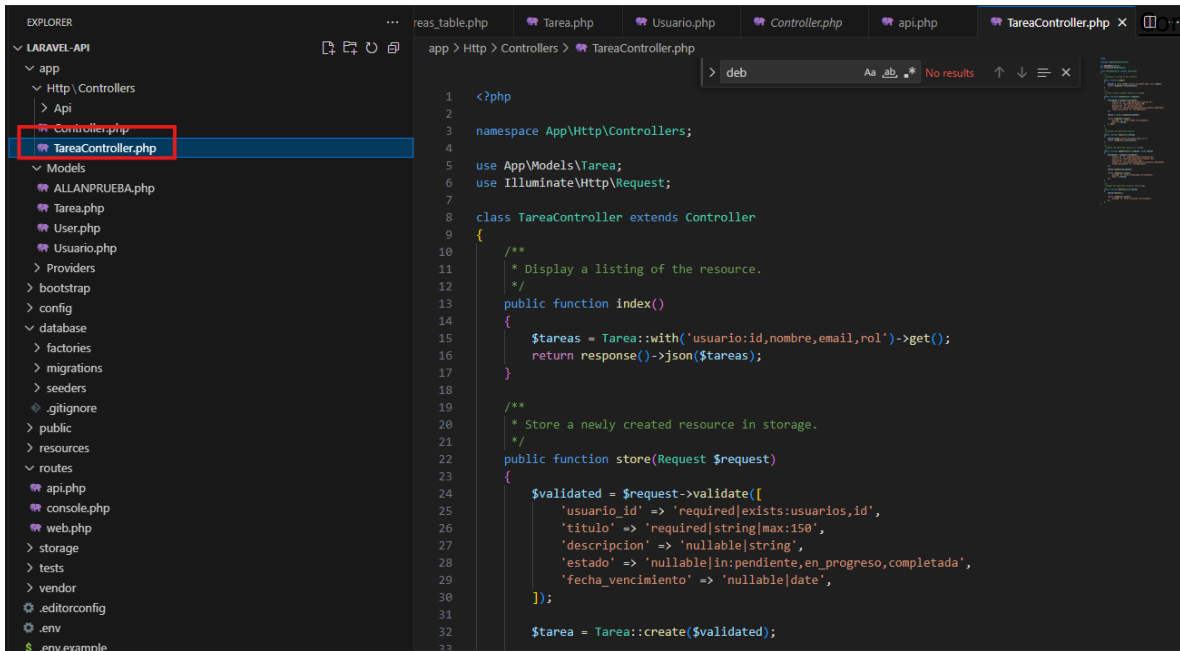
Para que nos funciones la base de datos y podamos verla mediante el navegador Creamos un modelo llamado Tarea.php



Se crea un controlador llamado tarea controller.php

Tarea.php = hoja de Excel donde guardas los datos de las tareas.

TareaController.php = la persona que recibe tus pedidos, los procesa y actualiza la hoja de Excel o te devuelve la info que pediste.



The screenshot shows an IDE with a file explorer on the left and a code editor on the right. In the file explorer, the file `TareaController.php` is highlighted under the `Http\Controllers` directory. The code editor displays the following PHP code for `TareaController.php`:

```
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use App\Models\Tarea;
6 use Illuminate\Http\Request;
7
8 class TareaController extends Controller
9 {
10     /**
11      * Display a listing of the resource.
12      */
13     public function index()
14     {
15         $tareas = Tarea::with('usuario:id,nombre,email,rol')->get();
16         return response()->json($tareas);
17     }
18
19     /**
20      * Store a newly created resource in storage.
21      */
22     public function store(Request $request)
23     {
24         $validated = $request->validate([
25             'usuario_id' => 'required|exists:usuarios,id',
26             'titulo' => 'required|string|max:150',
27             'descripcion' => 'nullable|string',
28             'estado' => 'nullable|in:pendiente,en_progreso,completada',
29             'fecha_vencimiento' => 'nullable|date',
30         ]);
31
32         $tarea = Tarea::create($validated);
33     }
34 }
```