1-12-2023

pc

ALLAN XAVIER HERRERA AREVALO

PRUEBA TECNICA

FULLSTACK (LARAVEL)

**Prueba Logica**

**El lago musical**

Un día, un explorador comenzó a seguir algunos sonidos extraños, luego llegó a un lago donde encontró la fuente: 3 animalitos haciendo sonidos parecidos a una canción. Pasó poco tiempo y el explorador pudo diferenciar qué animal producía cada sonido.

Sonidos

* Rana: brr, birip, brrah, croac
* Libélula: fiu, plop, pep
* Grillo: cric-cric, trri-trri, bri-bri

Después de un tiempo descubrió que estaban "cantando" juntos. Siempre que la rana comenzaba con brr, la libélula respondía a su sonido frotando su cola con una rama produciendo un sonido como fiu, después de la libélula, el grillo continuaba con cric-cric pero, cada vez que la rana sonaba como brrah o croac todos los animales se silenciaron. por un tiempo, y luego continuaron de nuevo. Antes de que el cielo se oscureciera, el explorador pudo escribir 3 "canciones" que todos hicieron juntos y se le ocurrieron estas notas.

Canciones

* brr, fiu, cric-cric, brrah
* pep, birip, trri-trri, croac
* bri-bri, plop, cric-cric, brrah

EJERCICIO

Usando el lenguaje de programación con el que te sienta más cómodo, escriba un programa que le permita recibir un sonido dado de la lista de sonidos que hace cada animal y devuelva los sonidos restantes de cualquiera de las 3 canciones que escribió el explorador, por ejemplo:

* Cuando se le da brr debe reproducir fiu, cric-cric, brrah de acuerdo con la primera canción.
* Cuando se le da birip, debe reproducir trri-trri, croac según la segunda canción.
* Cuando se le da plop debe reproducir cric-cric, brrah de acuerdo con la tercera canción.
* Cuando se le da croac o brrah, no debería reproducir nada de acuerdo con todas las canciones.

**Solución**

Para la construcción y posterior solución del problema planteado se empleó como lenguaje de programación de JavaScript

const playSong = (sound) => {

  try {

    const songsList = [

      ["brr", "fiu", "cric-cric", "brrah"],

      ["pep", "birip", "trri-trri", "croac"],

      ["bri-bri", "plop", "cric-cric", "brrah"],

    ];

    for (const song of songsList) {

      const i = song.indexOf(sound);

      if (i !== -1) {

        return song.slice(i + 1);

      }

    }

    return [];

  } catch (error) {

    console.log(error);

  }

};

// CASOS

console.log(playSong("brr"));

console.log(playSong("birip"));

console.log(playSong("plop"));

console.log(playSong("croac"));

console.log(playSong("brrah"));

**Resultados Obtenidos**

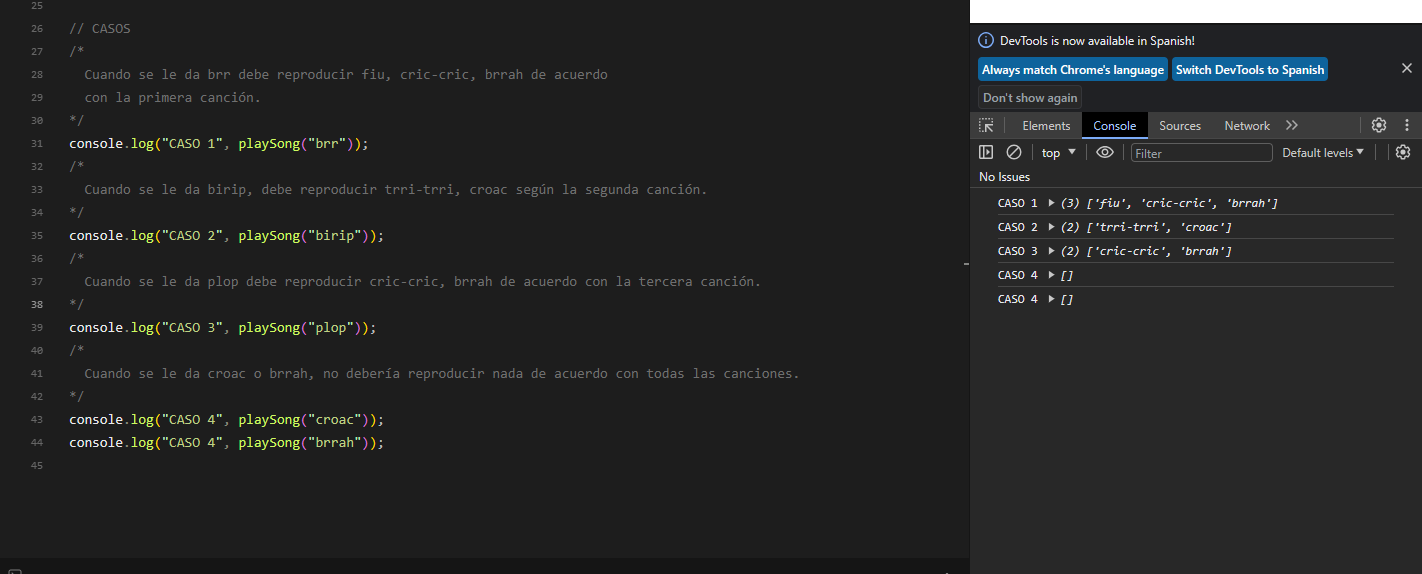


Ilustración 1 resultados de la prueba lógica

**Adjunto el código**: [index.js - nodebox - CodeSandbox](https://codesandbox.io/p/sandbox/peaceful-moon-klh7hw?file=%2Fsrc%2Findex.js%3A5%2C11)

**Test Full Stack - To Do List**

El siguiente ejercicio consiste en crear una “Lista de tareas”. Debe de completarse usando como base el repositorio provisto en el siguiente link, como respuesta a la prueba esperamos un pull request (o un enlace del proyecto) con el código que solucione los problemas a continuación planteados.

Utilizando el framework Laravel implementar un aplicativo web que permita administrar un TODO list (ver diseño sugerido), con las siguientes características:

* Los campos de las tareas son título, orden y un indicador que determine si está realizada o no.
* El usuario podrá filtrar las tareas.
* El usuario podrá ordenar las tareas.
* Las tareas se podrán ser eliminadas.
* El frontend podría estar escrito en Vue + Vuetify (o tiene libertad de escoger el framework que guste).
* La base de datos se creará usando migrations.
* Pantalla de login/registro. (Opcional)
* Asociar tareas por usuario (completar login/registro). (Opcional)

**Desarrollo**

Para el desarrollo de la aplicación se empleó como herramientas de desarrollo el editor visual studio code y el entorno de desarrollo Laragon, además para la construcción del frontend se empleó la biblioteca React js junto con Inertia, pueden revisar más a profundidad en los siguientes enlaces:

* [Inertia.js - The Modern Monolith (inertiajs.com)](https://inertiajs.com/)
* [Tailwind CSS - Rapidly build modern websites without ever leaving your HTML.](https://tailwindcss.com/)
* [Inertia | Laravel Jetstream](https://jetstream.laravel.com/stacks/inertia.html)

Inicio de la aplicación

Para arrancar la aplicación puede abrir una terminar integrada en vs code y ejecutar en la raíz del proyecto “npm run dev”

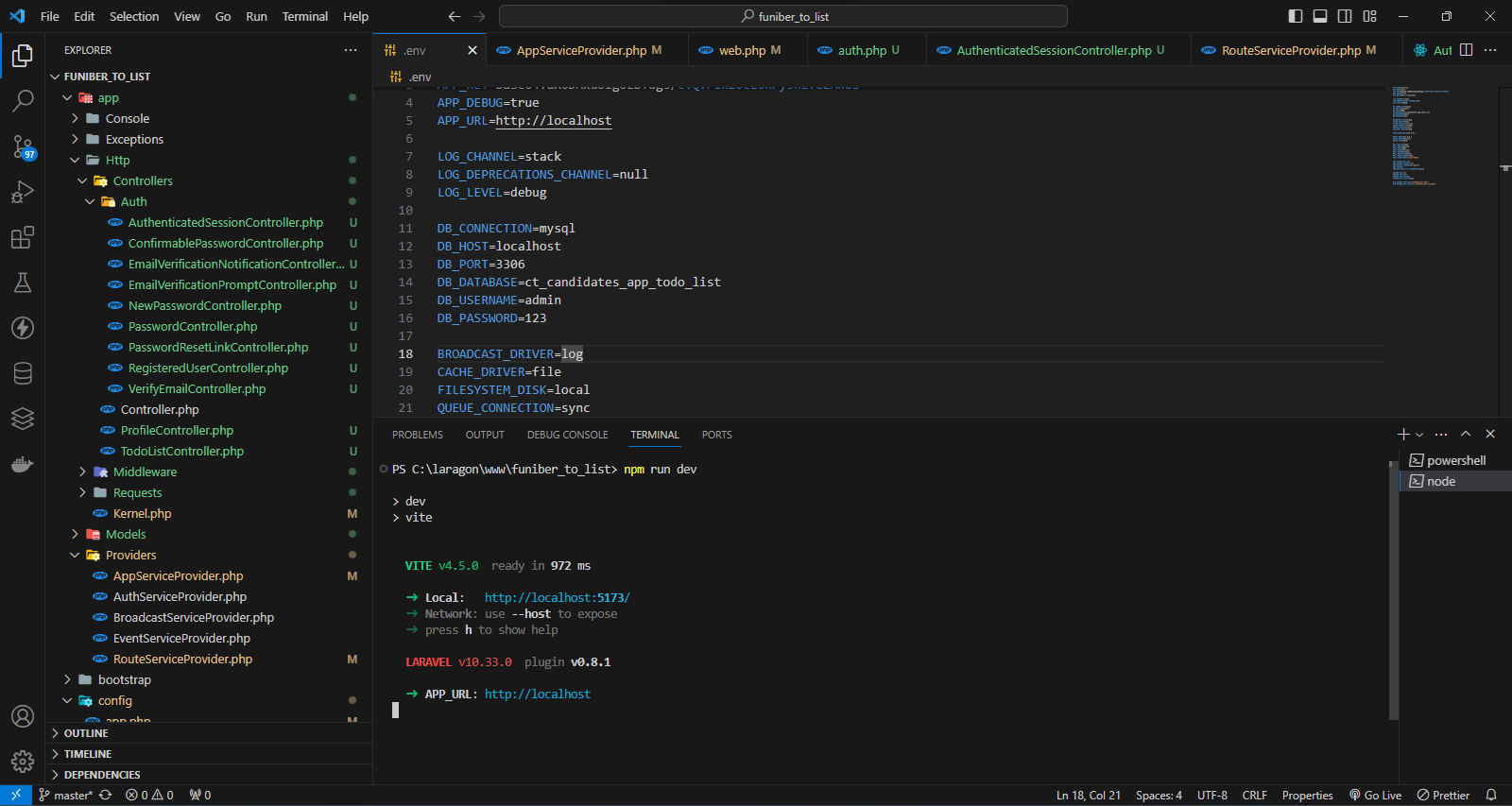


Ilustración 2 Inicio del proyecto

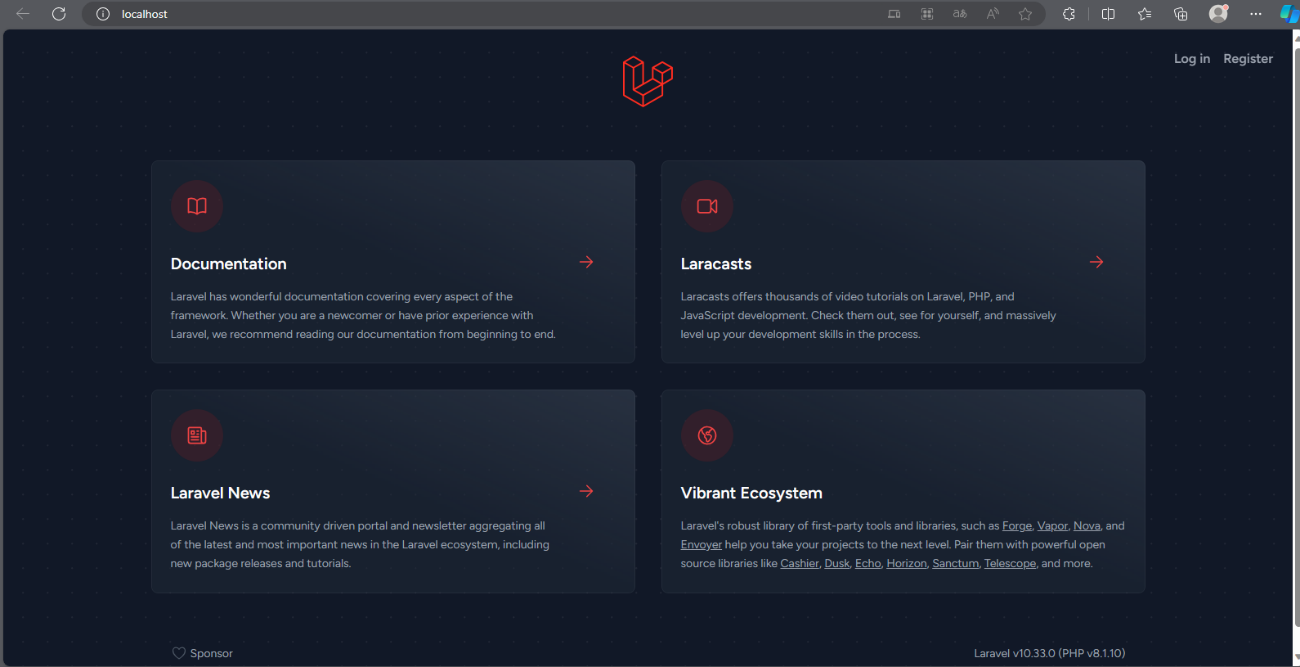


Ilustración 3 Pantalla Inicial

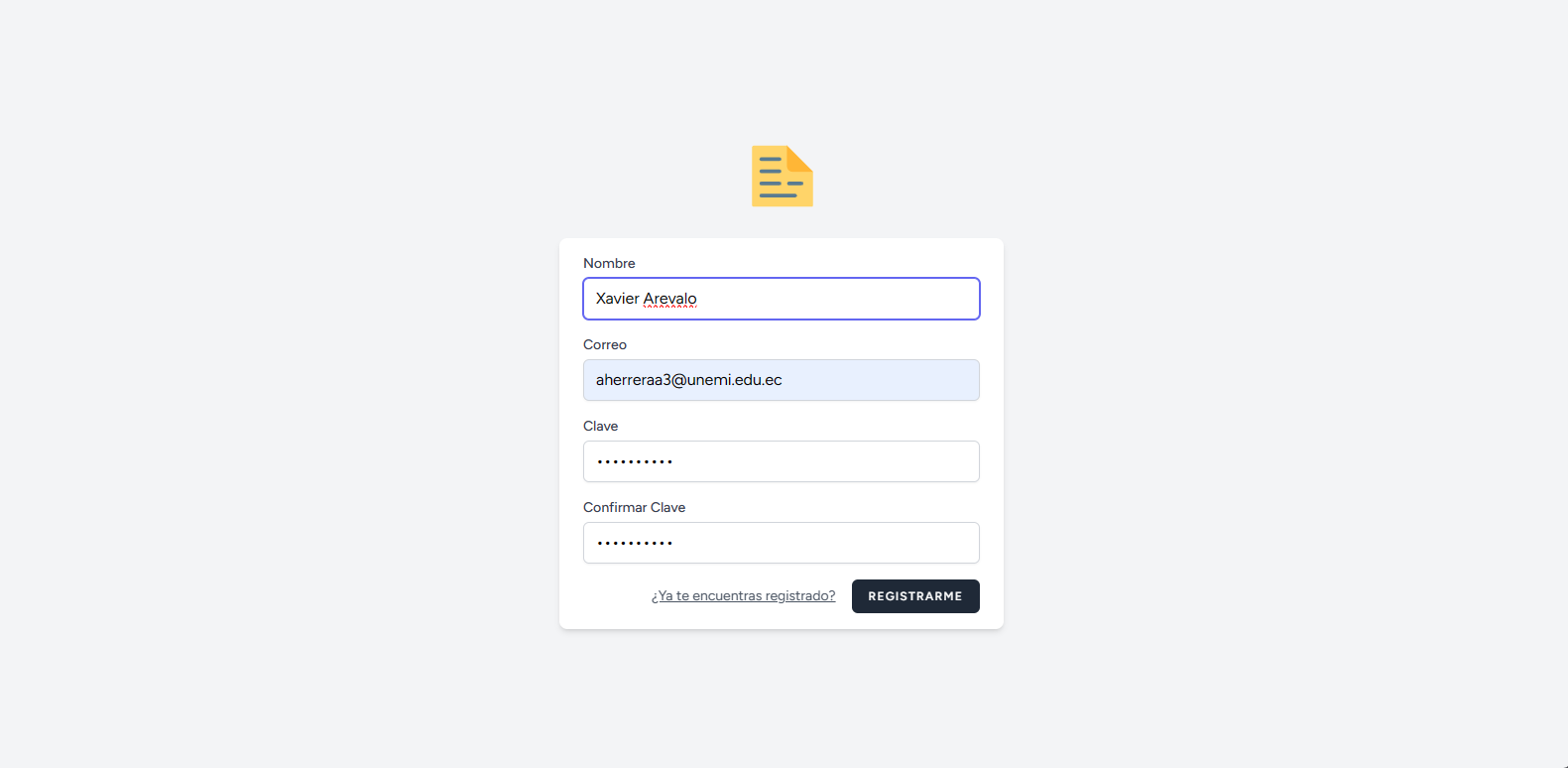


Ilustración 4 Registro de usuario

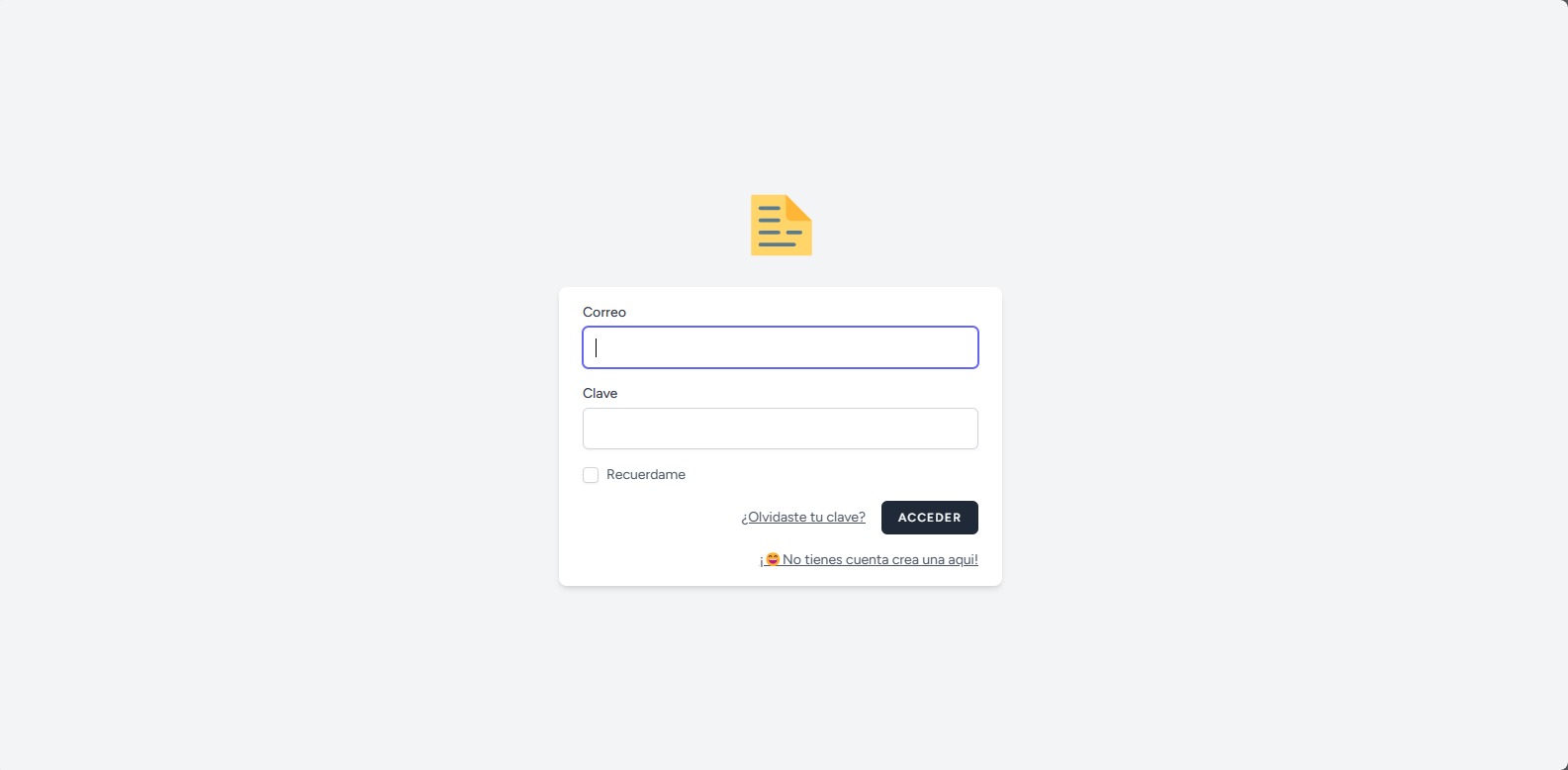


Ilustración 5 Inicio de sesión

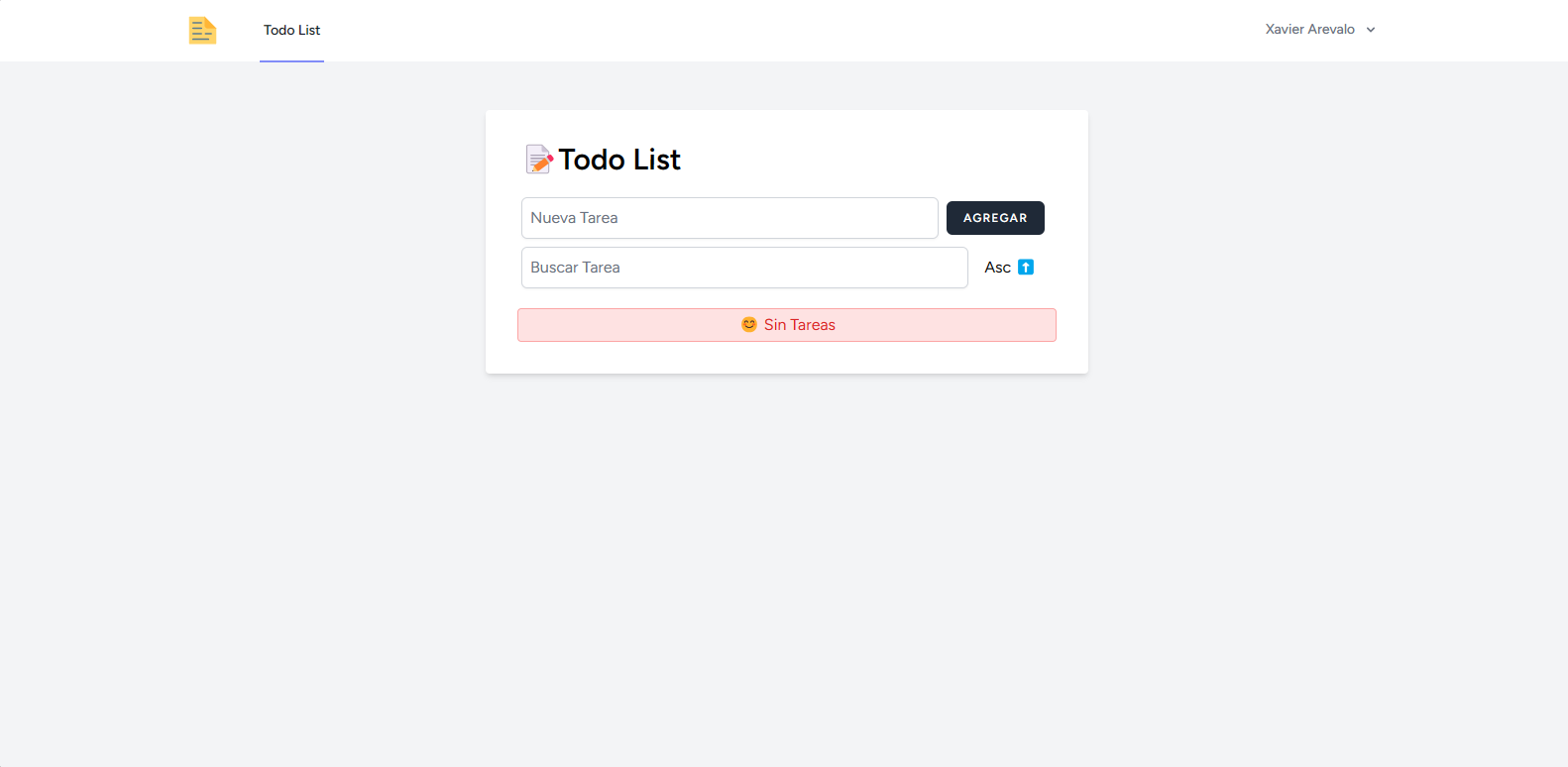


Ilustración 6 Todo List

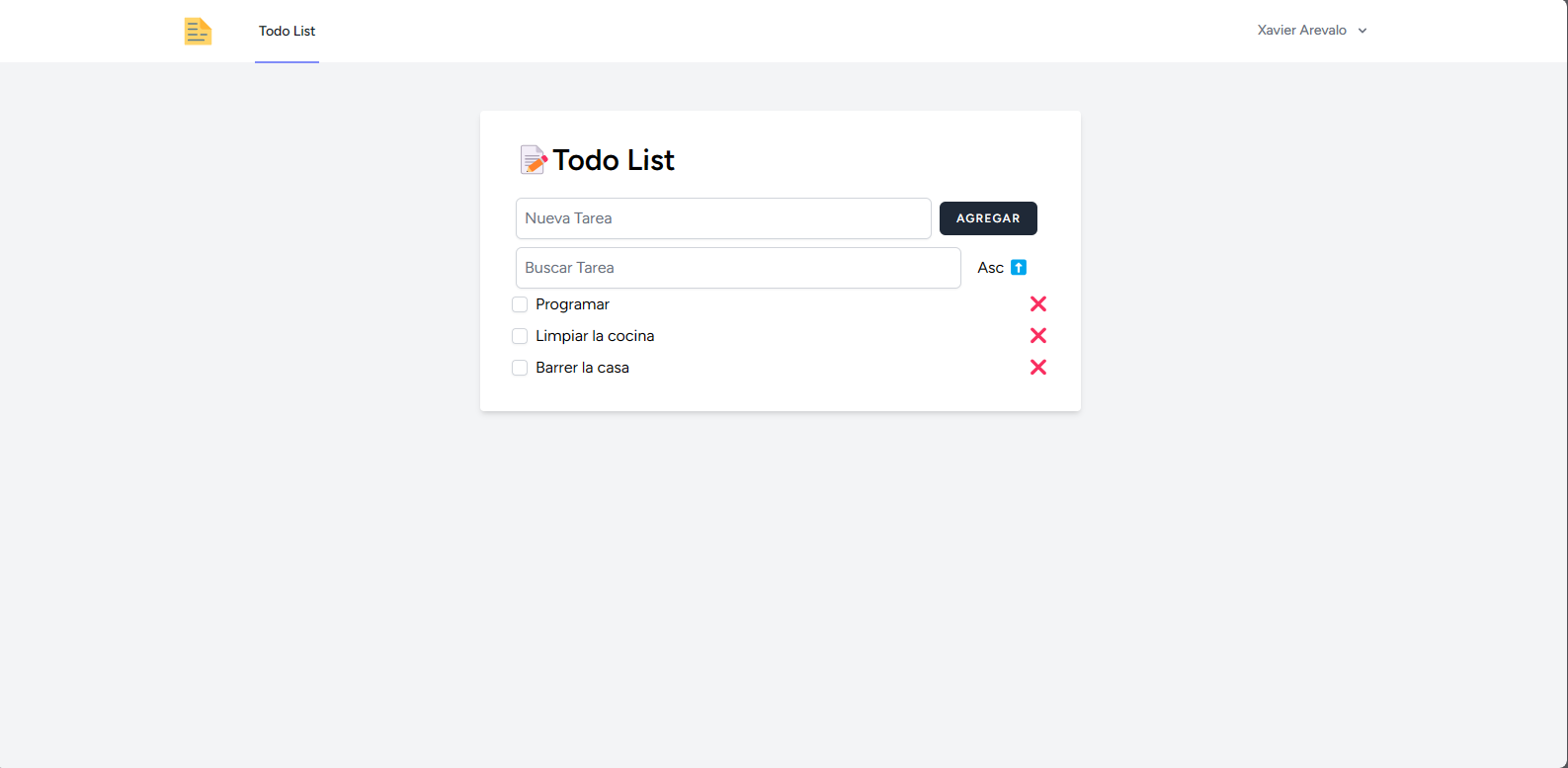


Ilustración 7 Agregar Tarea

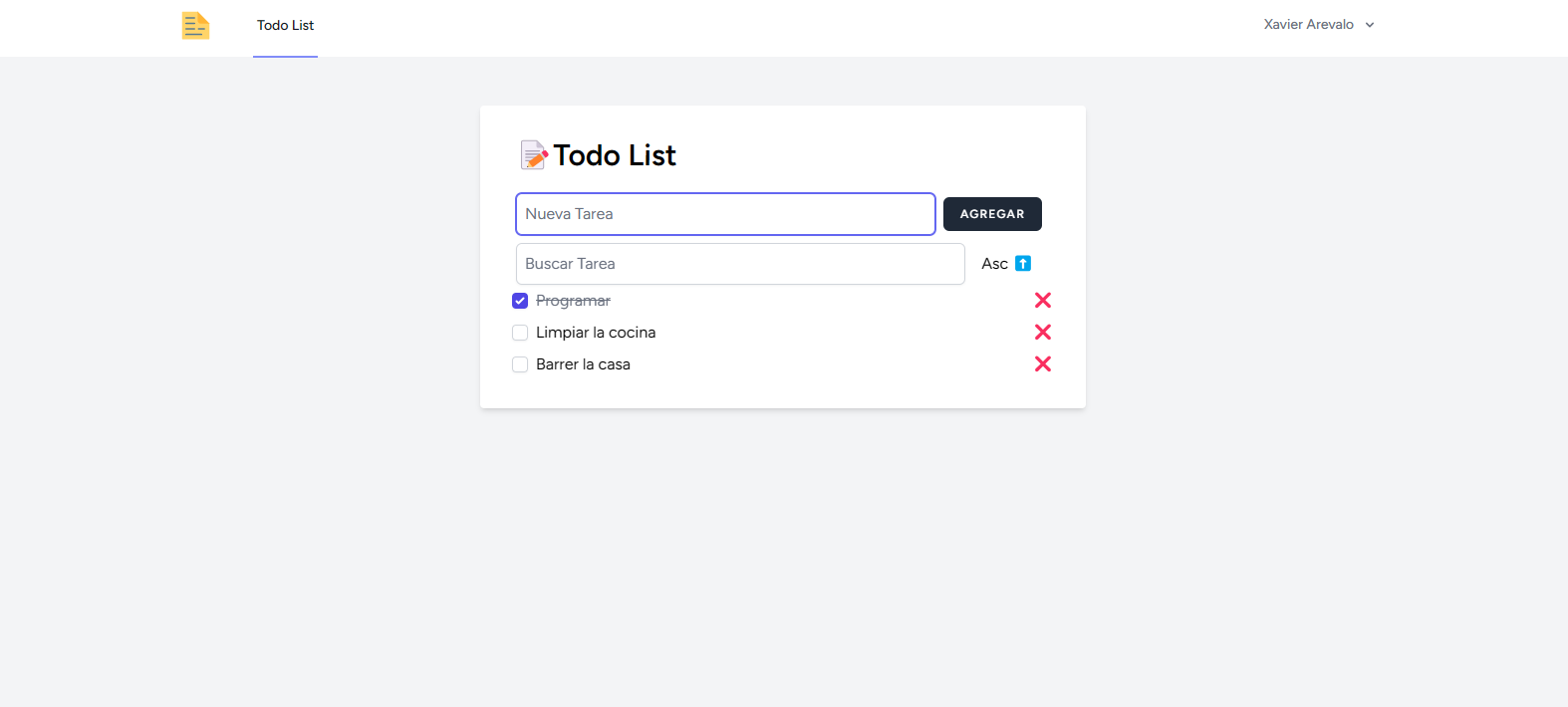


Ilustración 8 Completar Tarea

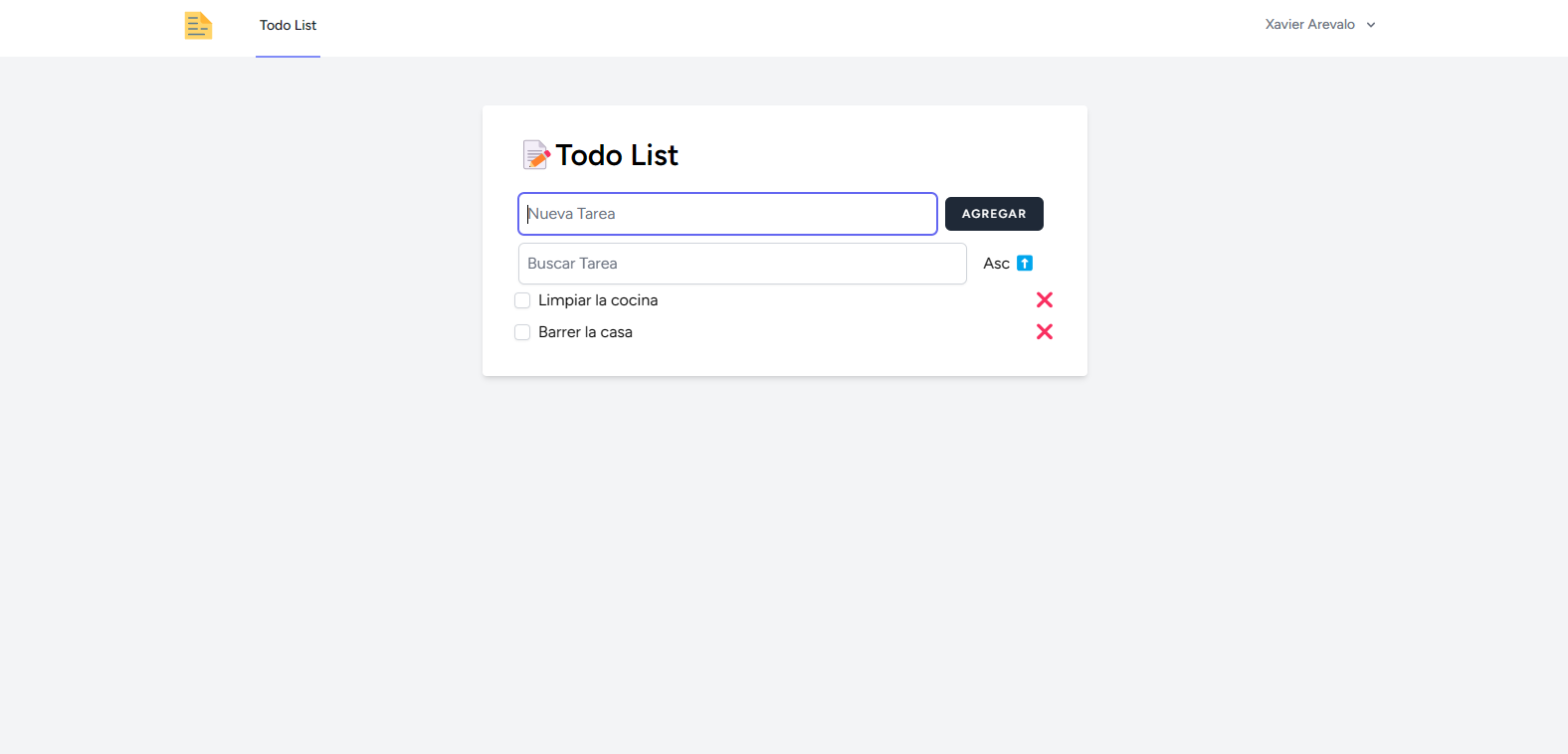


Ilustración 9 Eliminar Tarea

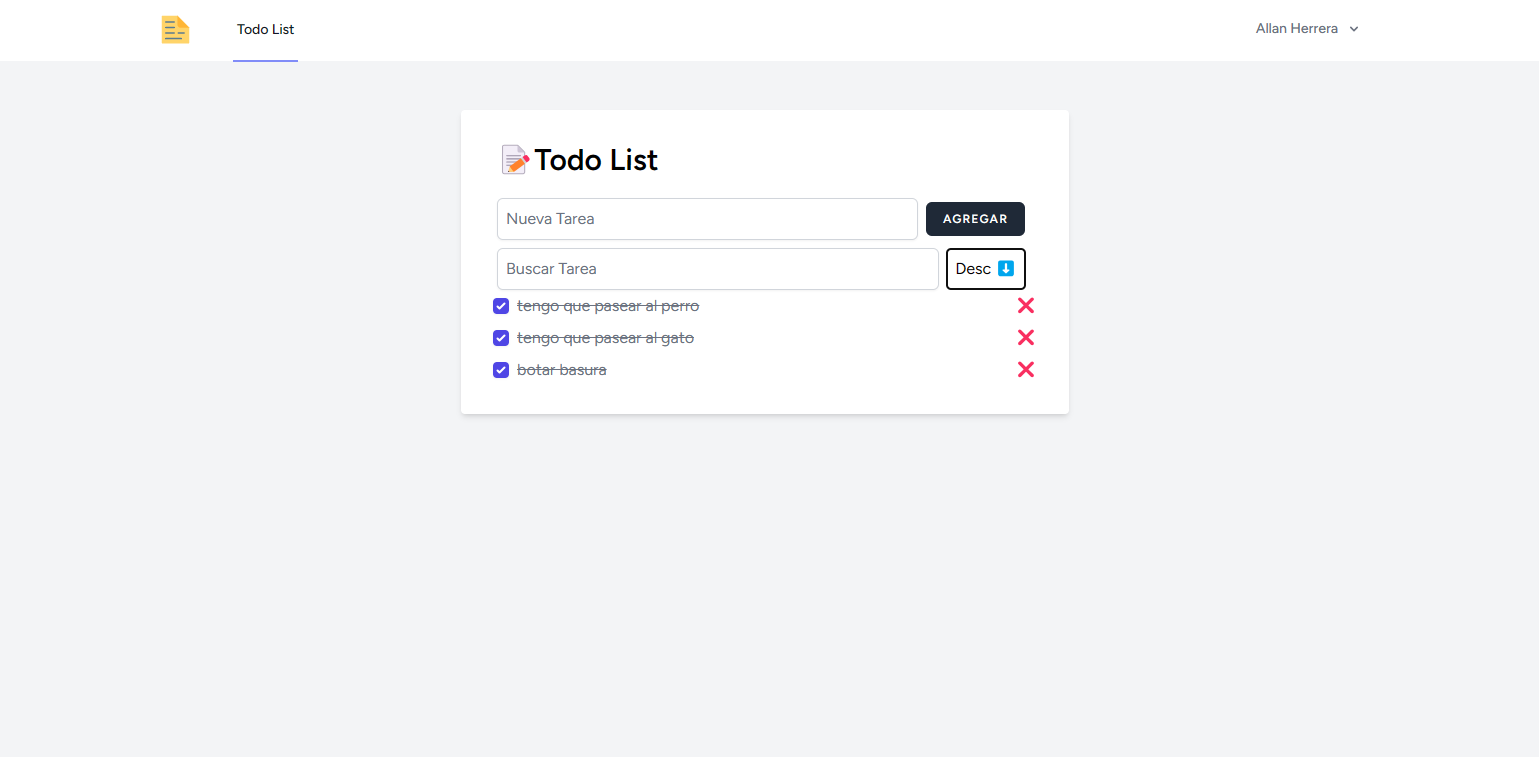


Ilustración 10 Tarea de otro usuario (ordenada de forma descendente)

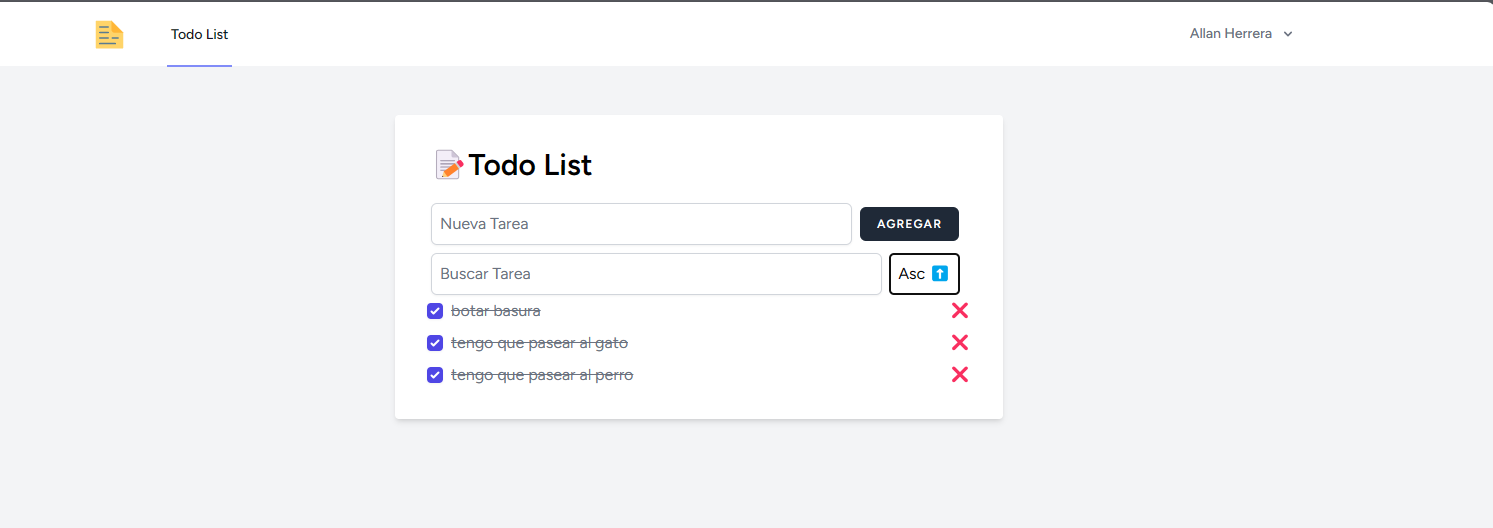


Ilustración 11 Tareas Ordenadas de forma Ascendente

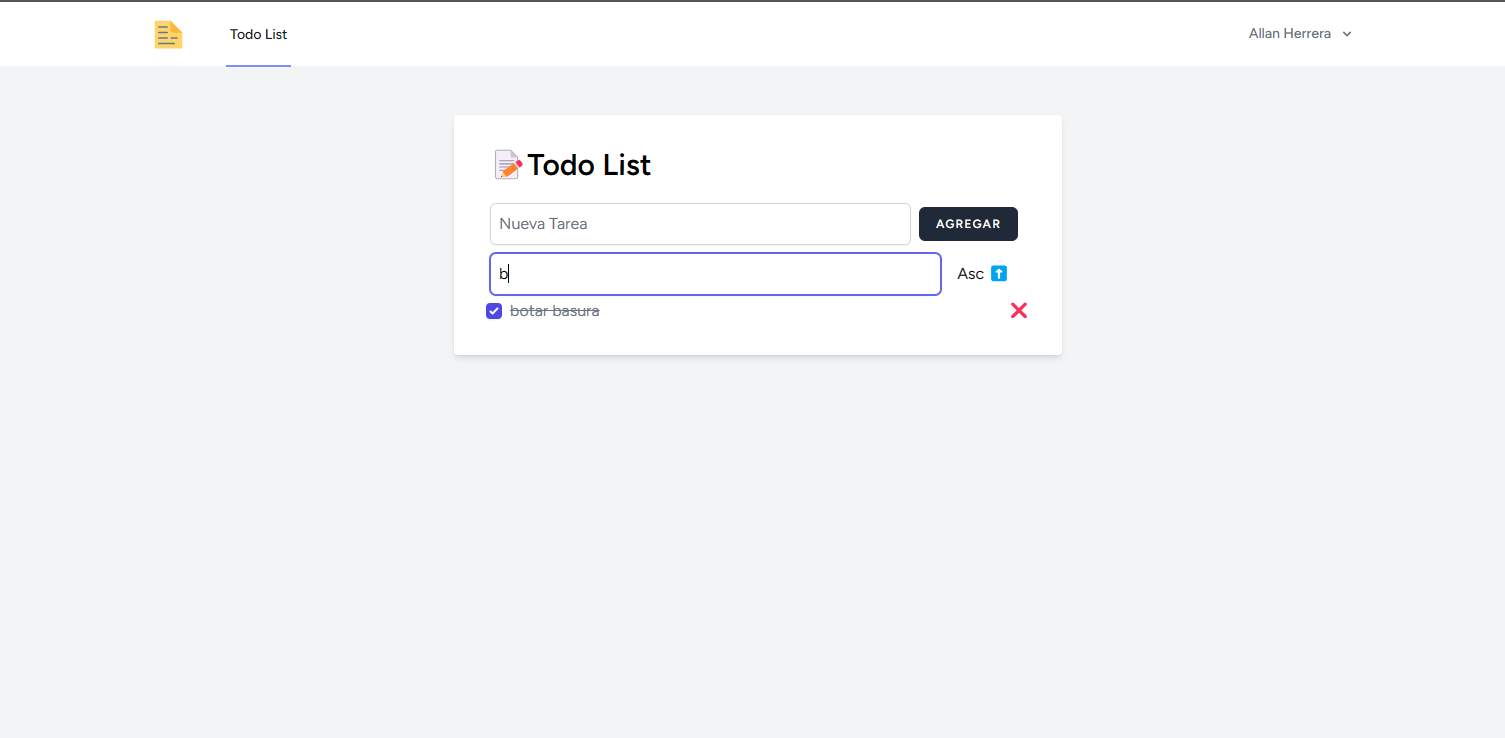


Ilustración 12 Buscando tarea por el titulo

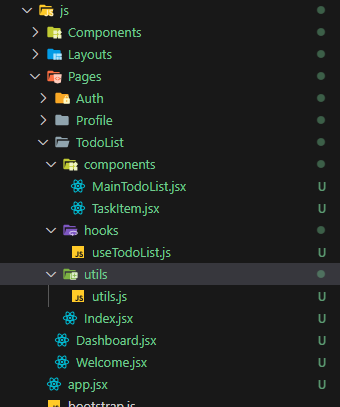


Ilustración 13 Estructura del proyecto (Frontend)

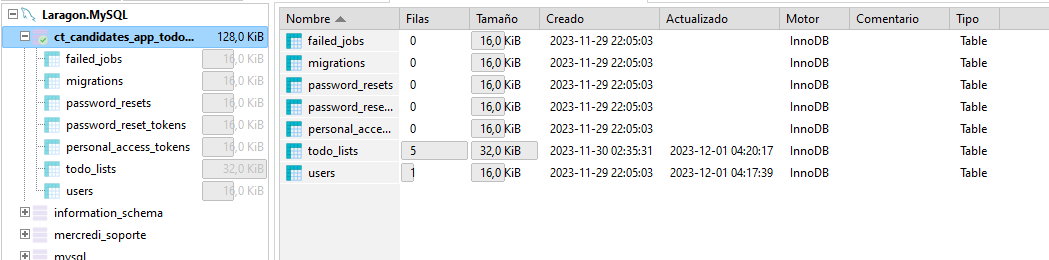


Ilustración 14 Base de datos

Repositorio: <https://github.com/axha0188/ct-candidates-app.git>