Totalges

DOCUMENTACION APP

GIOM ASesorias | Oscar Gonzalez

App WEb – ADministracion Deportiva.

Academias de Futbol

2021

**Autor del Documento**

Cristian Gonzalez – Ingeniero en Informática

**Datos de Contacto Empresa**

Email:

Página Web : www.Giom.cl

Teléfono :

**versión del Documento**

2.0

Fecha : 11-02-2021

Indice

[Capítulo 1.- Requerimientos Iniciales. 5](#_Toc63956448)

[Lo que la Empresa Necesitaba. 5](#_Toc63956449)

[Lo que Ofrecemos. 6](#_Toc63956450)

[Capítulo 2.- Estructura Organizacional 7](#_Toc63956451)

[Organigrama 7](#_Toc63956452)

[Descripción de Cargos 7](#_Toc63956453)

[SuperAdministrador 7](#_Toc63956454)

[Administrador 7](#_Toc63956455)

[Profesor 8](#_Toc63956456)

[Alumno 8](#_Toc63956457)

[Apoderado 8](#_Toc63956458)

[Capítulo 3.- Foda Inicial (antes del desarrollo de la app) 9](#_Toc63956459)

[Análisis Interno 9](#_Toc63956460)

[Fortaleza 9](#_Toc63956461)

[Debilidades 9](#_Toc63956462)

[Análisis Externo 9](#_Toc63956463)

[Oportunidades 9](#_Toc63956464)

[Amenazas 9](#_Toc63956465)

[Capítulo 4.- Marco Conceptual y Metodológico 10](#_Toc63956466)

[Ciclo de Vida 10](#_Toc63956467)

[Método de Captura de Datos 11](#_Toc63956468)

[Toma de Requerimientos. 11](#_Toc63956469)

[Reuniones Semanales. 11](#_Toc63956470)

[Observación Directa 11](#_Toc63956471)

[Capítulo 5.- Objetivos 12](#_Toc63956472)

[Objetivo General 12](#_Toc63956473)

[Objetivos Específicos 12](#_Toc63956474)

[Capítulo 6.- Foda Actual (una vez desarrollada la app) 13](#_Toc63956475)

[Análisis Interno 13](#_Toc63956476)

[Fortalezas 13](#_Toc63956477)

[Debilidades 13](#_Toc63956478)

[Análisis Exterior 13](#_Toc63956479)

[Oportunidades 13](#_Toc63956480)

[Amenazas 13](#_Toc63956481)

[Capítulo 7.- Diseño Lógico de la App 14](#_Toc63956482)

[Diagramas de Flujo 14](#_Toc63956483)

[Sitio Web 15](#_Toc63956484)

[Super Administrador 16](#_Toc63956485)

[Administrador 17](#_Toc63956486)

[Profesor 18](#_Toc63956487)

[Alumnos 19](#_Toc63956488)

[Apoderado 19](#_Toc63956489)

[Casos de Uso 20](#_Toc63956490)

[SuperAdministrador 20](#_Toc63956491)

[Administrador 20](#_Toc63956492)

[Profesor 21](#_Toc63956493)

[Alumno 21](#_Toc63956494)

[Apoderado 22](#_Toc63956495)

[Capítulo 8.- Elección de Lenguaje y Framework de Desarrollo 23](#_Toc63956496)

[AdminLTE3 1](#_Toc63956497)

[Spatie 3](#_Toc63956498)

[Capítulo 9.- Código de la App 5](#_Toc63956499)

[Modelo 5](#_Toc63956500)

[Como Crearla. 5](#_Toc63956501)

[Estructura de un Model 5](#_Toc63956502)

[Vista 6](#_Toc63956503)

[Estructura de una vista 6](#_Toc63956504)

[Como Llamar una vista 9](#_Toc63956505)

[Controladores. 10](#_Toc63956506)

[Como crear un controlador. 10](#_Toc63956507)

[Estructura de un controlador 11](#_Toc63956508)

[Capitulo 10.- Views de la App 14](#_Toc63956509)

[Páginas de Inicio 14](#_Toc63956510)

[Welcome 15](#_Toc63956511)

[Escuelas 15](#_Toc63956512)

[Contacto 16](#_Toc63956513)

[Acerca de… 16](#_Toc63956514)

[Dentro de la App 17](#_Toc63956515)

[SuperAdministrador/Escuelas 17](#_Toc63956516)

[SuperAdministrador/Usuarios 18](#_Toc63956517)

[Administrador/Perfil 19](#_Toc63956518)

[Administrador/Registro Profesores 19](#_Toc63956519)

[Administrador/Registro Alumnos-Apoderados 20](#_Toc63956520)

[Administrador/Categorias 21](#_Toc63956521)

[Profesor/Perfil 22](#_Toc63956522)

[Profesor/Tomar Asistencia 22](#_Toc63956523)

[22](#_Toc63956524)

[Profesor/Ver Asistencia 23](#_Toc63956525)

[Profesor/Tomar Rendimiento 24](#_Toc63956526)

[Profesor/Nomina Jugadores 25](#_Toc63956527)

[Alumno/Perfil 25](#_Toc63956528)

[Apoderado/Perfil 26](#_Toc63956529)

# Capítulo 1.- Requerimientos Iniciales.

## Lo que la Empresa Necesitaba.

La empresa en este caso particular se encarga de Administrar Academias de Futbol en diferentes sectores del país, se encargan de todo lo que se refiere a administración tanto deportiva como gerencial.

Nos comentaron en un inicio que la principal intensión que tenían de generar la App era la de facilitar en gran medida la parte de gestión de las academias. Por lo que se necesitaría un nivel de jerarquía de usuarios y roles y que cada uno de estos tuviese permisos y responsabilidades con la aplicación.

Los principales roles y por lo tanto los principales usuarios que tendrían acceso a esta aplicación se separarían de la siguiente manera:

1. Superadministrador
2. Administradores
3. Profesores
4. Alumnos
5. Apoderados

Ya detallaremos en estos y sus funcionalidades más adelante.

Otro de los principales requerimientos establecidos fue que al ser una app web esta fuese diseñada de forma que sea responsive; queriendo decir con esto que la app sea utilizable tanto en ordenador, Tablet, smartphone o cualquier otro dispositivo que cuente con pantalla y navegador web.

## Lo que Ofrecemos.

Tomando en cuenta todos los requerimientos obtenidos se comenzó un estudio e investigación respecto de cómo debería verse, funcionar y manejarse la app web solicitada.

A lo que concluimos que la mejor opción sería crear un sitio web sencillo y amigable a la vista en la cual pudiese navegar cualquier tipo de persona, “esto se decidió de esta manera para poder atraer y tener contacto con nuevos clientes potenciales”, la web en sí misma no sería más que la cara visible de lo que sería la app en donde mostraremos las actuales academias que trabajan con nosotros, una página de contacto, y una página con información acerca de nuestra empresa.

Dentro de esta página web existe un botón de **LOGIN** en principio será la que nos conceda el acceso a las funciones de la app, esto claro solo para usuarios que ya nosotros hayamos aceptado, negociado e ingresado a la app.

Dentro de la app y dependiendo de tu rol dentro de la jerarquía de cada academia tendrás acceso a una u otras funciones que ya detallaremos más adelante en este documento.

# Capítulo 2.- Estructura Organizacional

Es en este punto donde ya comenzamos a hablar de manera más técnica y profesional sobre las funciones, roles, permisos y capacidades que tiene la app para su uso.

## Organigrama

## Descripción de Cargos

### SuperAdministrador

Este Rol / Usuario es básicamente el encargado de administrar la app, por lo cual posee todas las funcionalidades de los demás Usuarios / Roles, además puede registrar nuevas academias que requieran trabajar con la app. También es capaz de registrar nuevos superadministradores y administradores y asignarlos a las nuevas academias.

También es el único usuario que puede acceder al panel de administración en donde se mostrara información sobre usabilidad de la app en cuestión.

### Administrador

Este Rol / Usuario es quien estará encargado de una o más academias según requiera su cargo, este usuario dentro de la app es quien puede registrar a los profesores que trabajaran en la academia, registrar las categorías y asignarlas a su profesor encargado, registrar a los alumnos y asignarlos a su debida categoría correspondiente y registrar a los apoderados quienes a su vez estarán ligados a un alumno para su gestión, ya sea en términos de permisos como monetaria y administrativa.

También este Rol / Usuario será capaz de asignarle el pago de productos, matrículas o mensualidades a los alumnos registrados en escuelas. Y también podrá realizar la administración deportiva propia del Usuario / Rol Profesor, como gestionar la asistencia, evaluar rendimiento o hacer nóminas de alumnos para eventos.

### Profesor

Este Rol / Usuario es quien estará dentro de la academia que le corresponda, a cargo de una o más categorías de alumnos, y será el encargado de llevar constancia de la parte deportiva de los alumnos de sus categorías.

Este usuario tendrá también funciones administrativas como la de tomar asistencia de sus clases, realizar una toma de rendimiento de los alumnos que tenga a su cargo y también será el encargado de registrar y gestionar las nóminas de jugadores que participaran en ciertos eventos que se vayan a realizar y que la academia vaya a participar a lo largo del año.

Los cambios y avances de rendimiento de los alumnos de cada una de las categorías que tenga a si cargo serán de única responsabilidad de este usuario.

### Alumno

Este Rol / Usuario es quien estará inscrito en alguna de las academias y será parte de alguna de las categorías de esta.

Dentro de la app este usuario podrá ver su avance en términos de rendimiento deportivo.

### Apoderado

Este Rol / Usuario es básicamente la persona que dentro de la academia y de la app misma estará encargado como tutor de uno o más alumnos y será el responsable de presentar la documentación necesaria para que este anteriormente mencionado pueda ser incluido tanto en la academia como en la app.

También será el encargado de presentar la documentación medica en caso de ser necesaria para así evitar posibles accidentes a la hora de realizar las actividades.

A su vez este usuario tendrá la opción de ver su historial de pagos o atrasos dentro de la academia en función de los alumnos que tenga a su cuidado.

# Capítulo 3.- Foda Inicial (antes del desarrollo de la app)

## Análisis Interno

### Fortaleza

* Iniciativa y ganas de mejorar en los procesos educativos presentes en las academias
* Iniciativa y ganas de mejorar en los procesos administrativos presentes en las academias
* Infraestructura necesaria para el soporte de una plataforma online
* Contante capacitación del personal que lo requiera

### Debilidades

* Procesos de evaluación se registrar de manera manual lo cual lo vuelve lento y tedioso
* Falta de automatización en los procesos administrativos.

## Análisis Externo

### Oportunidades

* Expandir el alcance de academias que trabajan con nosotros.
* Adquirir Sponsors que ayuden a financiar continuas actualizaciones a la app
* Amplio margen de interés por parte de las academias de futbol en automatizar su administración

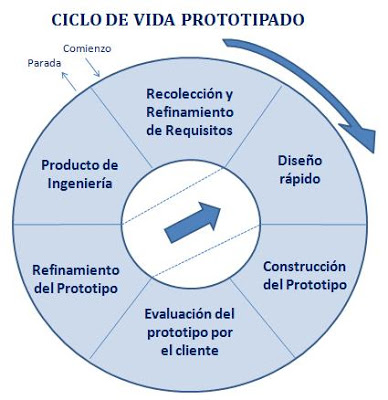
## Amenazas

* Variaciones de la planificación de la app
* Posible falta inicial de capacitaciones para los usuarios

# Capítulo 4.- Marco Conceptual y Metodológico

## Ciclo de Vida

El ciclo de vida que decidimos utilizar para este desarrollo es el de **PROTOTIPADO**, está basada en la creación de prototipos del producto final, para demostrar al cliente los avances efectuados y si satisface los requerimientos dados. Debido a que no es necesario tener un claro domino o conocimiento de los requerimientos que se necesiten satisfacer, este ciclo, permite corroborar a través de prototipos el avance del producto, y cambiarlo, mejorarlo, o modificarlo según los nuevos requerimientos que puedan aparecer.



## Método de Captura de Datos

### Toma de Requerimientos.

Antes de comenzar la planificación y desarrollo de la app nos reunimos con los encargados para tomar los requerimientos iniciales para la app dentro de los cuales obtuvimos la jerarquía que funcionaria dentro de la app, también se planifico un modelo de datos (Base de Datos) inicial y se especificó las funcione y permisos que tendría cada uno de los usuarios.

### Reuniones Semanales.

Se acordó que 2 veces por semana se debían realizar reuniones en conjunto , los encargados junto al equipo de desarrollo en los cuales se presentarían avances y prototipos iniciales para su posterior corrección “en caso de ser necesaria”.

En estas reuniones se mostrarán los módulos a desarrollar, así como también se planteará y explicará la metodología para implementar ciertos componentes.

### Observación Directa

Se observo el comportamiento de algunas academias de futbol para tomarlas como referencia a lo que debía realizar las funciones de la app. Además de múltiples Explicaciones por parte de los encargados con ejemplos claros de lo que se vive día a día en la administración de estas academias.

# Capítulo 5.- Objetivos

## Objetivo General

El objetivo General es el desarrollo de un sistema informático cuya función es brindar apoyo a la gestión de academias de futbol. Que permita el control, seguimiento de indicadores, entrenamiento, ejercicios y actividad física realizada en las clases impartidas dentro de las academias. De tal manera que automatice los procesos de toma de rendimiento, administración de asistencia, y nóminas de campeonato entre otras. Con esto se espera poder modernizar y automatizar los procesos más tediosos dentro de la administración y la gestión.

## Objetivos Específicos

* Desarrollar indicadores de gestión para las academias
* Almacenar los datos de asistencia de las clases
* Almacenar los datos de rendimiento de los alumnos
* Almacenar las nóminas de jugadores utilizadas en cada evento que se participe
* Gestionar el registro de los diferentes tipos de usuarios
* Entregar Consejo y ayuda dentro de la plataforma
* Monitorear el avance deportivo de los alumnos

# Capítulo 6.- Foda Actual (una vez desarrollada la app)

## Análisis Interno

### Fortalezas

* Sistema Moderno de gestión
* Sistema Moderno de Seguimiento
* Mejora Continua de los Módulos Utilizados

### Debilidades

* Posible mal uso de los usuarios

## Análisis Exterior

### Oportunidades

* Expansión a nivel nacional
* Negociación con sponsor interesados en colaborar

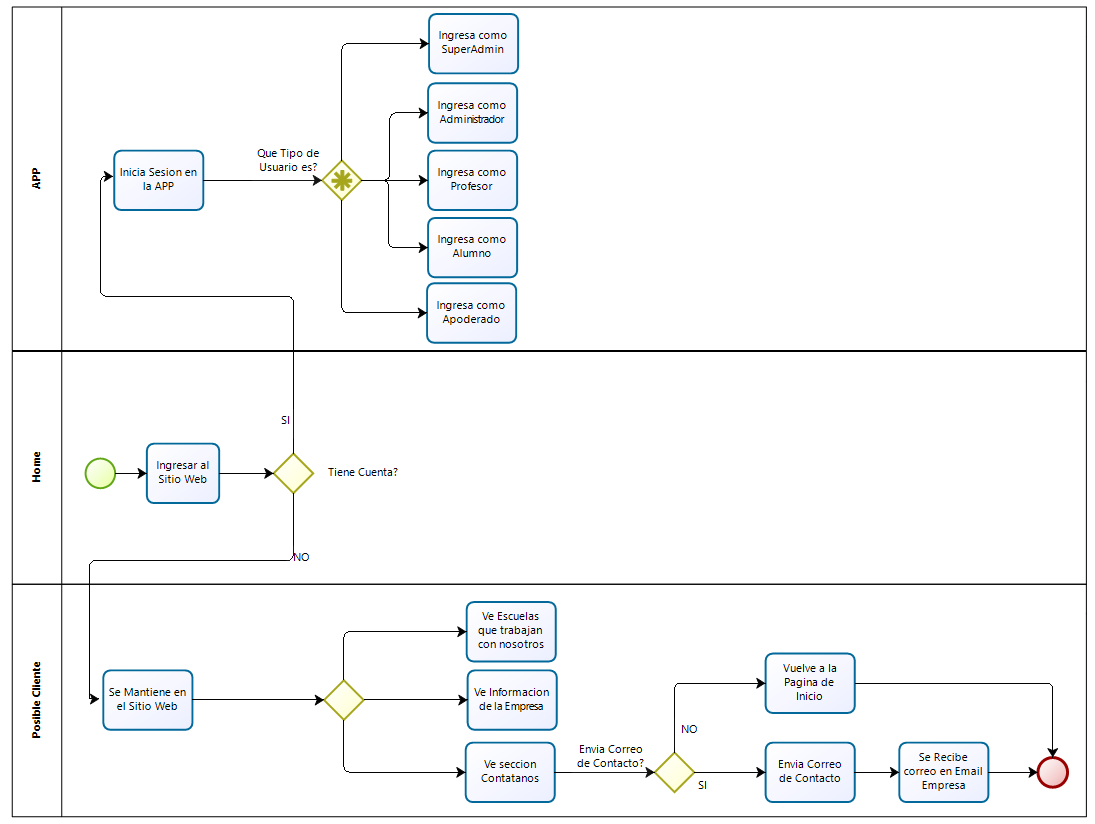
### Amenazas

* Ciber-Ataques

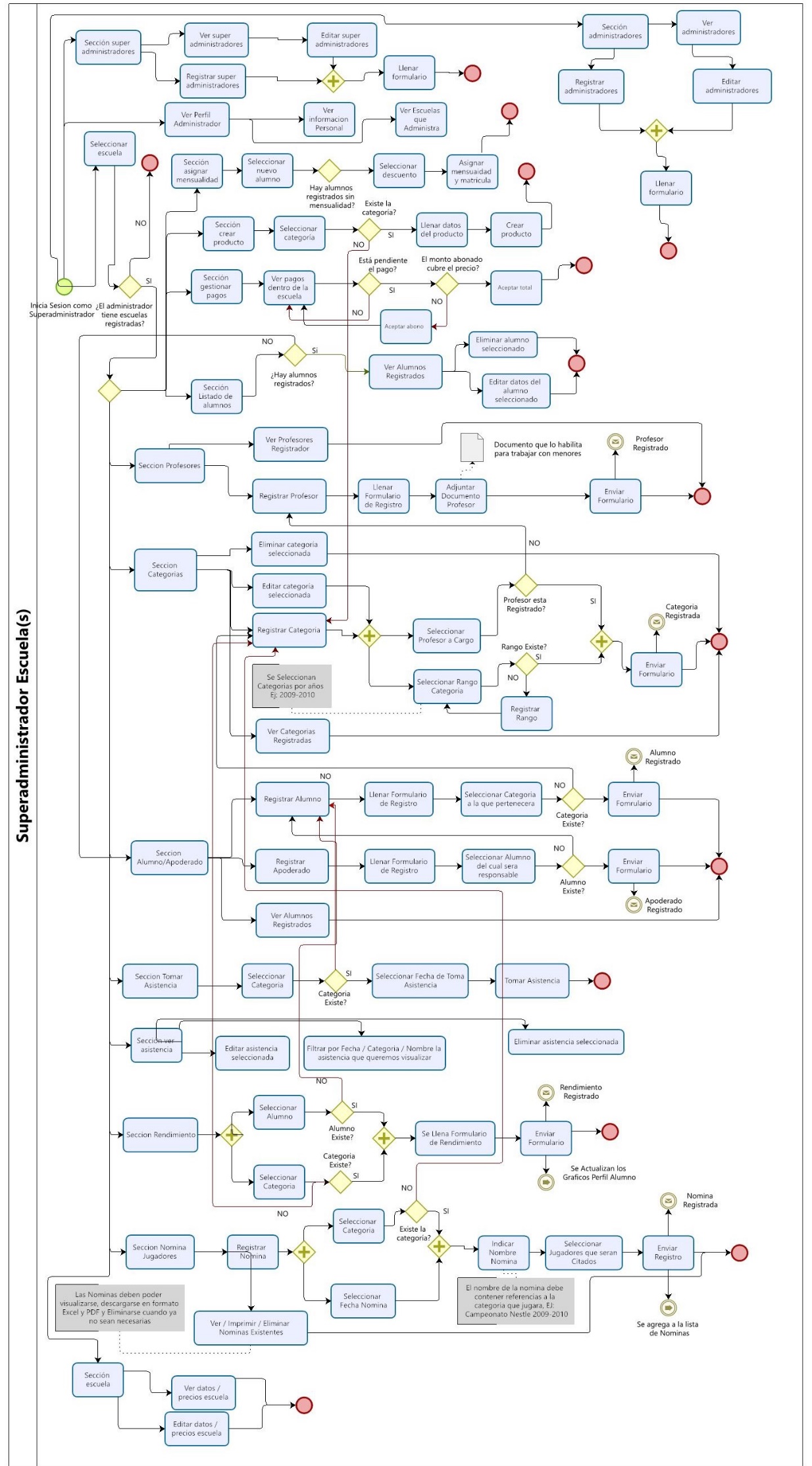
# Capítulo 7.- Diseño Lógico de la App

## Diagramas de Flujo

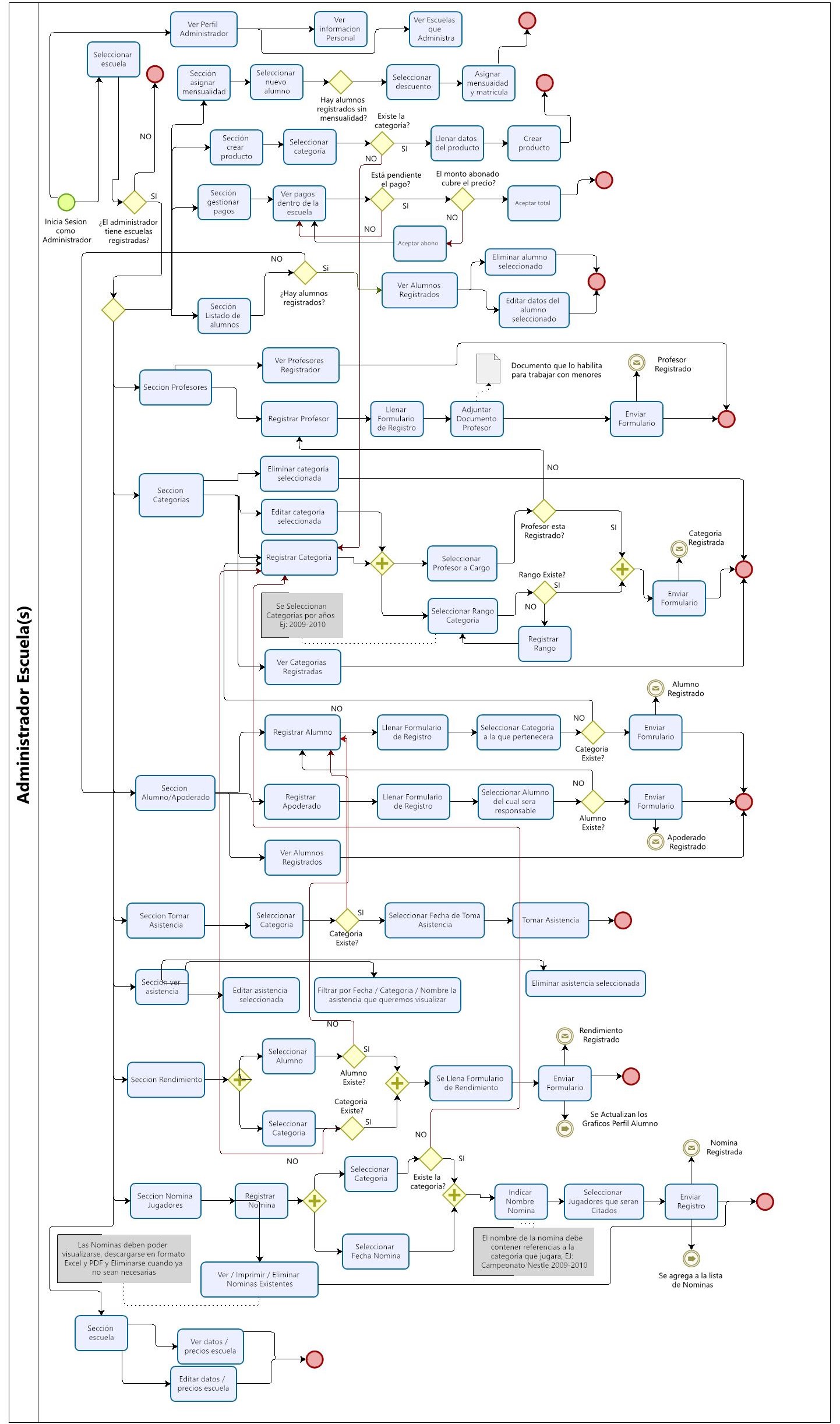
### Sitio Web



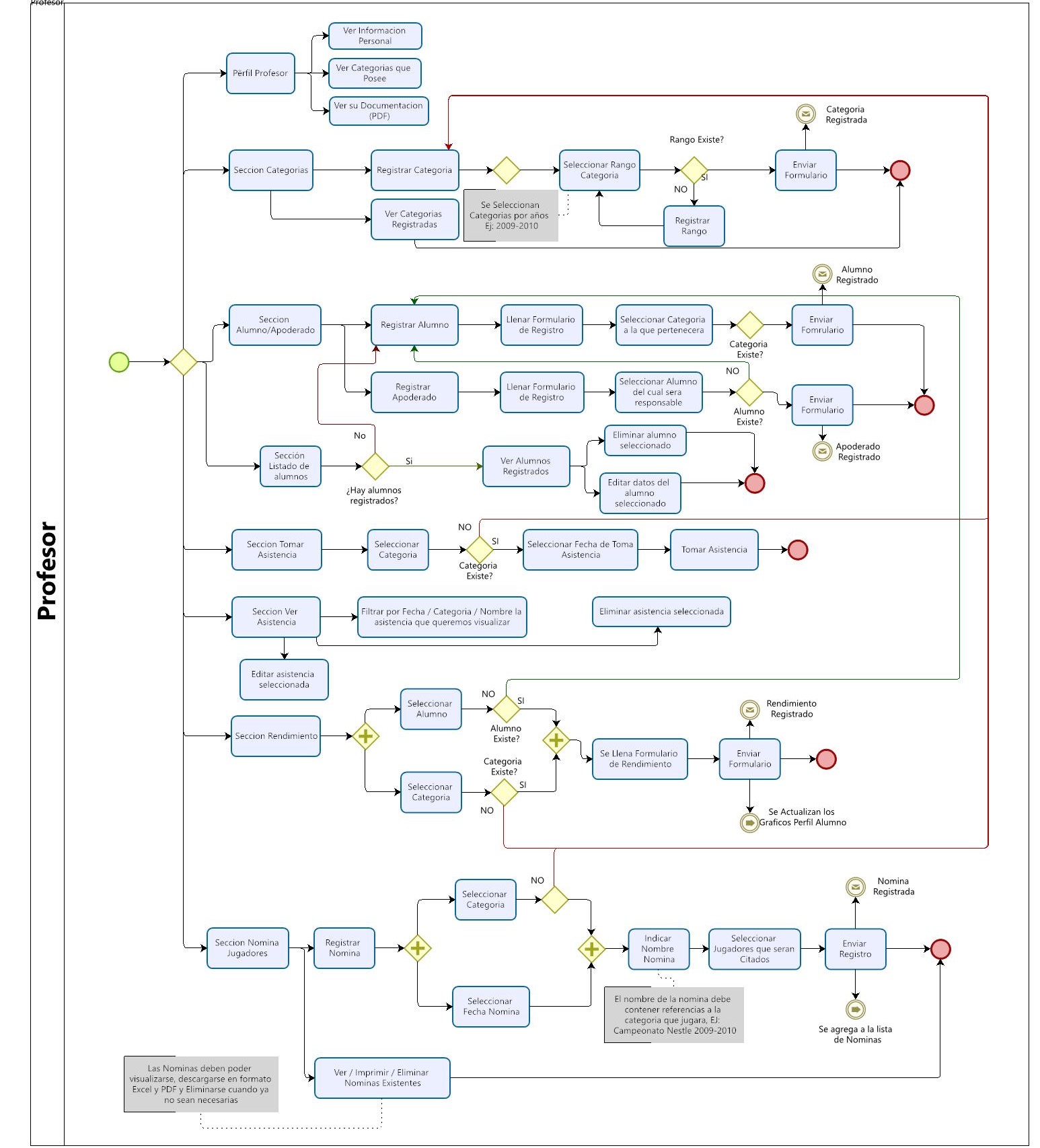
### Super Administrador



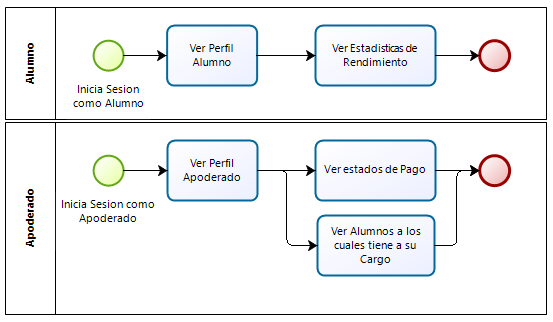
### Administrador



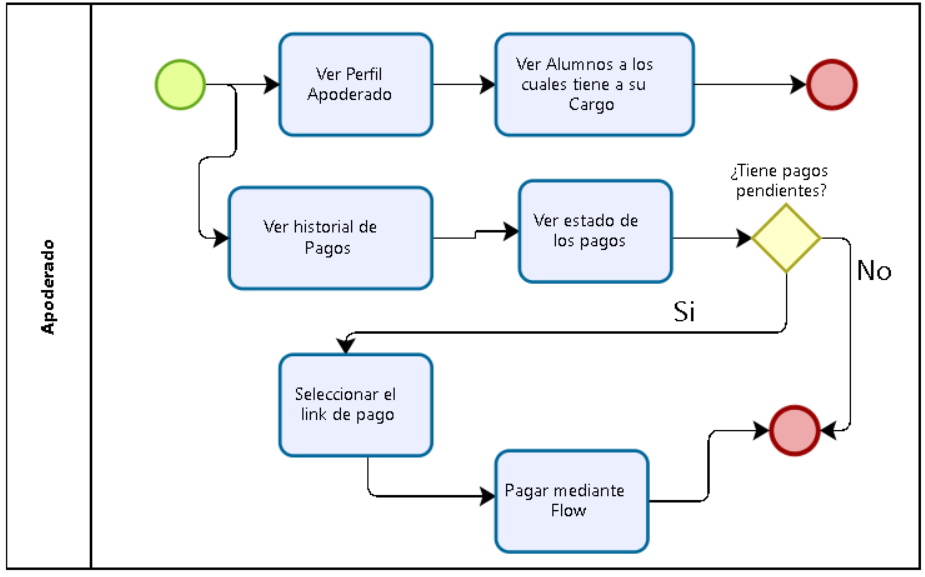
### Profesor



### Alumnos

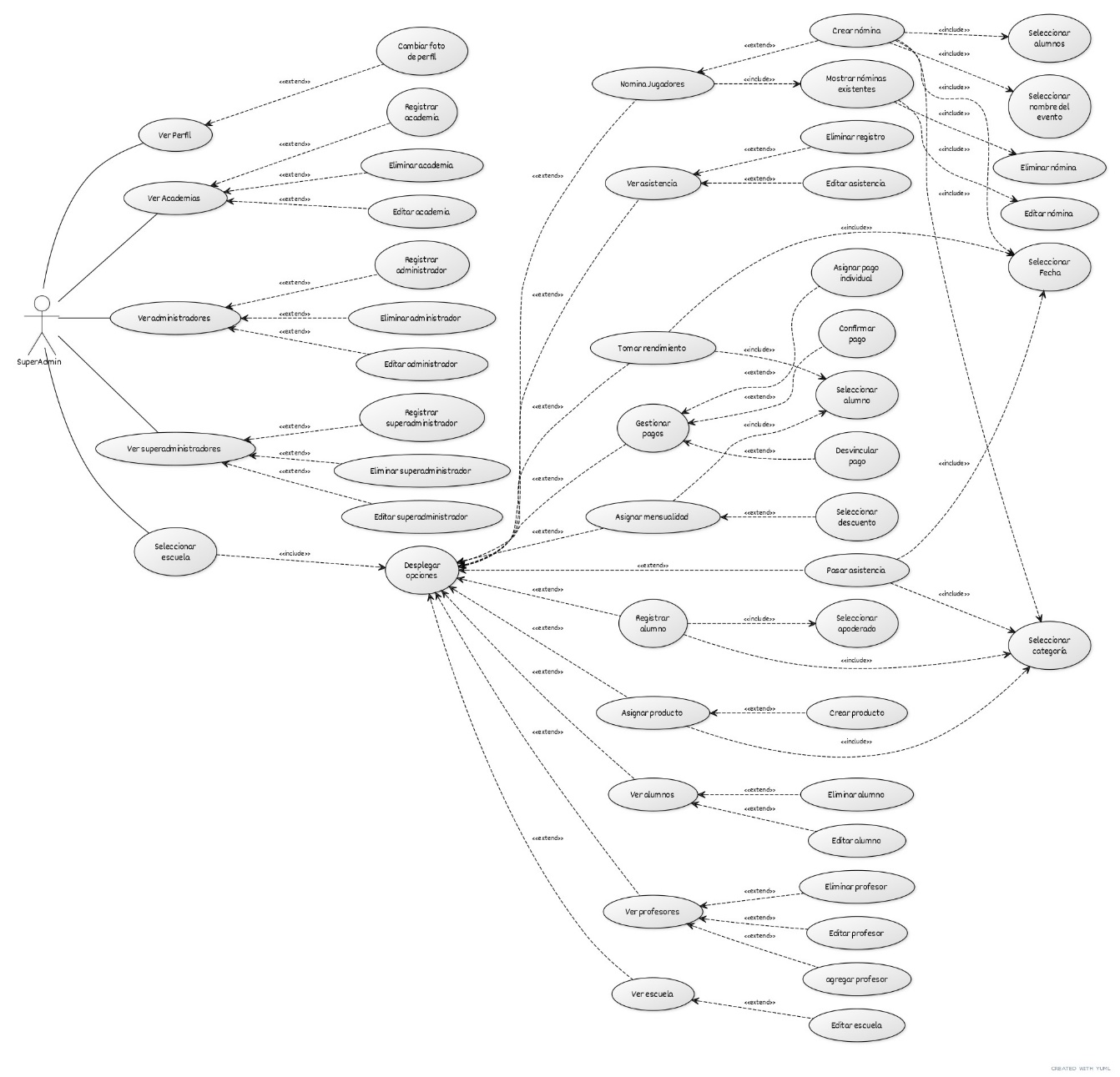


### Apoderado

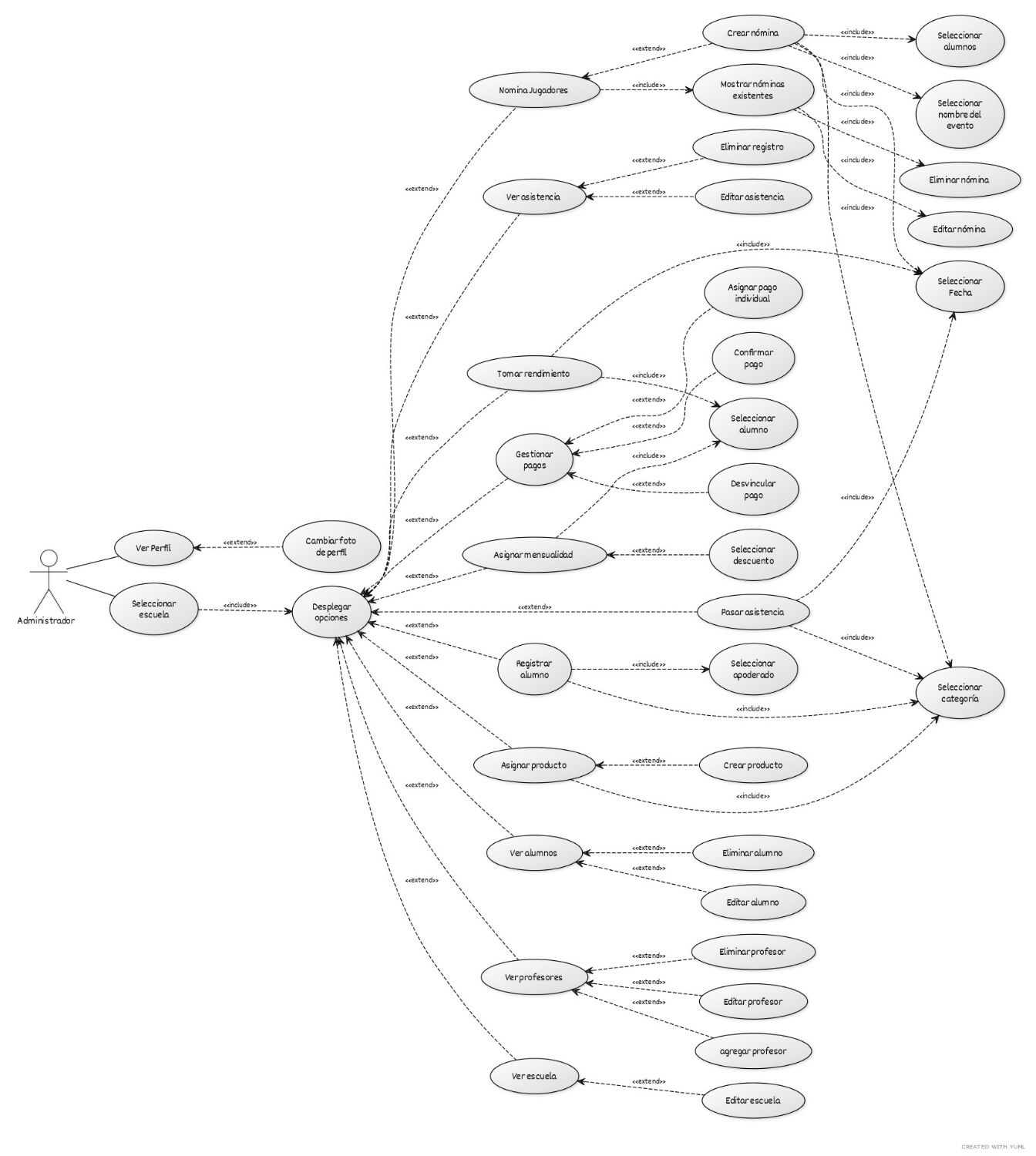


## Casos de Uso

### SuperAdministrador



### Administrador



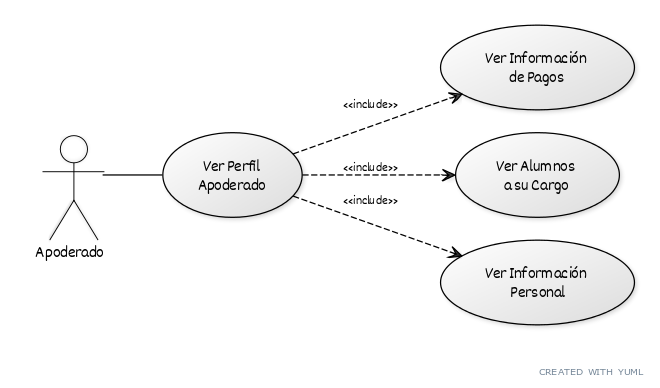
### Profesor



### Alumno



### Apoderado



# Capítulo 8.- Elección de Lenguaje y Framework de Desarrollo

El lenguaje seleccionado para la creación de esta app es **PHP**, se seleccionó este lenguaje por razones muy concretas.

* Documentación fácil de aprender y aplicar
* Comunidad amplia de desarrollo para tomar referencias
* Lenguaje 100% gratuito de utilizar
* Alto contenido de plugin y Apis creados por la comunidad
* Entre otras.

Además de seleccionar este lenguaje decidimos utilizar el frameworks por defecto más utilizado para **PHP** a nivel mundial, **LARAVEL.**

**LARAVEL** al igual que **PHP** cuenta con una amplia comunidad en la cual apoyarse y que está constantemente creando plugin y facilitando el desarrollo para los programadores.

Además de que este trabaja con el modelo de desarrollo MVC (Modelo / Vista / Controlador) el cual permite trabajar de manera más organizada y ágil las diferentes secciones y módulos que se incluirán dentro de la app.

Junto a estas herramientas de trabajo se decidió utilizar el editor de código VSCODE con una serie de extensiones que permitirá ver y manejar de manera más ágil la codificación de la app.

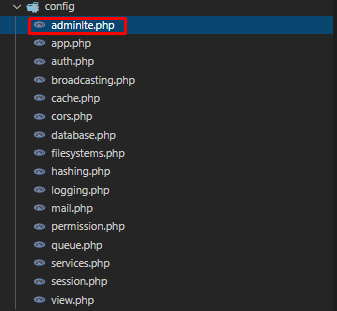
* Auto Remane Tag
* Better Comment
* Bracket Pair Colorizer
* Gitlens
* Intellisense for CSS
* Laravel Artisan
* Laravel Blade Snippets
* Laravel Snippets
* Live Serve
* PHP Intelephense
* Polacode
* Prettier
* SVG
* VS Color Picker
* VSCODE - caniuse
* VSCODE – icons
* yUML

Junto a lo antes mencionado también hemos agregado diferentes herramientas de ayuda que a continuación mencionaremos.

## AdminLTE3

Esta Herramienta es básicamente la que nos facilita la vista, uso y manejo de las Layout dentro de la app, el manejo de un menú frontal y lateral corren por parte de AdminLTE, así como también la opción de usar el método “can” para asignar los roles que podrán ver cada uno de los servicios.

Cualquier Cambio que se requiera hacer en esta herramienta se debe ubicar el archivo en la carpeta config. Como se muestra continuación.



#### Agregar Plugin en AdminLTE

'plugins' => [

        'Datatables' => [

            'active' => false,

            'files' => [

                [

                    'type' => 'js',

                    'asset' => false,

                    'location' => '//cdn.datatables.net/1.10.19/js/jquery.dataTables.min.js',

                ],

                [

                    'type' => 'js',

                    'asset' => false,

                    'location' => '//cdn.datatables.net/1.10.19/js/dataTables.bootstrap4.min.js',

                ],

                [

                    'type' => 'css',

                    'asset' => false,

                    'location' => '//cdn.datatables.net/1.10.19/css/dataTables.bootstrap4.min.css',

                ],

            ],

        ],

#### Manejo de Manu Lateral en AdminLTE

'menu' => [

        //Primera sección del Menu

        //PERFILES

        [

           'header' => 'PERFIL',

           'can' => 'Ver\_Perfil\_AdministradorEscuela'

        ],

        [

            'header' => 'PERFIL',

            'can' => 'Ver\_Perfil\_Profesor'

        ],

        [

            'header' => 'PERFIL',

            'can' => 'Ver\_Perfil\_Alumno'

        ],

        [

            'header' => 'PERFIL',

            'can' => 'Ver\_Perfil\_Apoderado'

        ],

            [

                'text' => 'Perfil Administrador',

                'url' => 'adminescuelas/perfiladmin',

                'icon' => 'fas fa-users-cog',

                'can' => 'Ver\_Perfil\_AdministradorEscuela',

            ],

            [

                'text' => 'Perfil Profesor',

                'url' => 'profesores\perfilProfesor',

                'icon' => 'fas fa-user-tie',

                'can' => 'Ver\_Perfil\_Profesor',

            ],

            [

                'text' => 'Perfil Alumno',

                'url' => 'alumnos\perfil',

                'icon' => 'fas fa-running',

                'can' => 'Ver\_Perfil\_Alumno',

            ],

            [

                'text' => 'Perfil Apoderado',

                'url' => 'apoderados/perfilApoderado',

                'icon' => 'fas fa-user-friends',

                'can' => 'Ver\_Perfil\_Apoderado',

            ],

## Spatie

Esta es una herramienta que nos permite crear y manejar los roles y permisos dentro de la app, de una manera más sencillo basándose en el uso de la creación de tablas que contengan los elementos antes mencionados y el uso de middlewares.

#### Creación de Permisos:

//LISTA DE PERMISOS PARA LA APP

        //creación de Perfiles de Usuario

        Permission::create(['name' => 'Crear\_Administrador']);

        Permission::create(['name' => 'Crear\_Profesor']);

        Permission::create(['name' => 'Crear\_Alumno\_Apoderado']);

        Permission::create(['name' => 'Crear\_Alumno\_Apoderado\_P']);

        Permission::create(['name' => 'Crear\_Categorias\_P']);

        //categoría

        Permission::create(['name' => 'Crear\_Categorias']);

        Permission::create(['name' => 'Eliminar\_Categorias']);

        //visualización de Perfiles de Usuario

        Permission::create(['name' => 'Ver\_Perfil\_Profesor']);

        Permission::create(['name' => 'Ver\_Perfil\_Alumno']);

        Permission::create(['name' => 'Ver\_Perfil\_Apoderado']);

        Permission::create(['name' => 'Ver\_Perfil\_AdministradorEscuela']);

        //administración Deportiva

        Permission::create(['name' => 'Tomar\_Asistencia']);

        Permission::create(['name' => 'Visualizar\_Asistencia']);

        Permission::create(['name' => 'Tomar\_Rendimiento']);

        Permission::create(['name' => 'Visualizar\_Rendimiento']);

        Permission::create(['name' => 'Generar\_Nomina']);

        //Escuelas

        Permission::create(['name' => 'Crear\_Escuelas']);

        Permission::create(['name' => 'Eliminar\_Escuelas']);

#### Creación de Roles:

//Roles a Crear

        $sadmin = Role::create(['name' => 'sadmin']);

        $admin = Role::create(['name' => 'admin']);

        $profesor = Role::create(['name' => 'profesor']);

        $alumno = Role::create(['name' => 'alumno']);

        $apoderado = Role::create(['name' => 'apoderado']);

#### Asignar Permisos a un Rol:

$sadmin->givePermissionTo([

            //Permisos que se le asignan al Rol :Super Administrador

            'Crear\_Administrador',

            'Crear\_Escuelas',

            'Eliminar\_Escuelas',

        ]);

#### Asignar Rol a un Usuario:

En este caso usaremos un ejemplo de un usuario no real creado con seeders.

Primero creo un usuario en este caso de tipo SuperAdministrador. Este usuario se almacenará en la tabla User de la base de datos con el ID = 1.

$superadmin=User::create([

            'name' => 'SuperAdministrador',

            'email' => 'superadmin@gmail.com',

            'password' => Hash::make('superadmin'),

            'tipo' => 'superadmin',

        ]);

Luego en mi código programo una secuencia que buscara al usuario ID = 1 dentro de la tabla User y le asignara un rol, esto se hace con las siguientes 2 líneas de Código.

$sadmin1 = User::find(1);

$sadmin1->assignRole('sadmin');

De esta forma el usuario que creamos se ligará al rol de superadministrador dentro de la base de datos y de manera automática adquirirá todos los permisos de dicho rol.

# Capítulo 9.- Código de la App

Como Mencionamos anteriormente laravel trabaja con el modelo MVC por lo que dar un buen uso a esta estructura es fundamental

## Modelo

Es donde se almacenan las características de una tabla dentro de la base de datos

### Como Crearla.

La forma más sencilla de crear un Modelo es utilizando las funciones de PHP Artisan como se muestra a continuación.

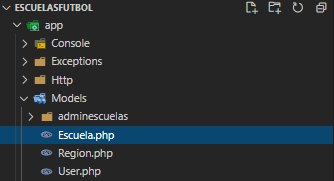
En VSCode abrimos una terminal con CTRL + Ñ, e introduciremos el siguiente Código



Esto creara el Model, pero no creara la migración de dicho Model. Para esto debemos agregar el comando -m al final de la línea de creación en la terminal.

### Estructura de un Model

Para comenzar debemos saber donde se almacenan regularmente los Model de nuestra app , para esto iremos a la ruta App/Models.



Ahora sabiendo donde están almacenados vemos su estructura.

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Escuela extends Model

{

    use HasFactory;

    public function user(){

        return $this->belongsTo('App\Models\User');

    }

    public function region(){

        return $this->belongsTo('App\Models\Region');

    }

}

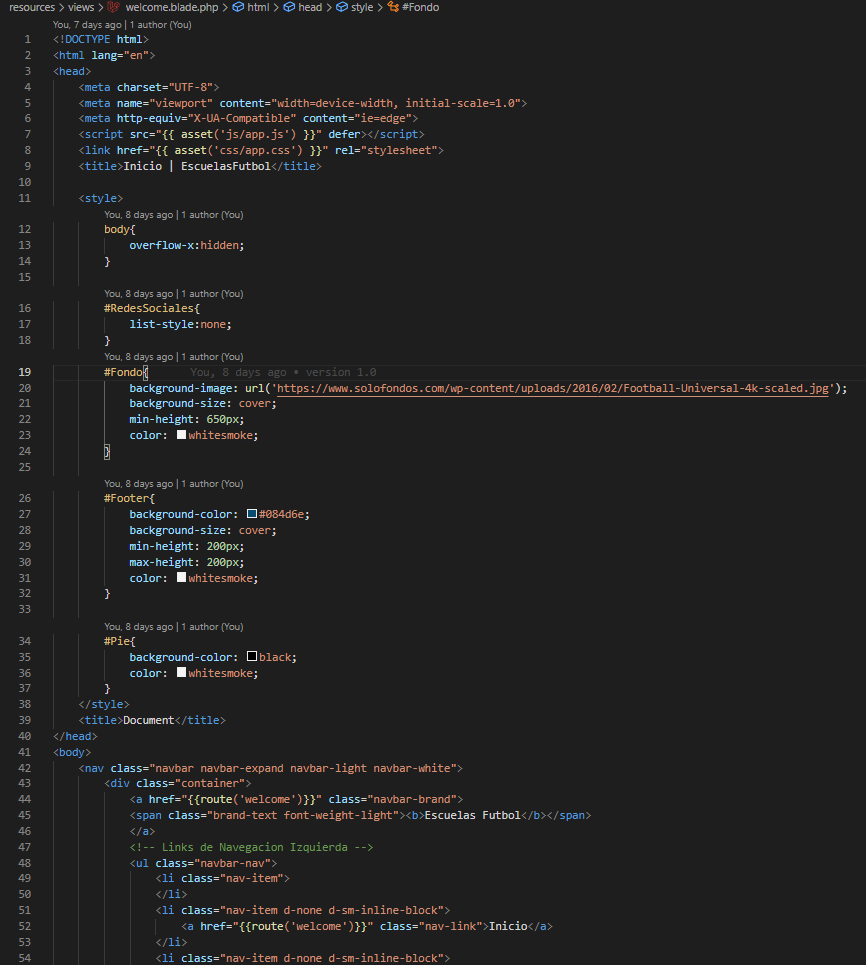
Aquí es donde declararemos la relación de las tablas.

## Vista

### Estructura de una vista

La vista es básicamente un conjunto de HTML + CSS + JS y es la cara visible al usuario de nuestra app. Para las vistas **LARAVEL** utiliza el funcionamiento de Blade que nos permite estructurar las vistas de forma diferente a como se utiliza en HTML normal. En esta app nosotros utilizamos ambas formas ya sea con HTML de siempre tanto como con Blade como se mostrará a continuación.

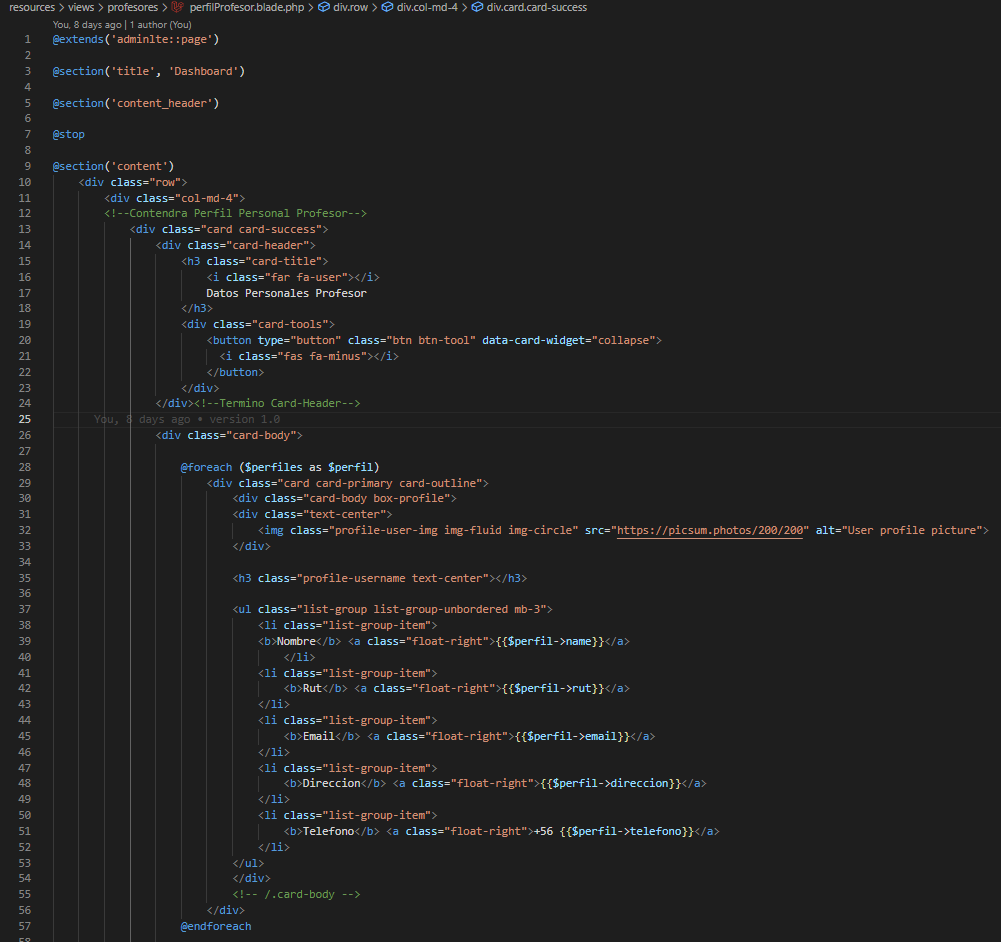
#### Pagina Welcome





Esta pagina como se puede apreciar utiliza la estructura clásica de un archivo.html

#### Pagina Perfil Profesor



Esta página utiliza el modelo de Blade y nada más comenzar a leer el archivo podemos notar que la estructura es diferente cambiando las clásicas etiquetas de <body> o de <head> por la modalidad y uso de los @section siendo reemplazados de la siguiente manera.

* <body> 🡪 @section(‘content’)
* <head> 🡪 @section(‘content\_header’)

También nos permite el uso de @extends que en este caso llama al Layout de AdminLTE explicado previamente.

### Como Llamar una vista

A diferencia de HTML normal y corriente para llamar a una vista en laravel no basta con dar la URL de la ubicación del archivo dentro de un <a href=”#”>, laravel usa el enrutamiento para mostrar los documentos y estos son a su ves manejados por controladores que ya mencionaremos más adelante.

#### Ejemplo de una Ruta:

//Rutas del welcome

Route::get('/contacto',[ContactoController::class,'index'])->name('contactanos');

Route::get('/acercadenosotros',[AcercadeNosotrosController::class,'index'])->name('acerca\_de');

Route::get('/escuelaswelcome', [EscuelasWelcomeController::class,'index'])->name('escuelaswelcome');

En este ejemplo real de la app utilizamos las rutas para llamar a todas las pestañas que aparecen en la vista Welcome.



La estructura de una ruta se define de la siguiente manera:

**Route::get** 🡪 Método de envió de datos puede ser GET o POST.

**(‘/contacto’**… 🡪 hace referencia a la ubicación del archivo vista dentro de la app.

… **,[ContactoController**…**]** 🡪 hace referencia al controlador que manejara esta ruta

… **::class , ‘índex’]** 🡪 hace referencia a la función dentro del controlador que manejara esta ruta.

**->name(‘contacto’)** 🡪 es el nombre que le asignamos a la ruta para posteriormente llamarla.

Luego de generar la estructura en nuestra vista donde necesitamos llamar a la ruta para posteriormente mostrar otra vista utilizamos el siguiente Código.

EJ: estamos en la pagina de inicio “Welcome” y queremos ir a “Contacto”

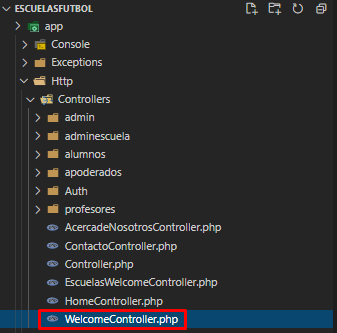
Al precio Contacto dentro del menu este tendrá detrás el comando **{{Route(‘contacto’)}}** el cual llamada a dicha ruta con este nombre y esta se encargará de contactar al controlador y posteriormente devolver la vista contacto.

## Controladores.

Los Controladores dentro de una app de Laravel son los que se encargan del funcionamiento interno o **back-end** de la app misma, las rutas recurren a estos para realizar operaciones tan simples como devolver la vista solicitada.

### Como crear un controlador.

Al igual que como vimos en el Model la mejor forma que nos indica Laravel para crear un controlador no es directamente creando un nuevo archivo dentro de la carpeta Controllers si no que es utilizando los comandos de PHP Artisan, antes de hacer esto claro, debemos saber donde se almacenan dichos controladores.



Esta siempre será la ruta por defecto de los controladores.

¿Pero bien ahora como debemos crearlos mediante la terminal? Pues de la siguiente manera.



### Estructura de un controlador

<?php

namespace App\Http\Controllers;

use App\Models\Escuela;

use Illuminate\Support\Facades\DB;

use Illuminate\Http\Request;

class WelcomeController extends Controller

{

    public function index(){

        $escuelas = DB::select('SELECT \* FROM escuelas ORDER BY RAND() LIMIT 4');

        //$escuelas = Escuela::all();

        //return view('welcome',compact('escuelas'));

        return view('welcome', ['escuelas' => $escuelas]);

    }

}

En la estrucuta básica de un controlador la función index viene por defecto, en la mayoría de los casos este se utiliza para retornar de una vista a otra y manejar las rutas.

#### Registro en un controlador.

public function register(Request $request){

            $newUser = User::create([

                'name' => $request->name,

                'email' => $request->email,

                'tipo' => Role::find(4)->name,

                'password' => Hash::make($request->password),

            ]);

            //variable almacena el id del usuario recién creado

            $id\_user = $newUser->id;

            //Almacenamos en la BD los datos del usuario

            $newUser->save();

            $newAlumno = Alumno::create([

                'rut' => $request->rut,

                'name\_alumno' => $request->name,

                'email\_alumno' => $request->email,

                'fecha\_nacimiento' => $request->date,

                'direccion' => $request->direccion,

                'telefono' => $request->telefono,

                'categorias' => $request->categoria1,

                'user' => $id\_user,

            ]);

            $newAlumno->save();

            return redirect()->back()->with('success','Alumno Registrado');

        }

Como podemos observar al registrar utilizamos modelos previamente vistos como es utilizar el Model para crear al usuario, y luego te redirecciona a la página donde estábamos mostrando un mensaje de que el registro se hizo satisfactoriamente.

Por supuesto en los controladores pueden ir funciones simples tanto como complejas , incluir Código PHP y JS.

Ejemplo de esto es el controlador de la asistencia el cual incluye filtros, llamada de datos, entre otras cosas

public function mostrar(Request $request){

        $alumnos = DB::table('alumnos')

            ->when($request, function ($query, $request) {

                for ($i = 0; $i < $request->cantidad; $i++) {

                    $ind = $i+1;

                    $query->orwhere('categorias', '=', $request->$ind);

                }

            })

            ->get();

        $asistencias = DB::table('asistencias')

            ->when($alumnos, function ($query, $alumnos) {

                for ($i = 0; $i < count($alumnos); $i++) {

                    $query->orwhere('alumno', '=', $alumnos[$i]->id);

                }

            })

            ->get();

        $datos = array();

        for ($i=0; $i < count($asistencias); $i++) {

            $asist = $asistencias[$i]->asistencia;

            $nombreAsist = '';

            switch($asist){

                case 0:

                    $nombreAsist = 'Presente';

                    break;

                case 1:

                    $nombreAsist = 'Ausente';

                    break;

                case 2:

                    $nombreAsist = 'Atrasado';

                    break;

            }

            $alumno = DB::table('alumnos')

            ->when($request, function ($query, $request) {

                for ($i = 0; $i < $request->cantidad; $i++) {

                    $ind = $i+1;

                    $query->where('categorias', '=', $request->$ind);

                }

            })

            ->orwhere('id','=',$asistencias[$i]->alumno)

            ->get();

            $categoria = DB::table('categorias')

            ->where('id','=',$alumno[0]->categorias)

            ->get();

            $rango =Rango\_Categoria::find($categoria[0]->rango);

            $edad = Carbon::parse($alumno[0]->fecha\_nacimiento)->age;

            $fecha = explode('-',$asistencias[$i]->fecha);

            $fecha = array\_reverse($fecha);

            $fecha = implode('-',$fecha);

            $arr = array();

            array\_push($arr, $alumno[0]->name\_alumno);

            array\_push($arr,$edad);

            array\_push($arr, $nombreAsist);

            array\_push($arr, $fecha);

            array\_push($arr, $categoria[0]->id);

            array\_push($arr, $rango);

            array\_push($datos,$arr);

        }

        return view('profesores/listaAsistencia',compact('datos'));

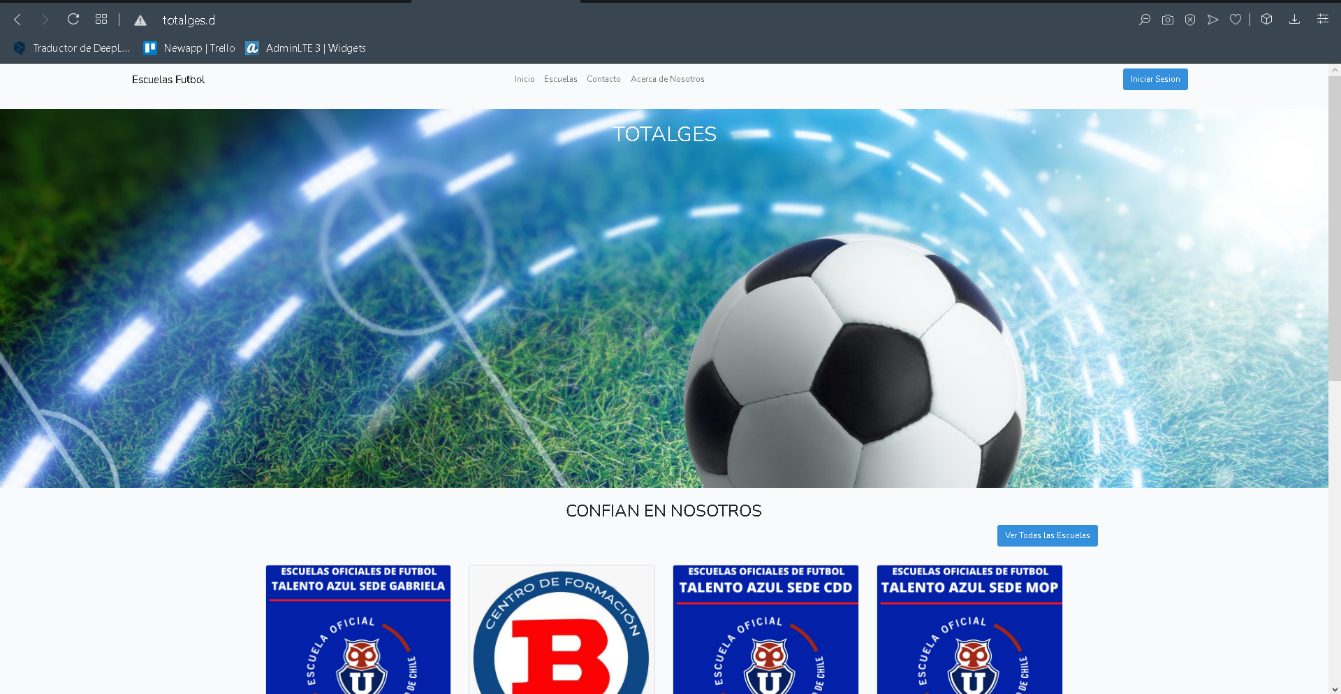
    }

Este Código mostrado nos presenta la vista de ver asistencia en la cual uno puede filtrar la información desde cualquier de los campos que aparecen.

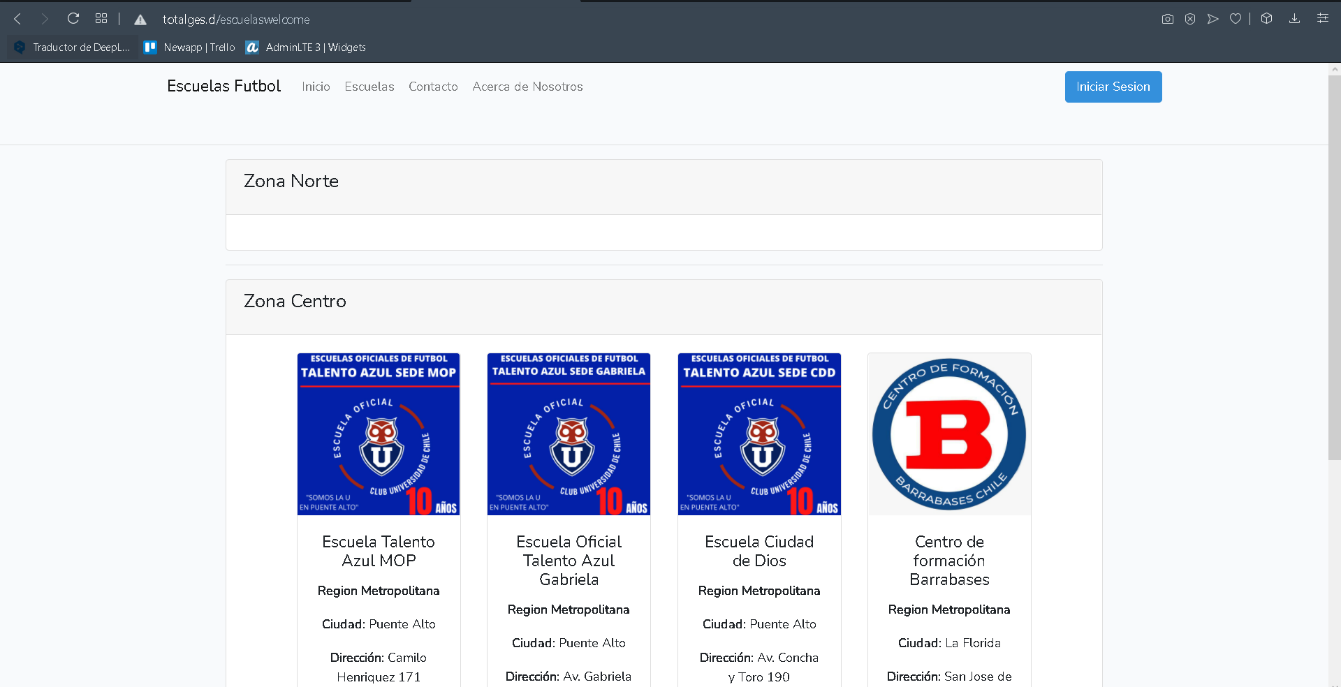
# Capitulo 10.- Views de la App

## Páginas de Inicio

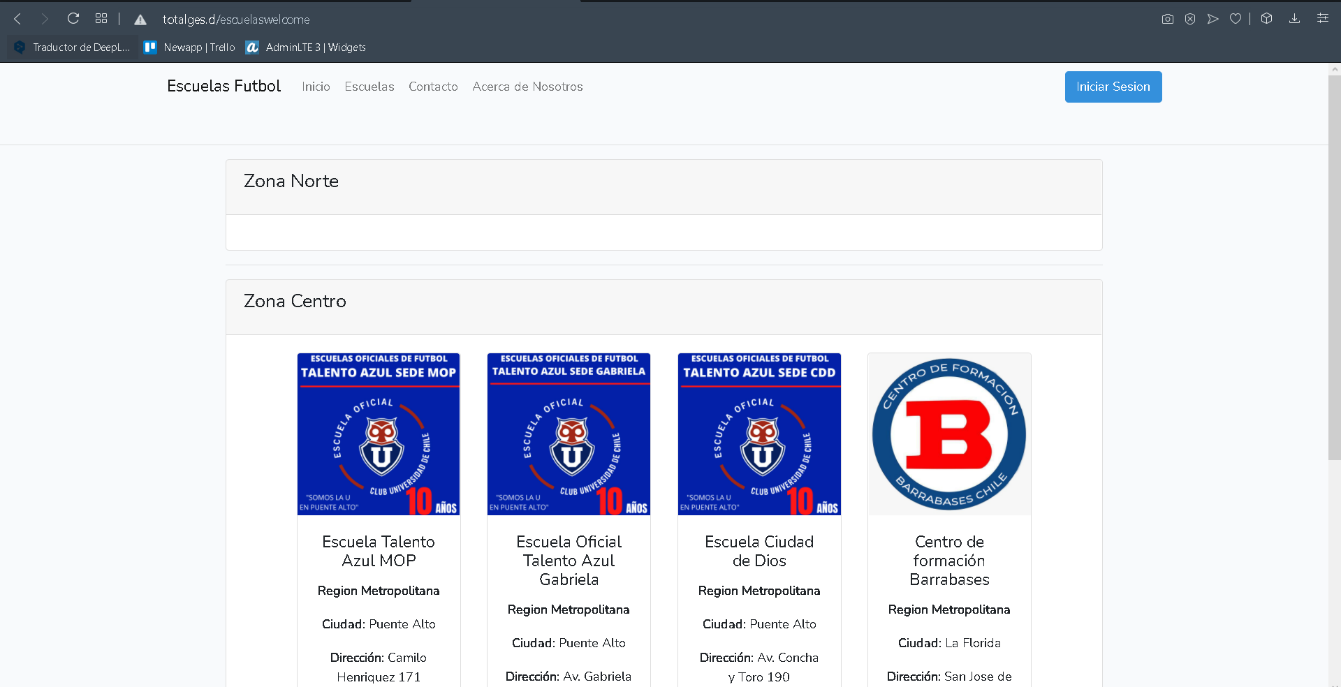
### Welcome



### Escuelas



### Contacto

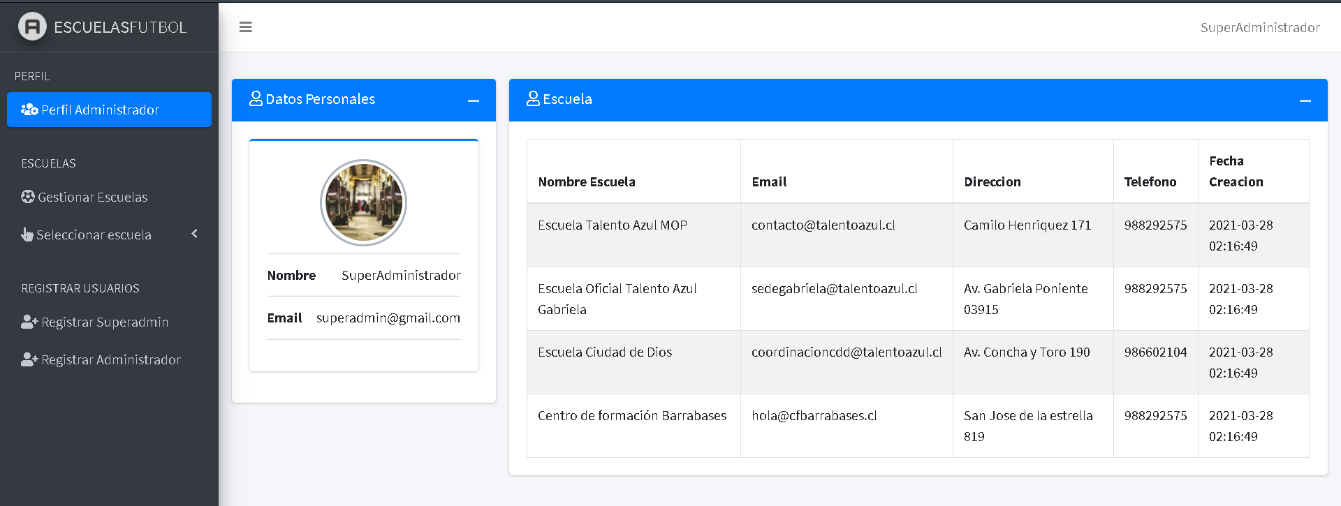


### Acerca de…

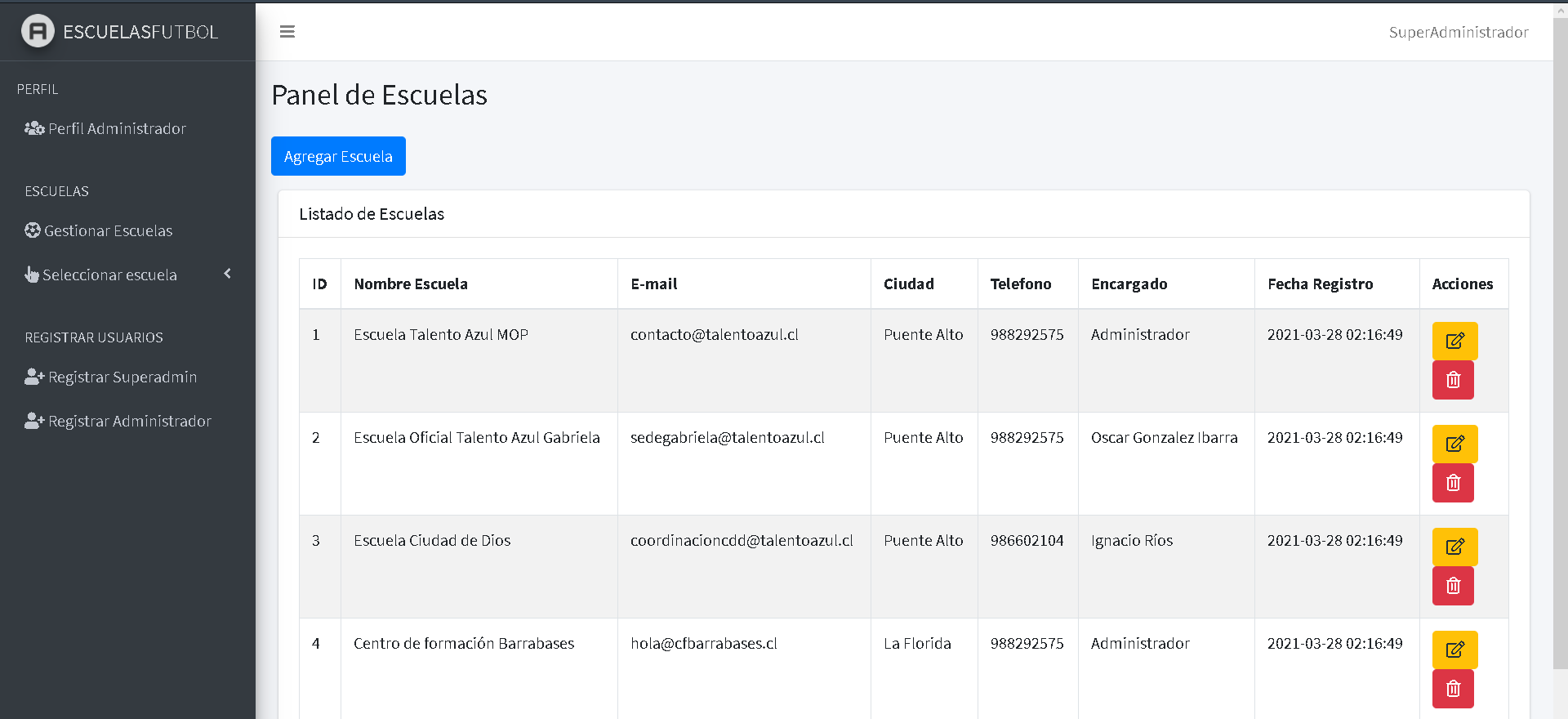


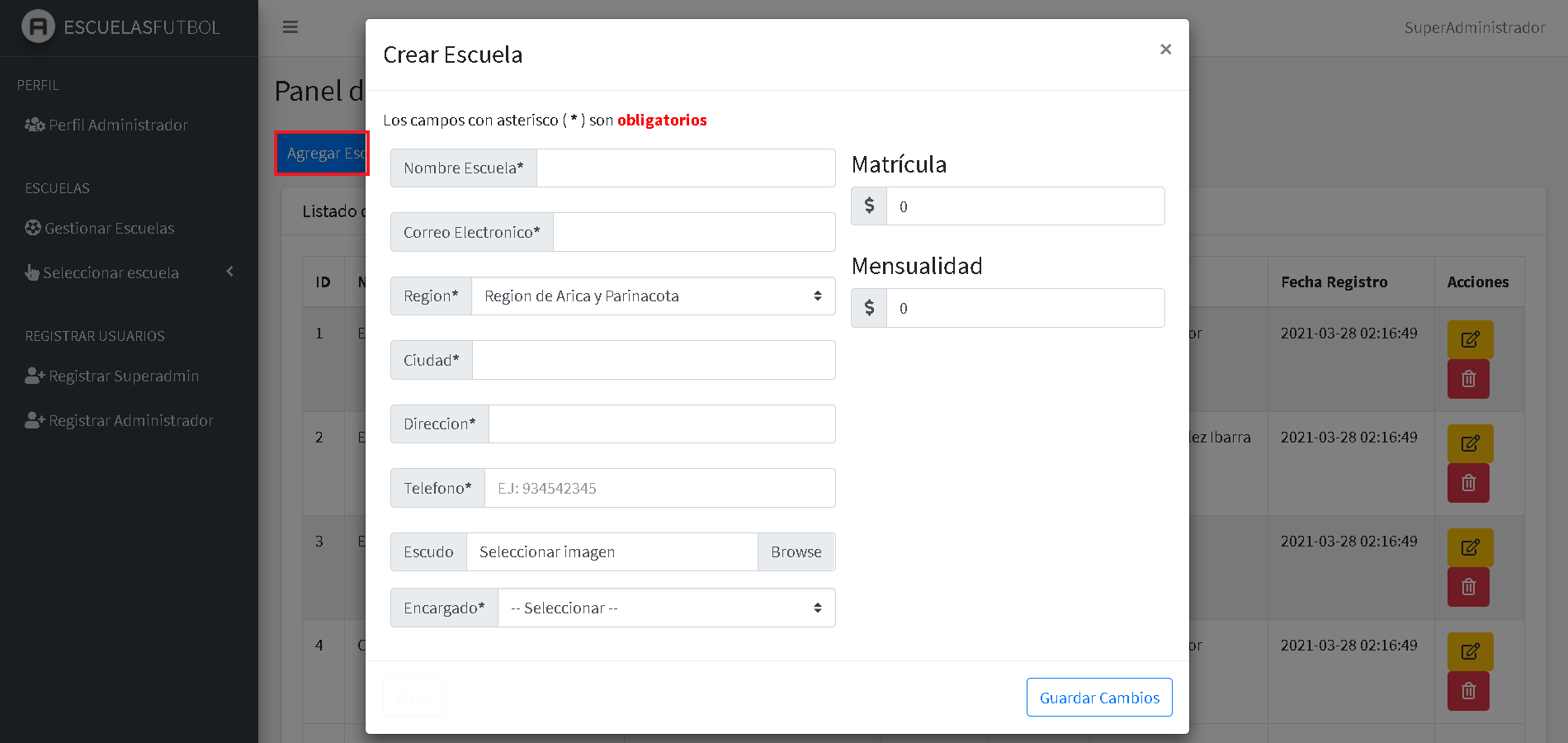
## Dentro de la App

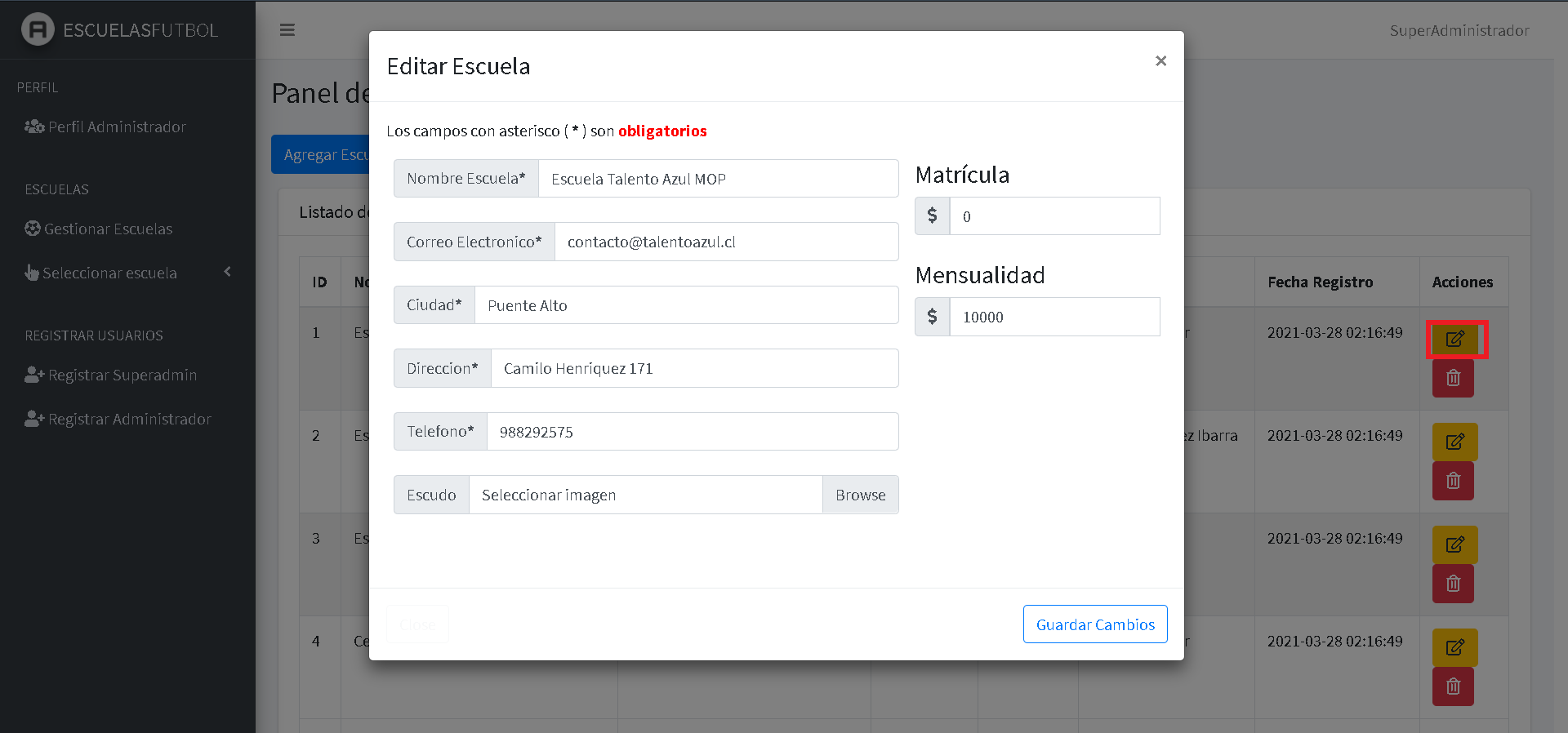
### SuperAdministrador/Perfil



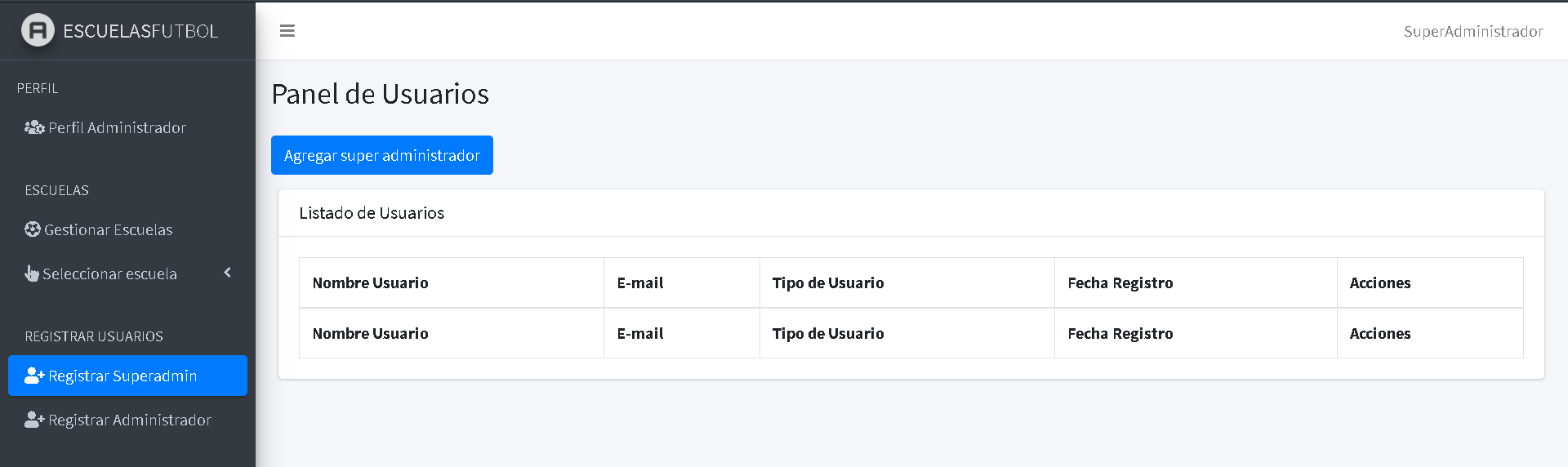
### SuperAdministrador/Gestionar escuelas



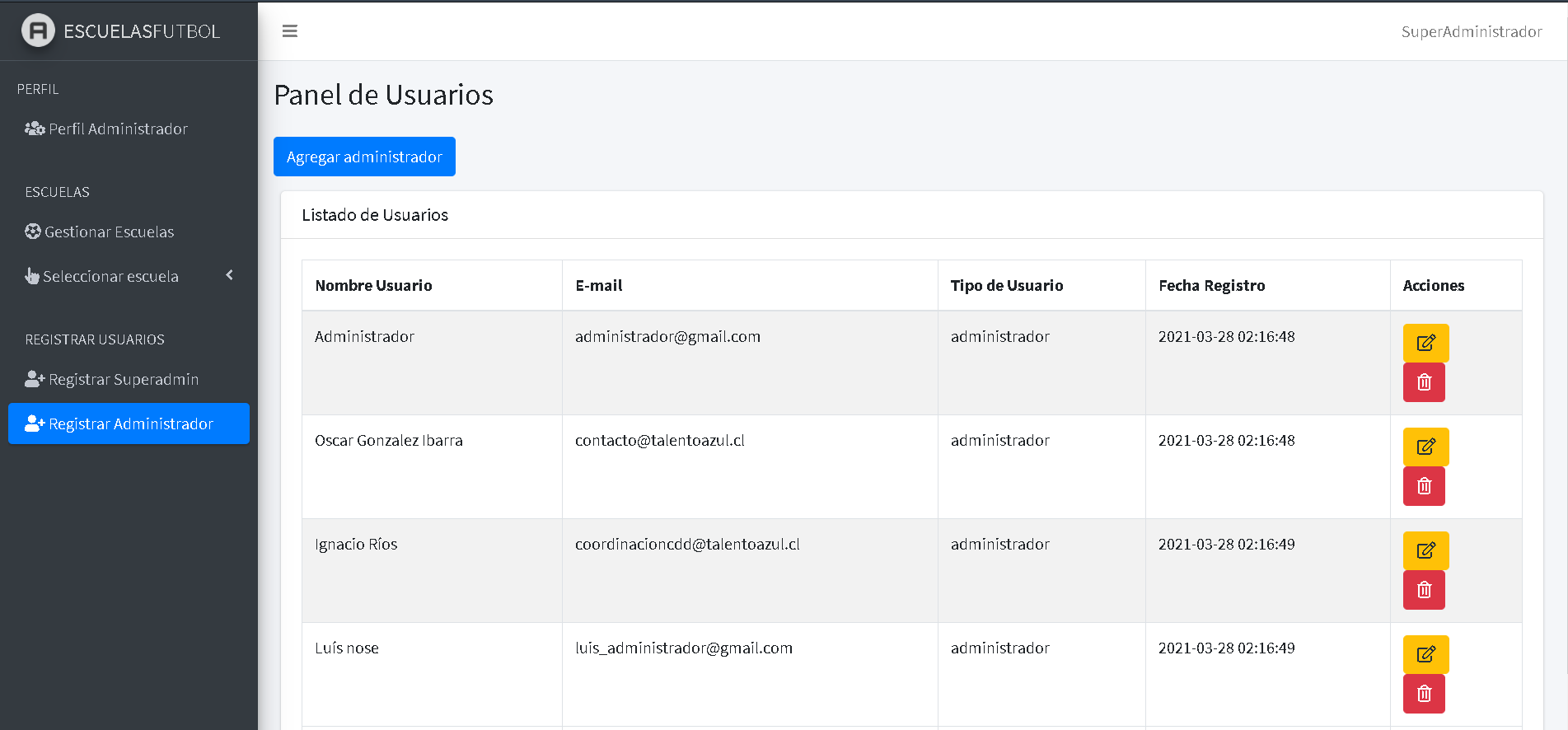


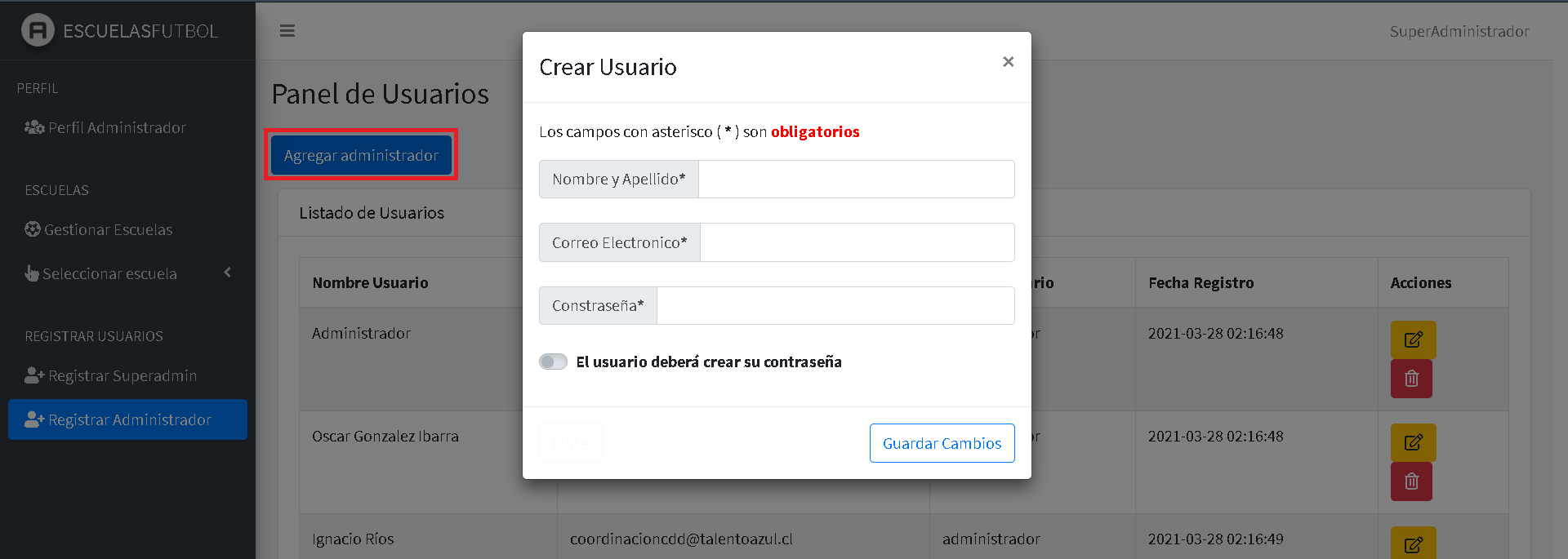


### SuperAdministrador/Superadmin



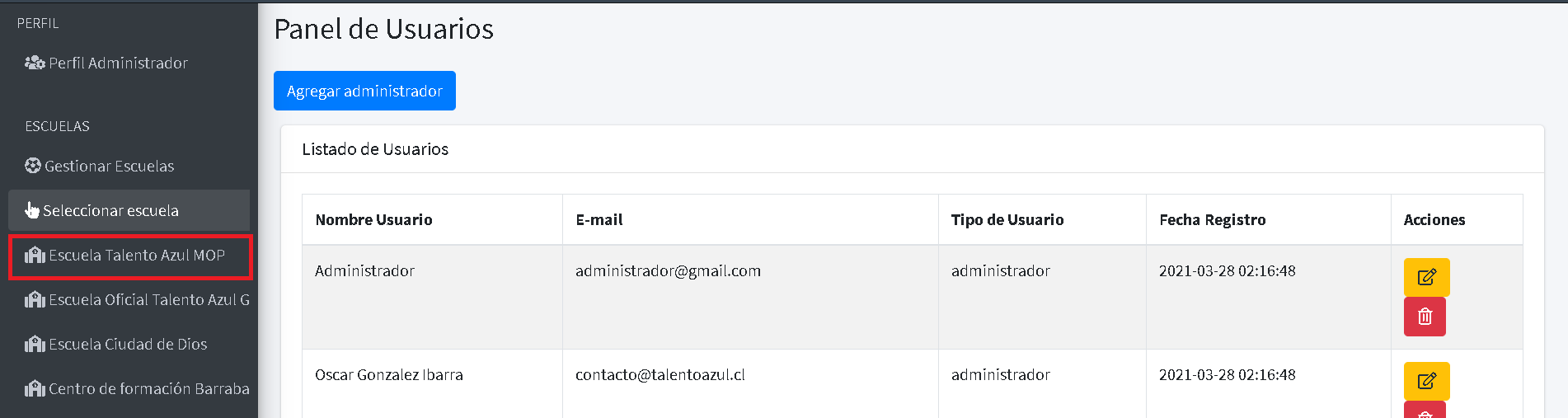
### SuperAdministrador/Administrador



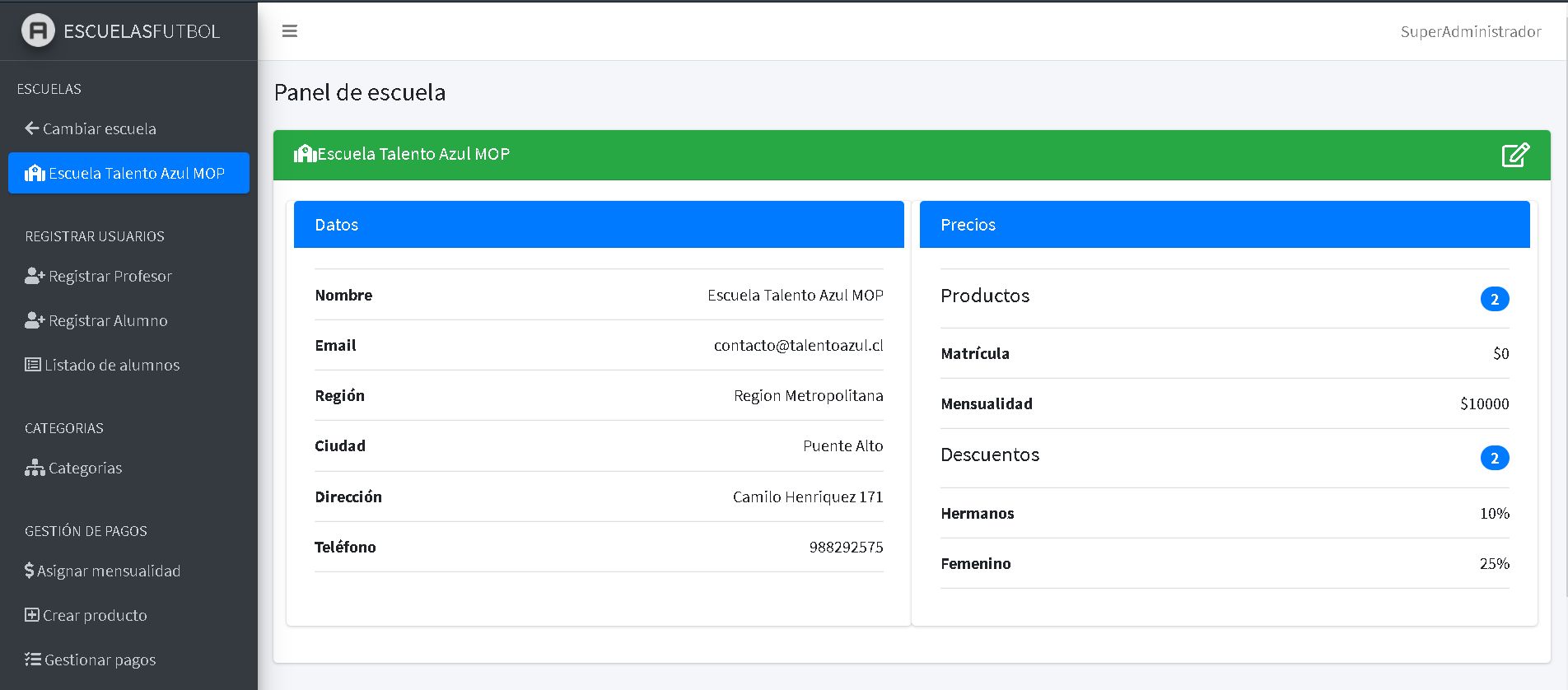




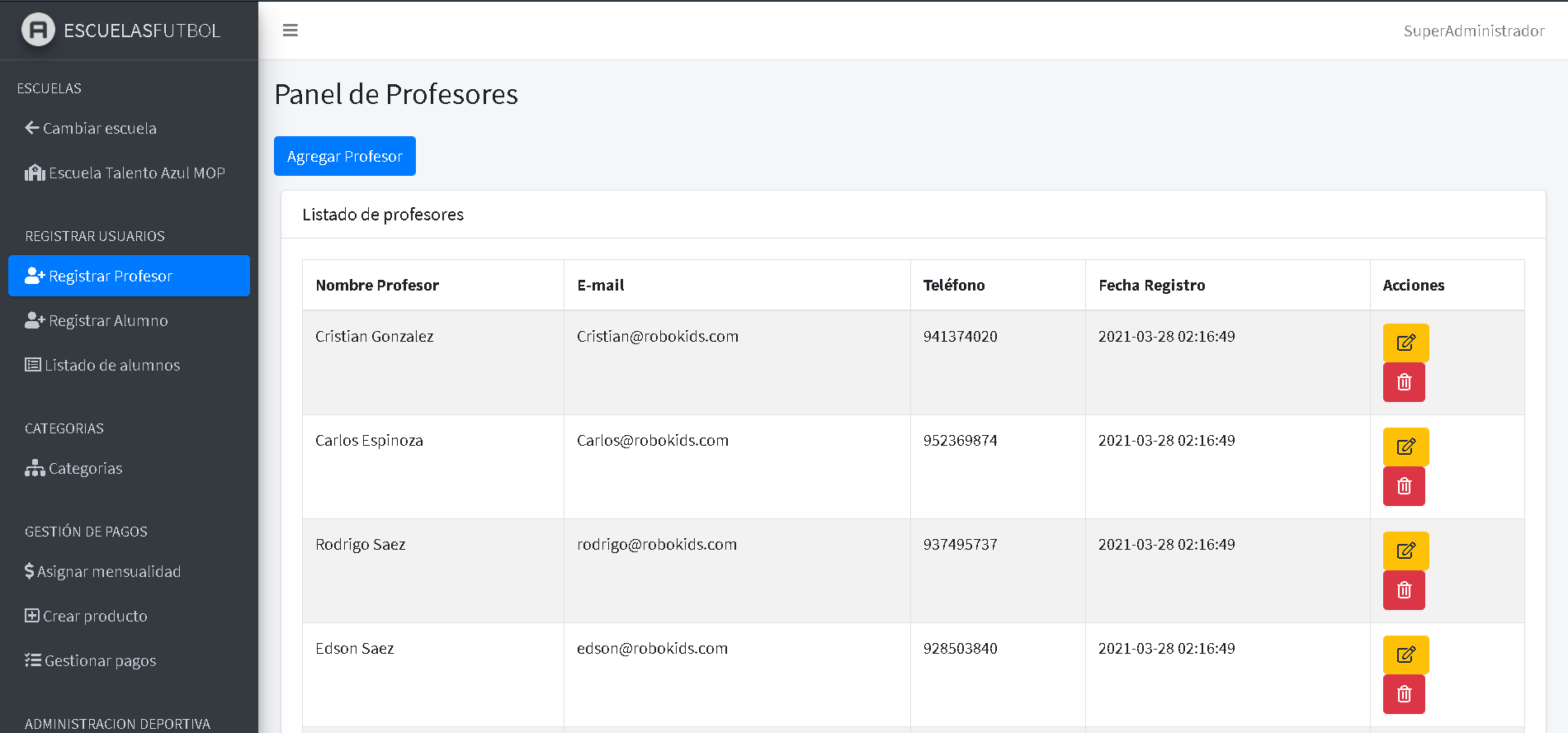
### SuperAdministrador/Seleccionar escuela



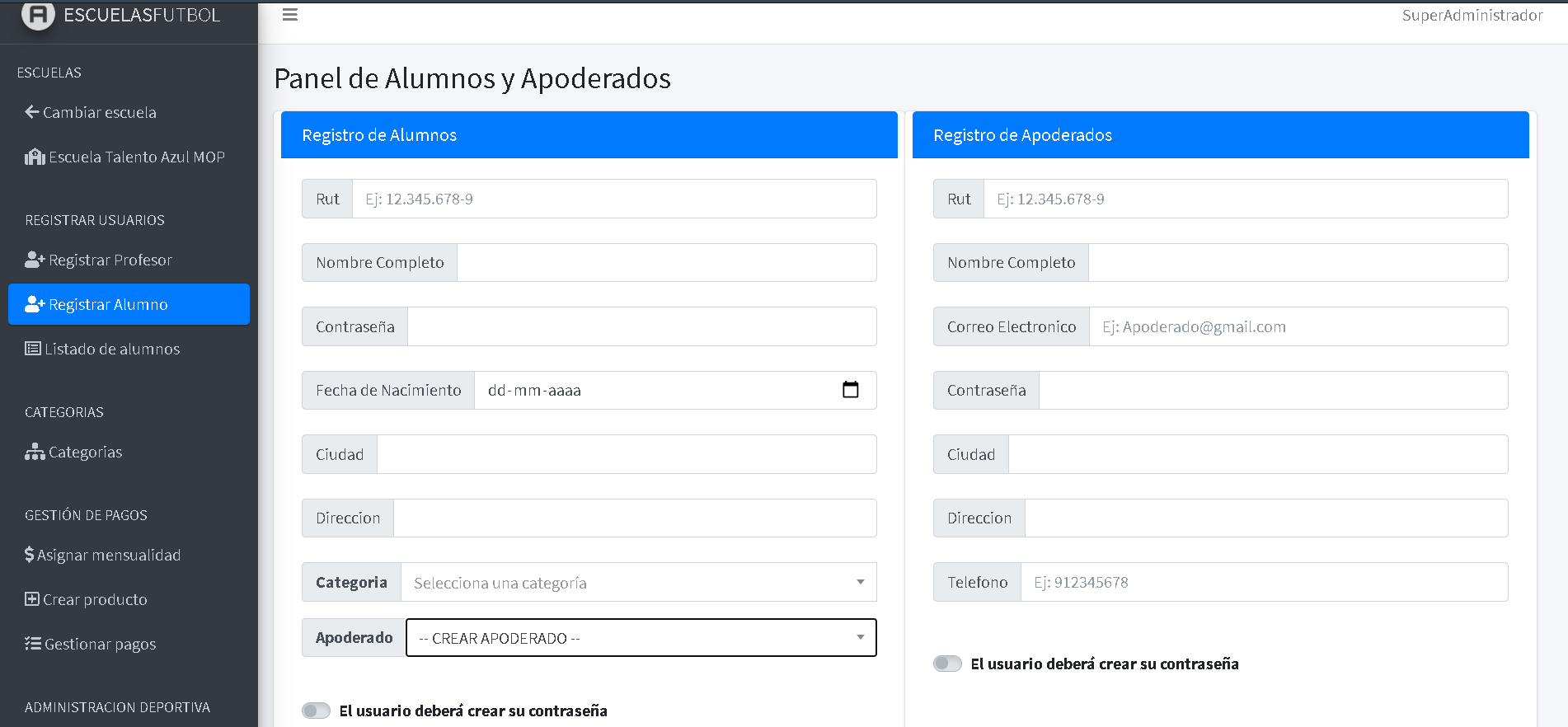
### SuperAdministrador/Ver escuela



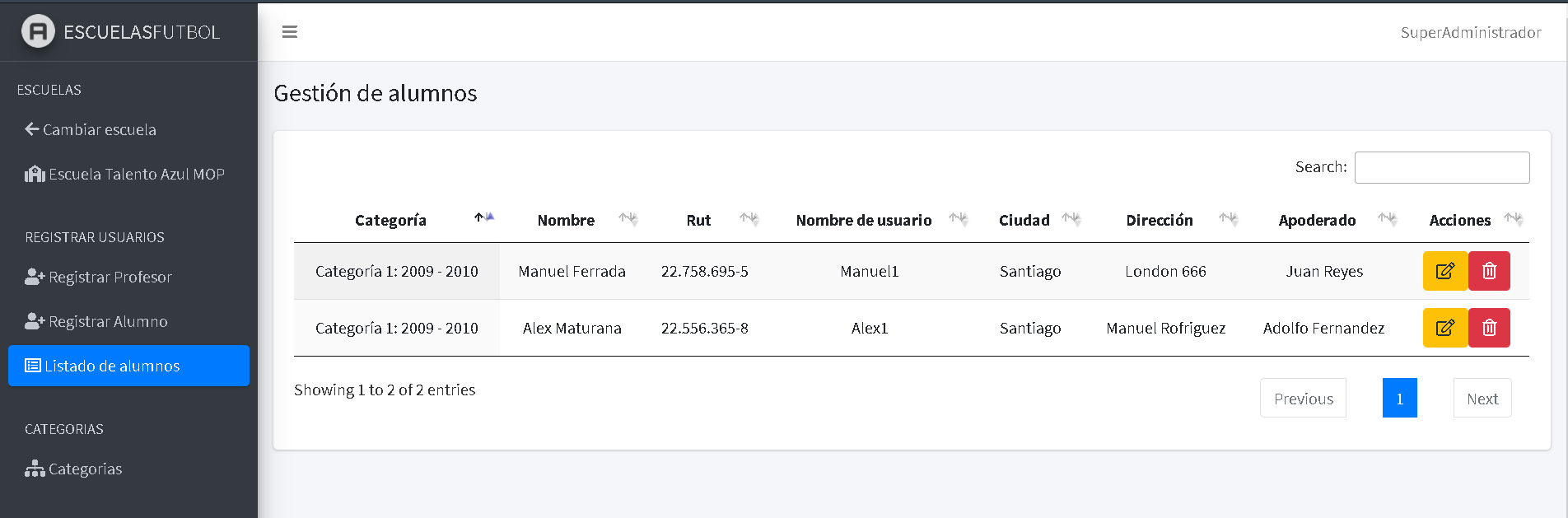
### SuperAdministrador/Registrar profesor



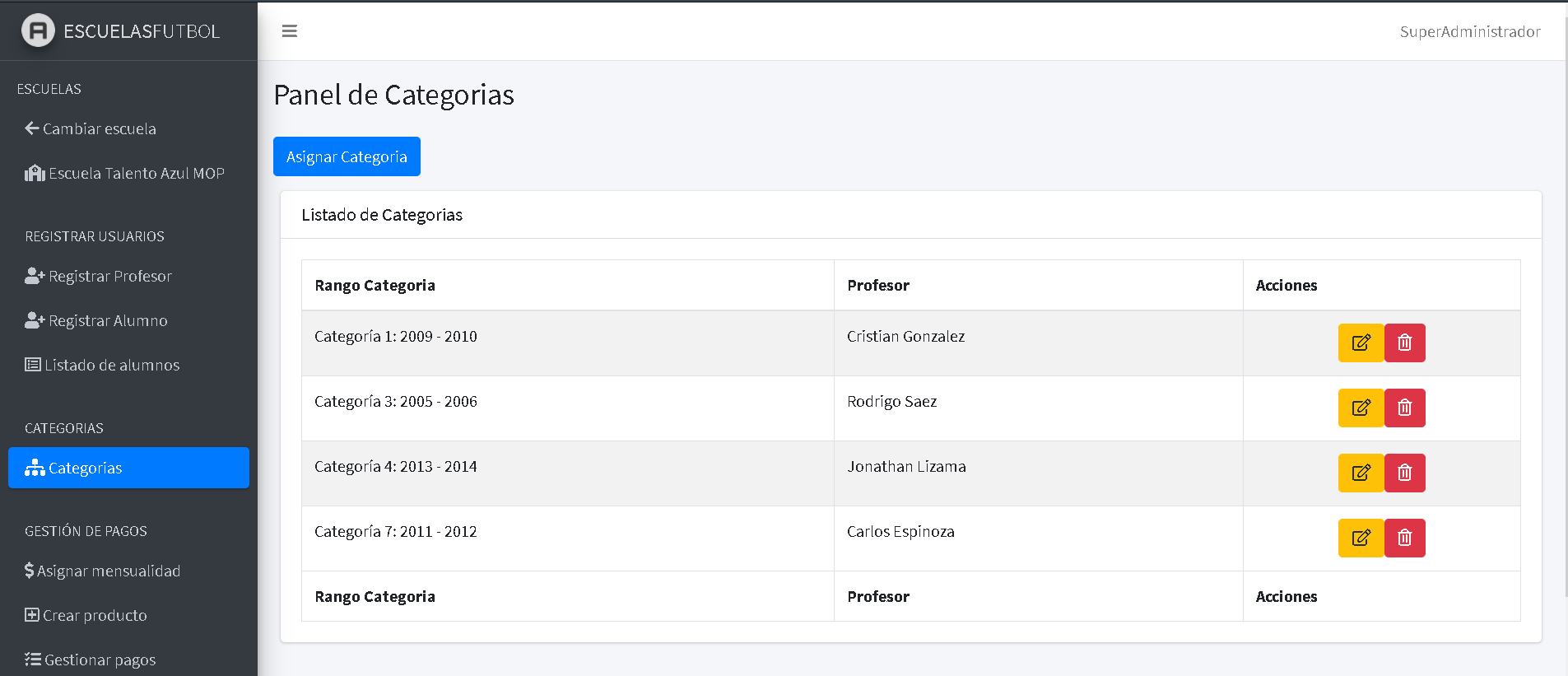
### SuperAdministrador/Registrar alumno



### SuperAdministrador/Listado de alumnos



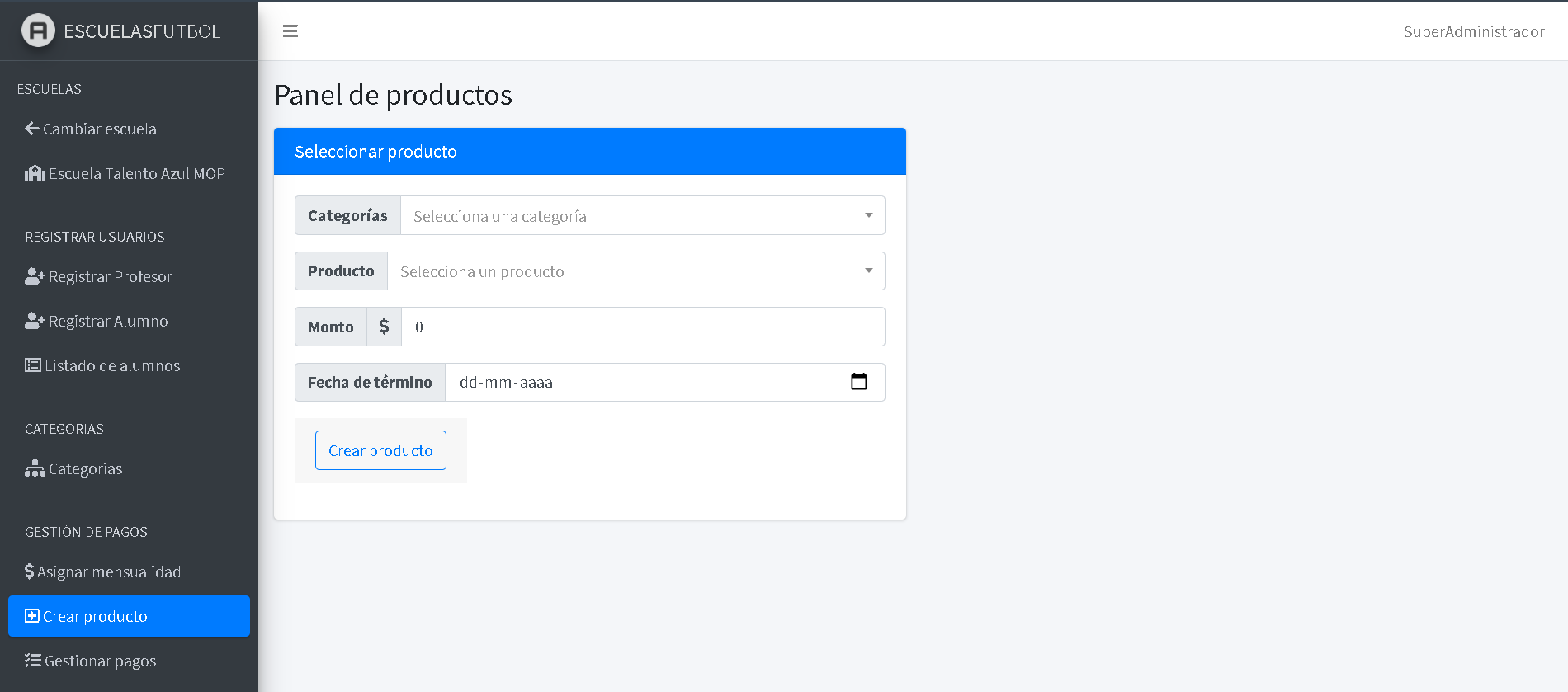
### SuperAdministrador/Categorías



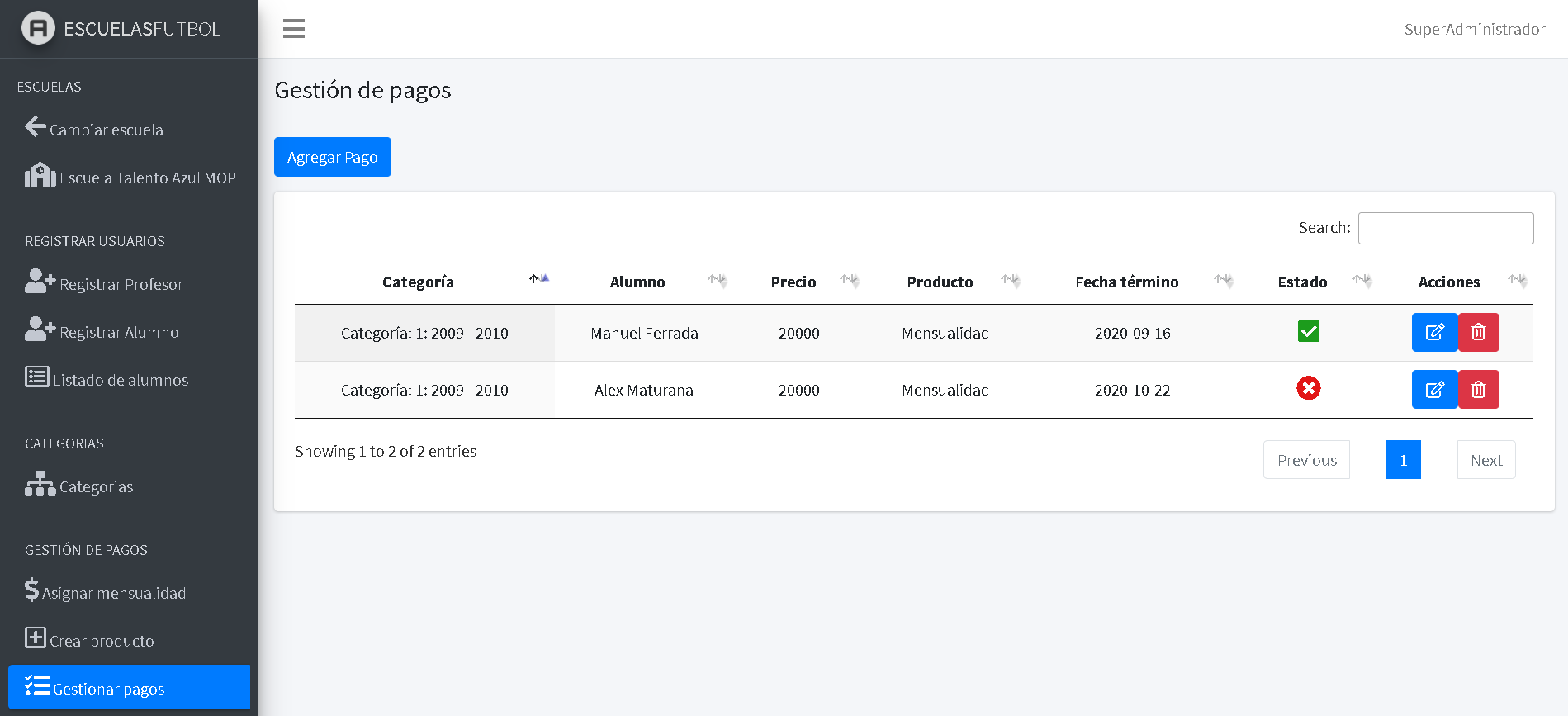
### SuperAdministrador/Asignar mensualidad



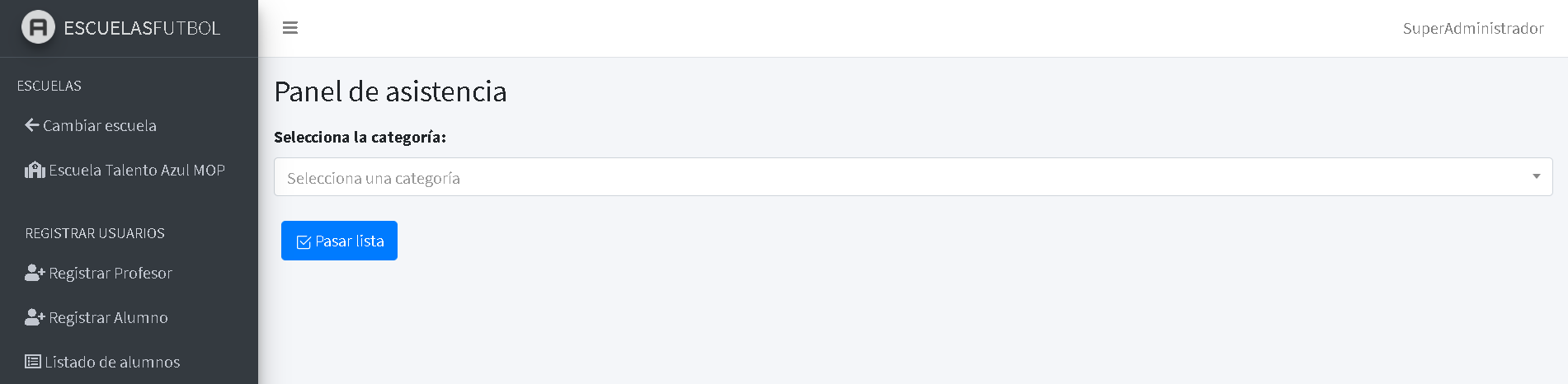
### SuperAdministrador/Crear producto

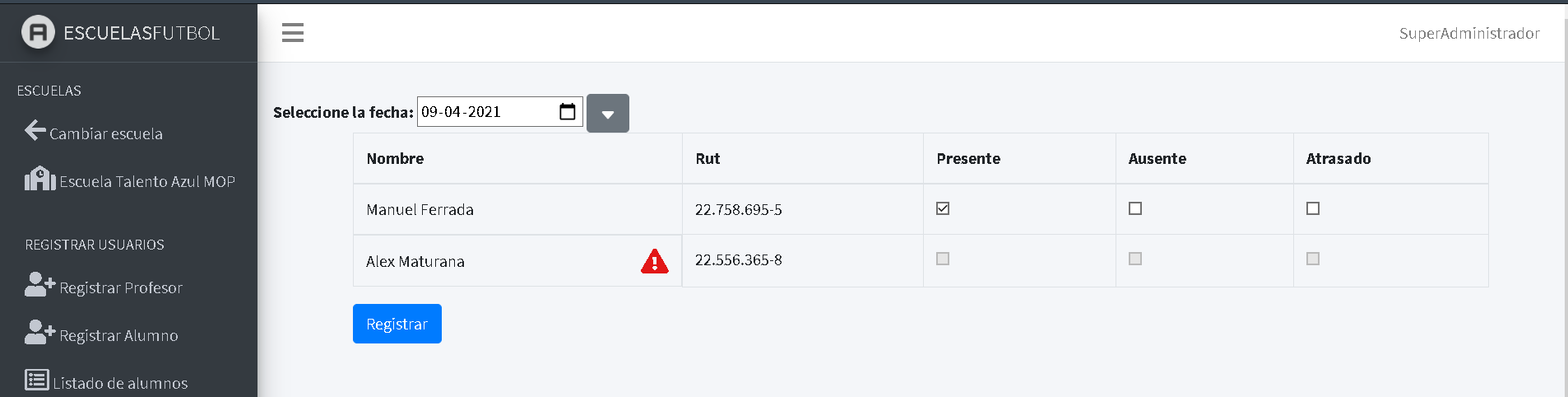


### SuperAdministrador/Gestionar pagos

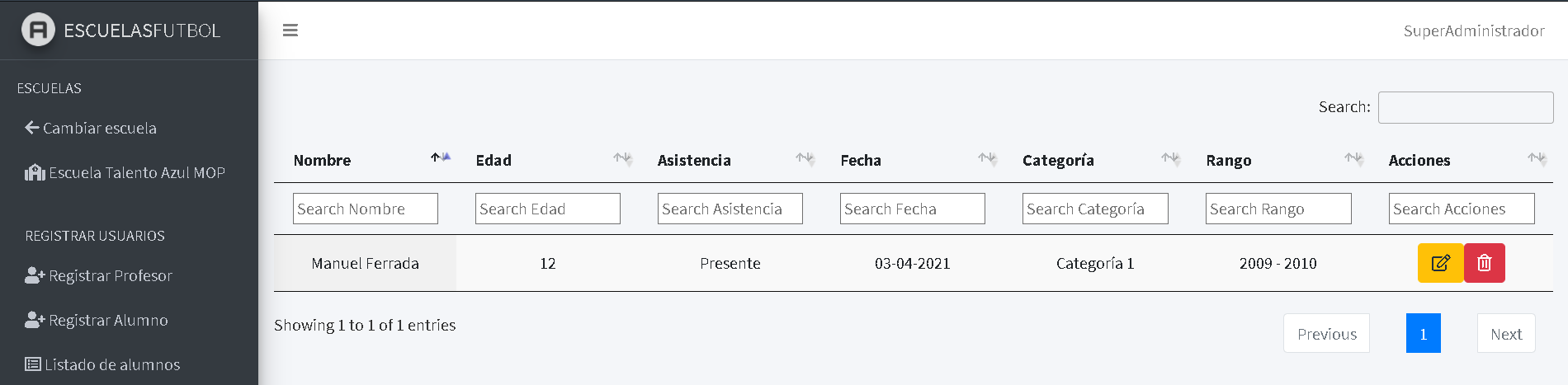


### SuperAdministrador/Tomar asistencia

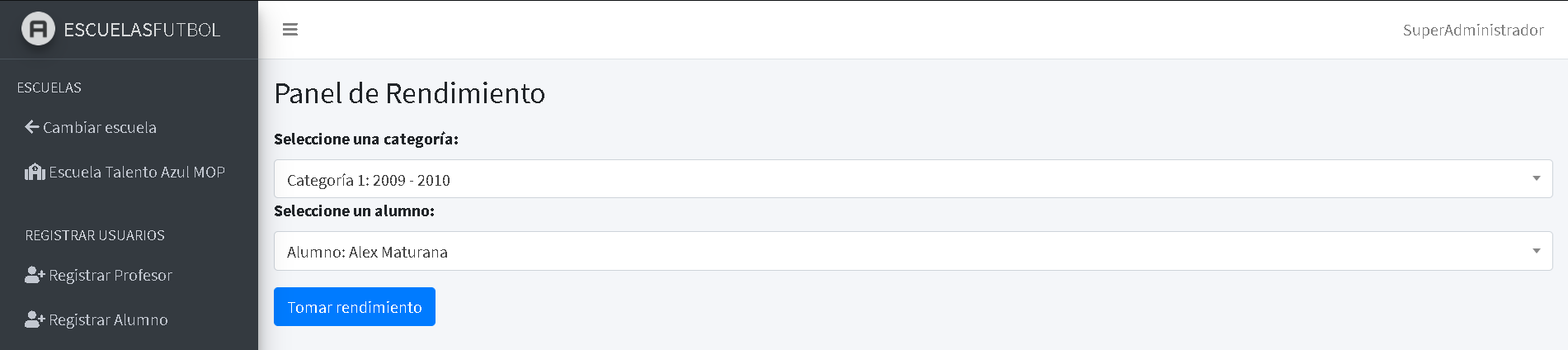


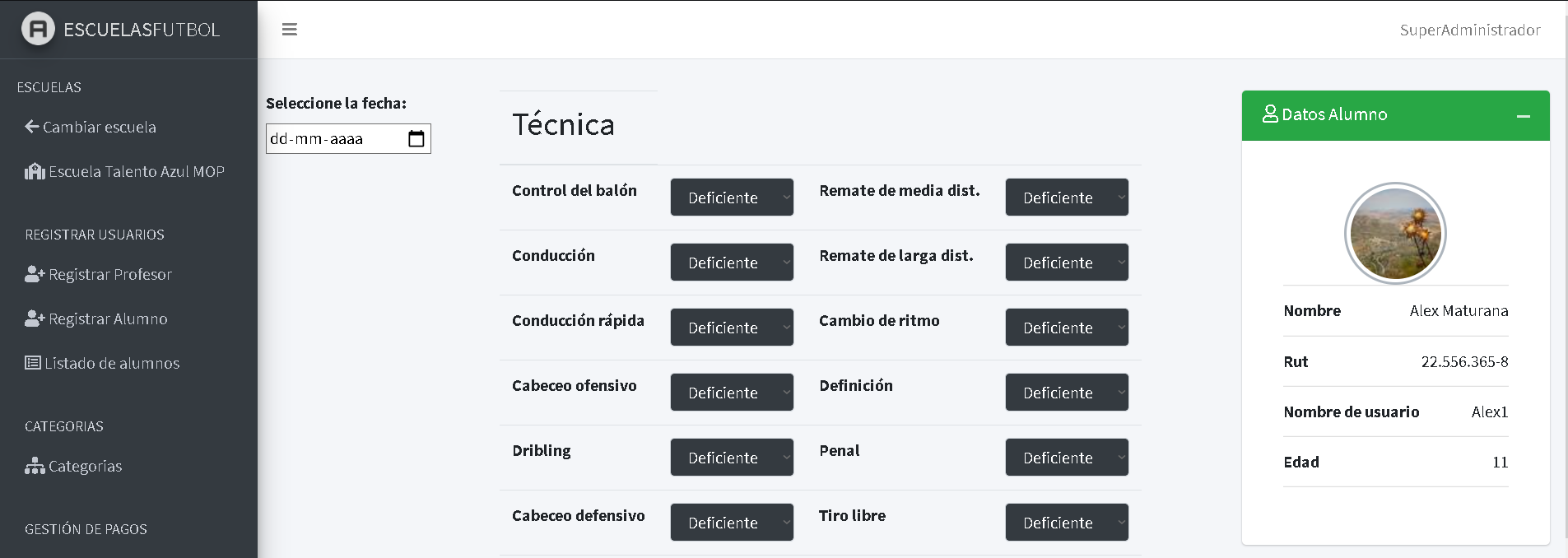


### SuperAdministrador/Ver asistencia

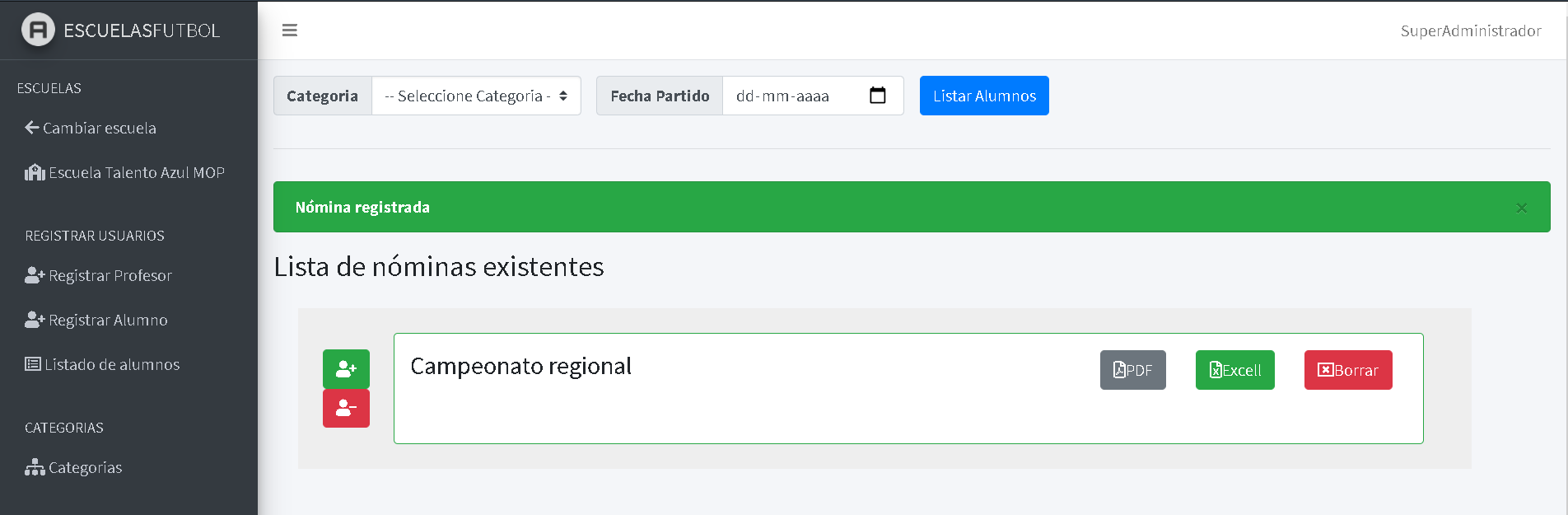


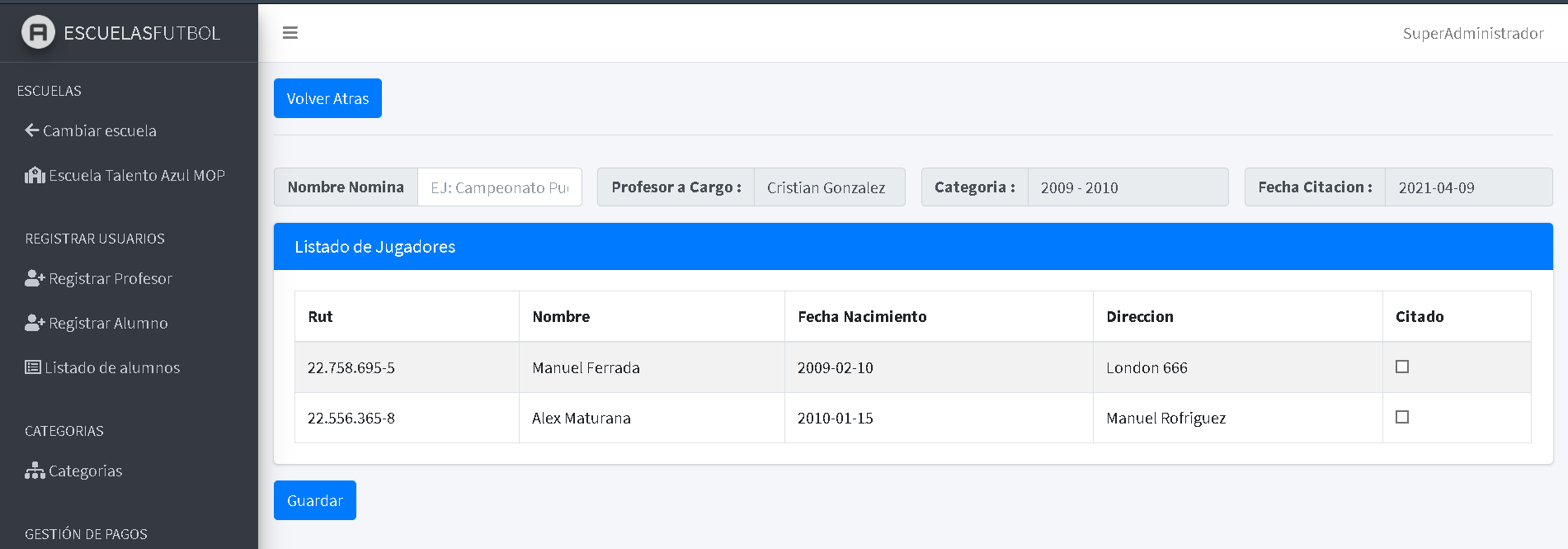
### SuperAdministrador/Tomar rendimiento



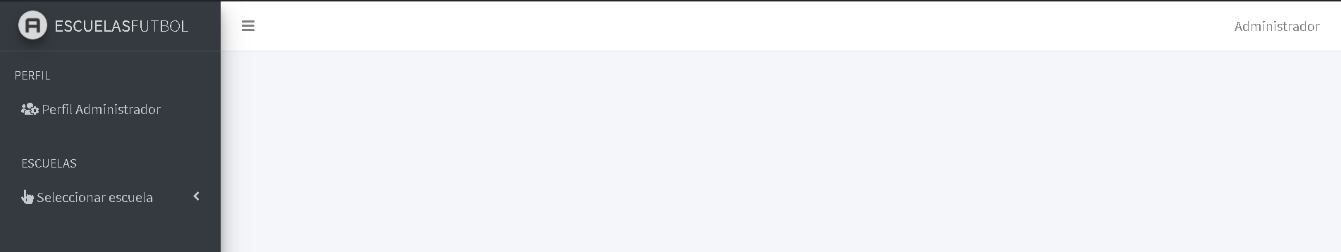


### SuperAdministrador/Nóminas de jugadores

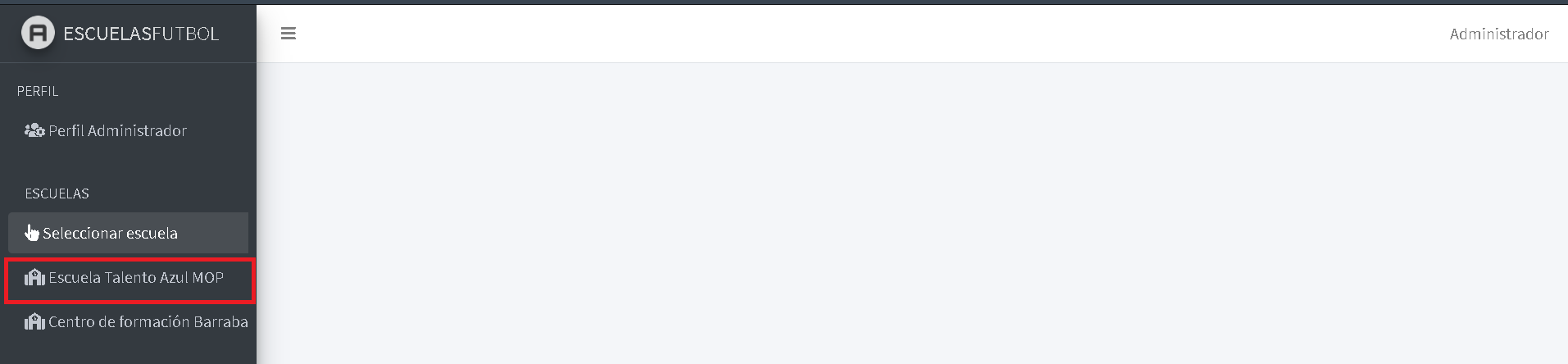




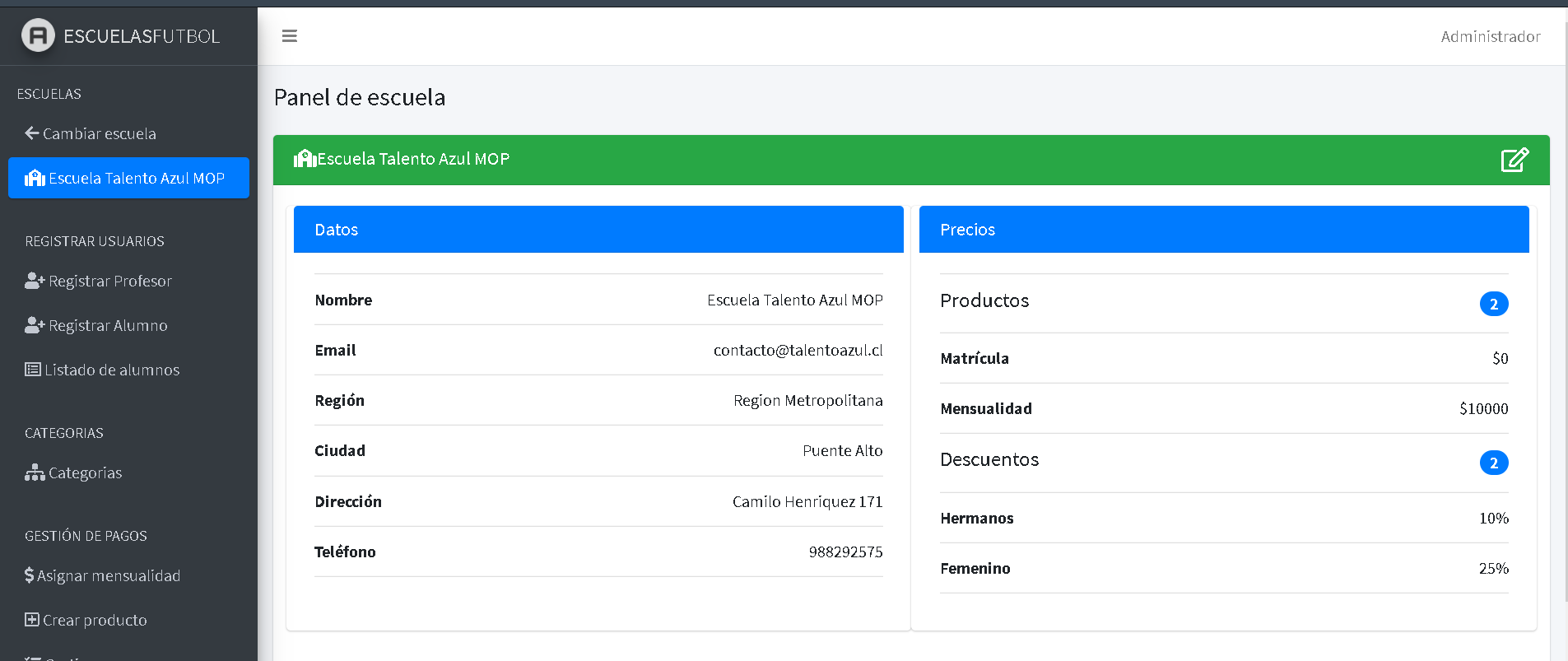
### Administrador/Perfil



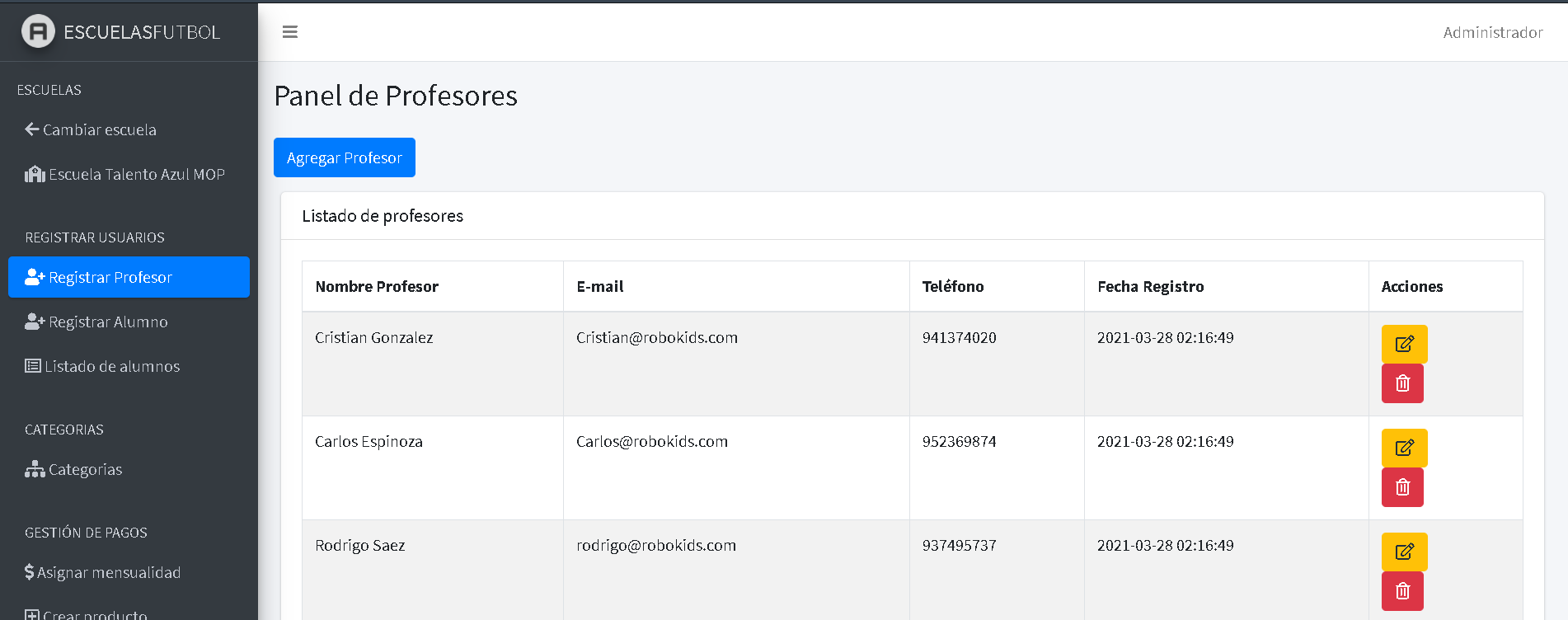
### Administrador/Seleccionar escuela



### Administrador/Ver escuela

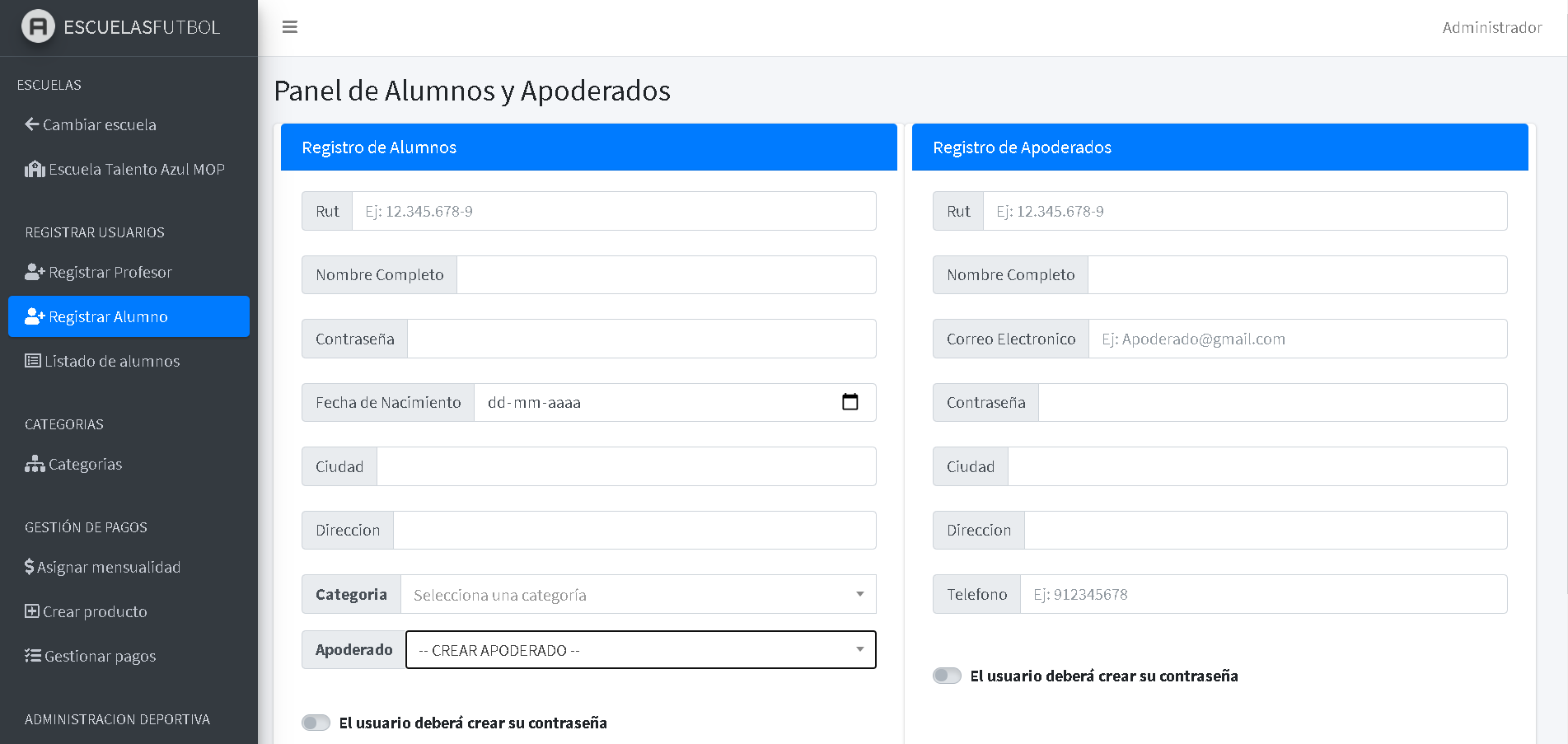


### Administrador/Registro Profesores

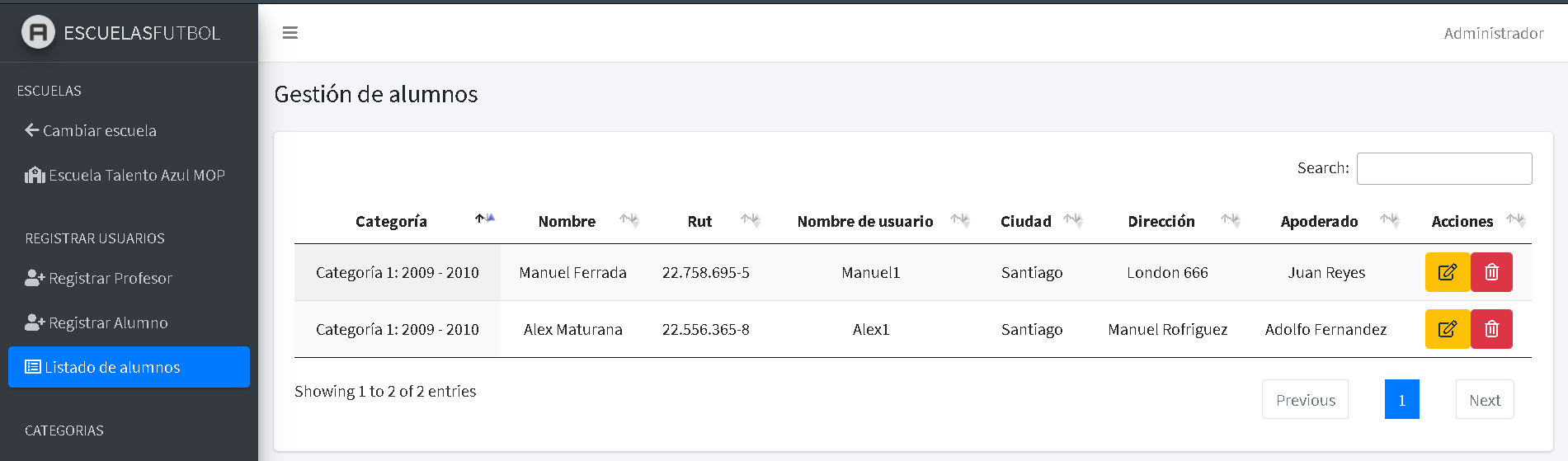




### Administrador/Registro Alumnos-Apoderados



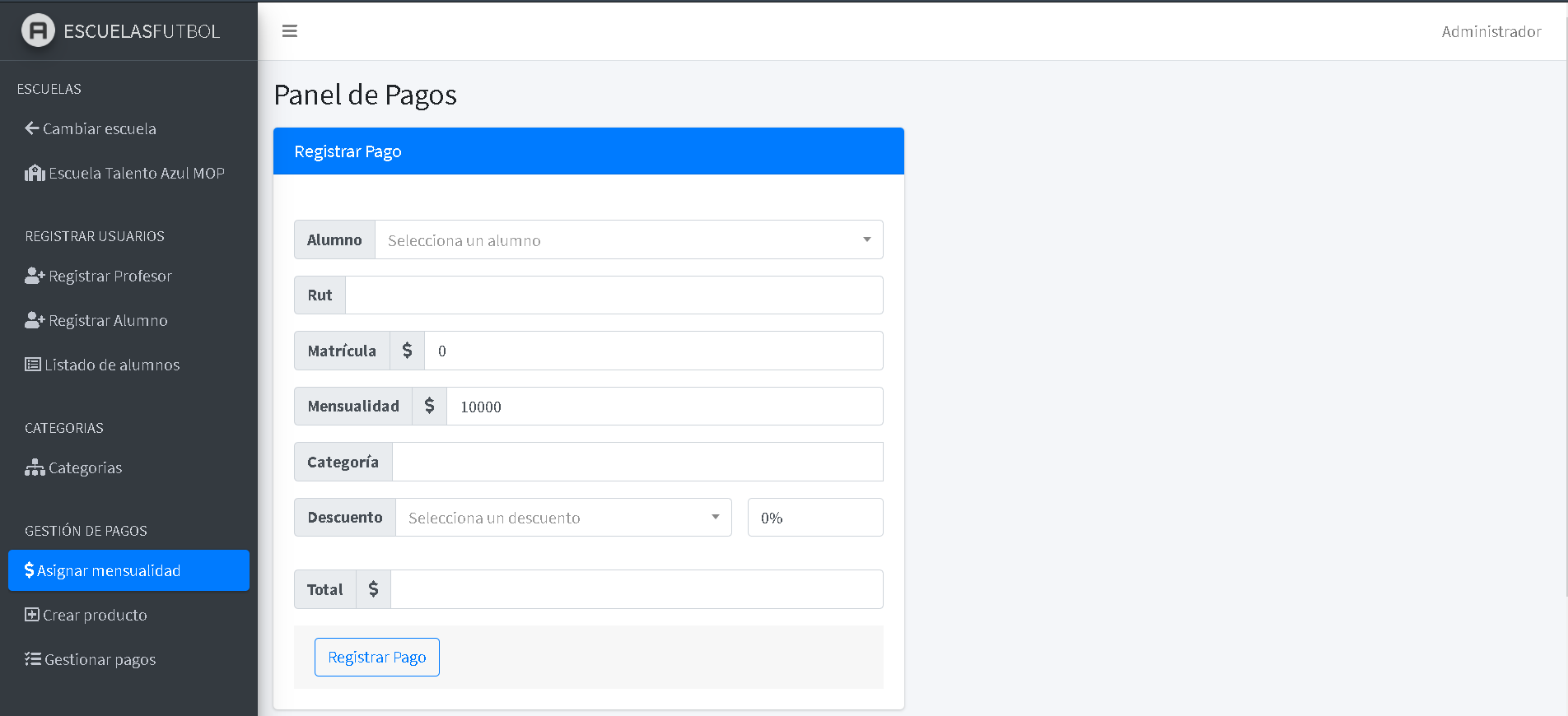
### Administrador/Ver Alumnos



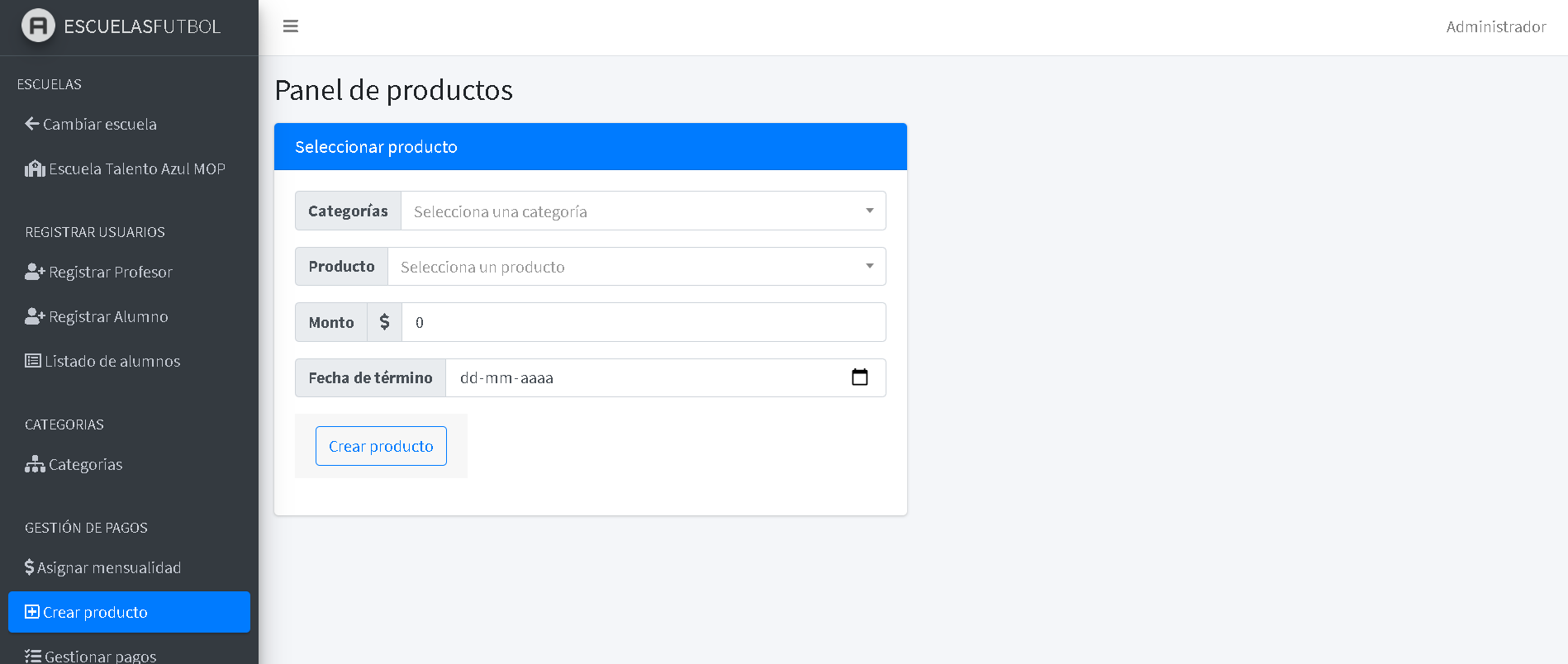
### Administrador/Categorías



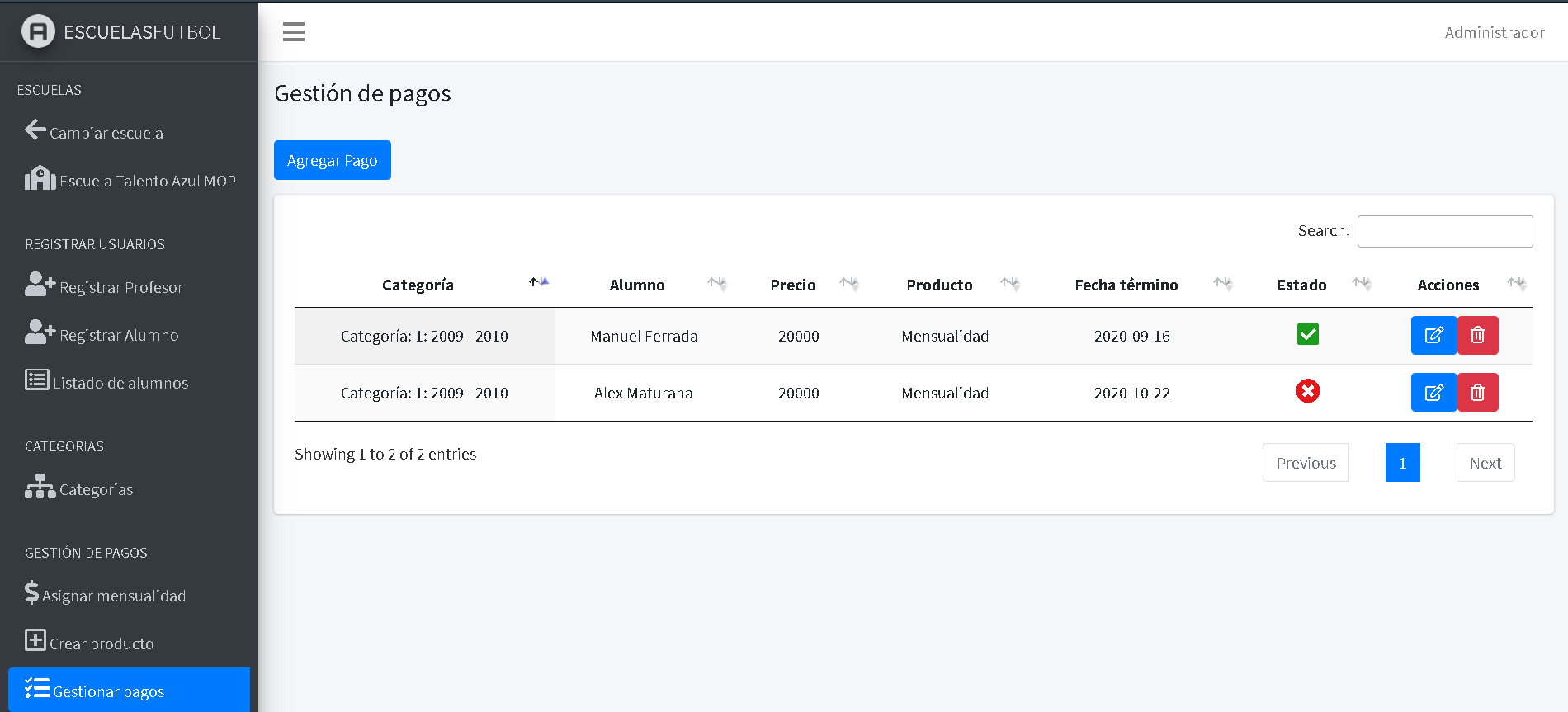
### Administrador/Mensualidad



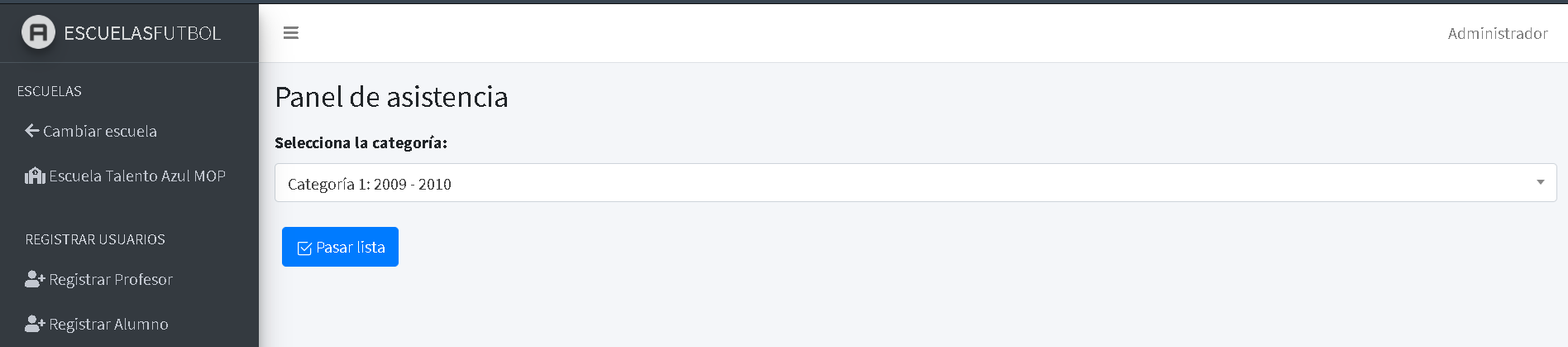
### Administrador/Producto

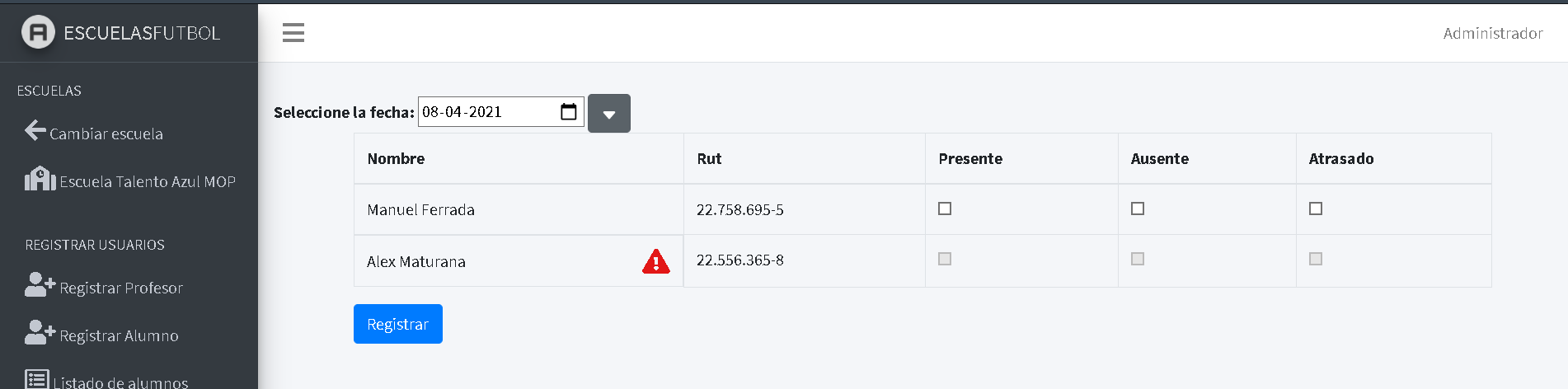


### Administrador/Pagos



### Administrador/Tomar asistencia

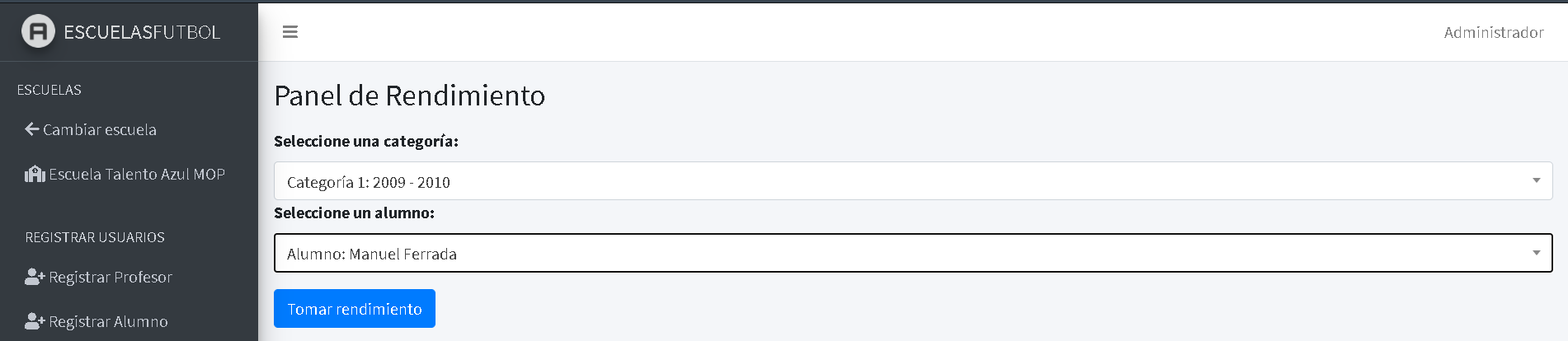


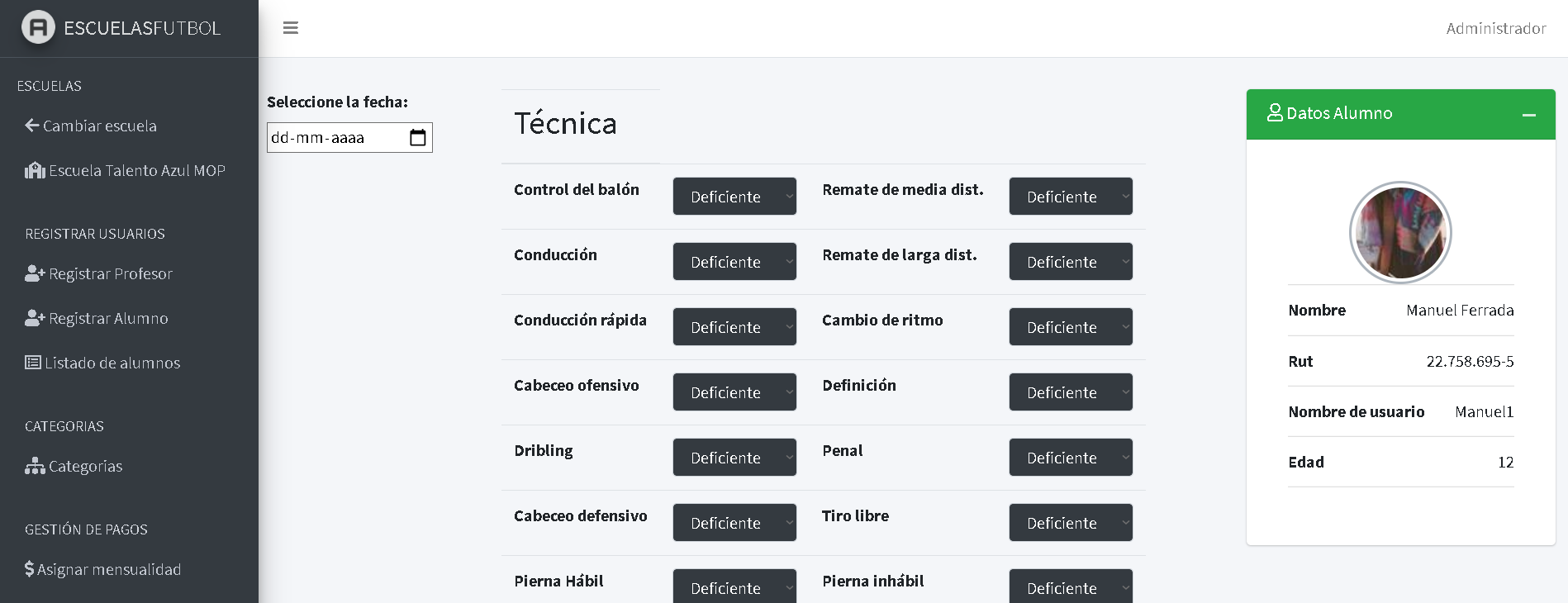


### Administrador/Ver asistencia



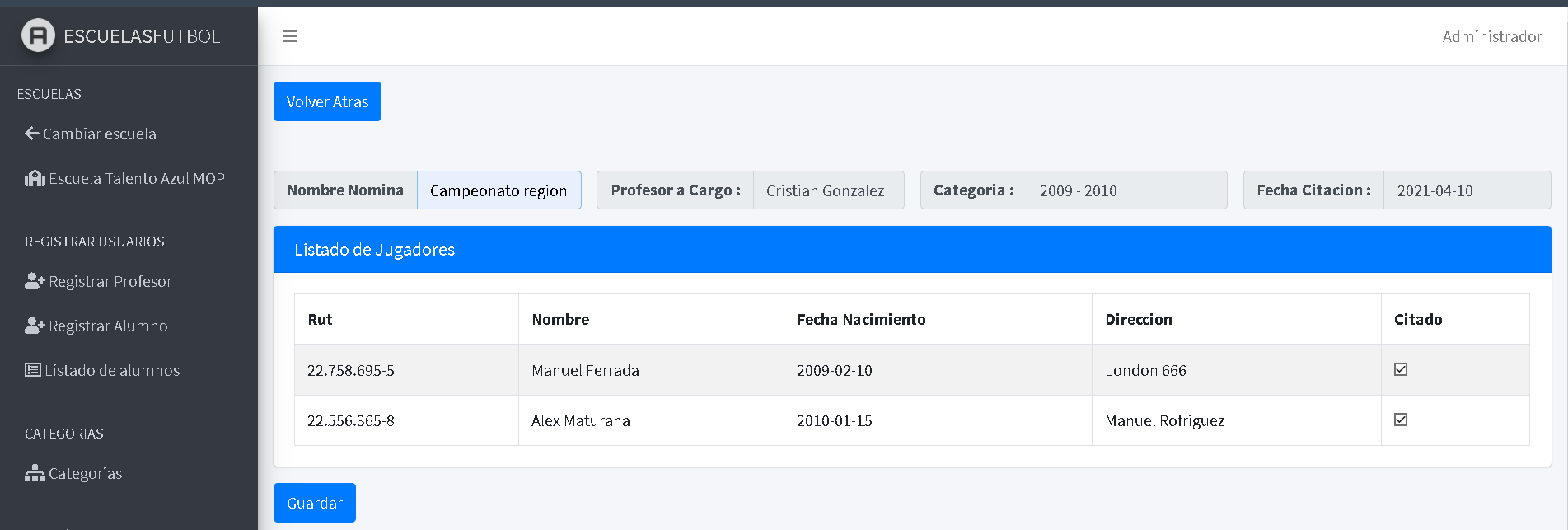
### Administrador/Rendimiento



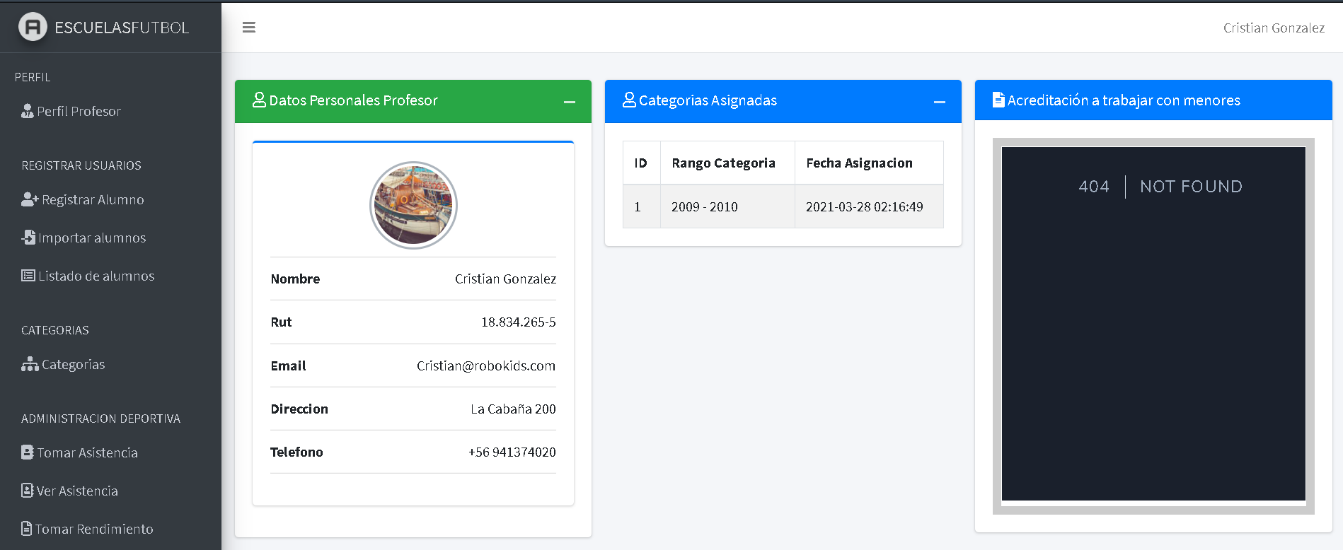


### Administrador/Nóminas Jugadores

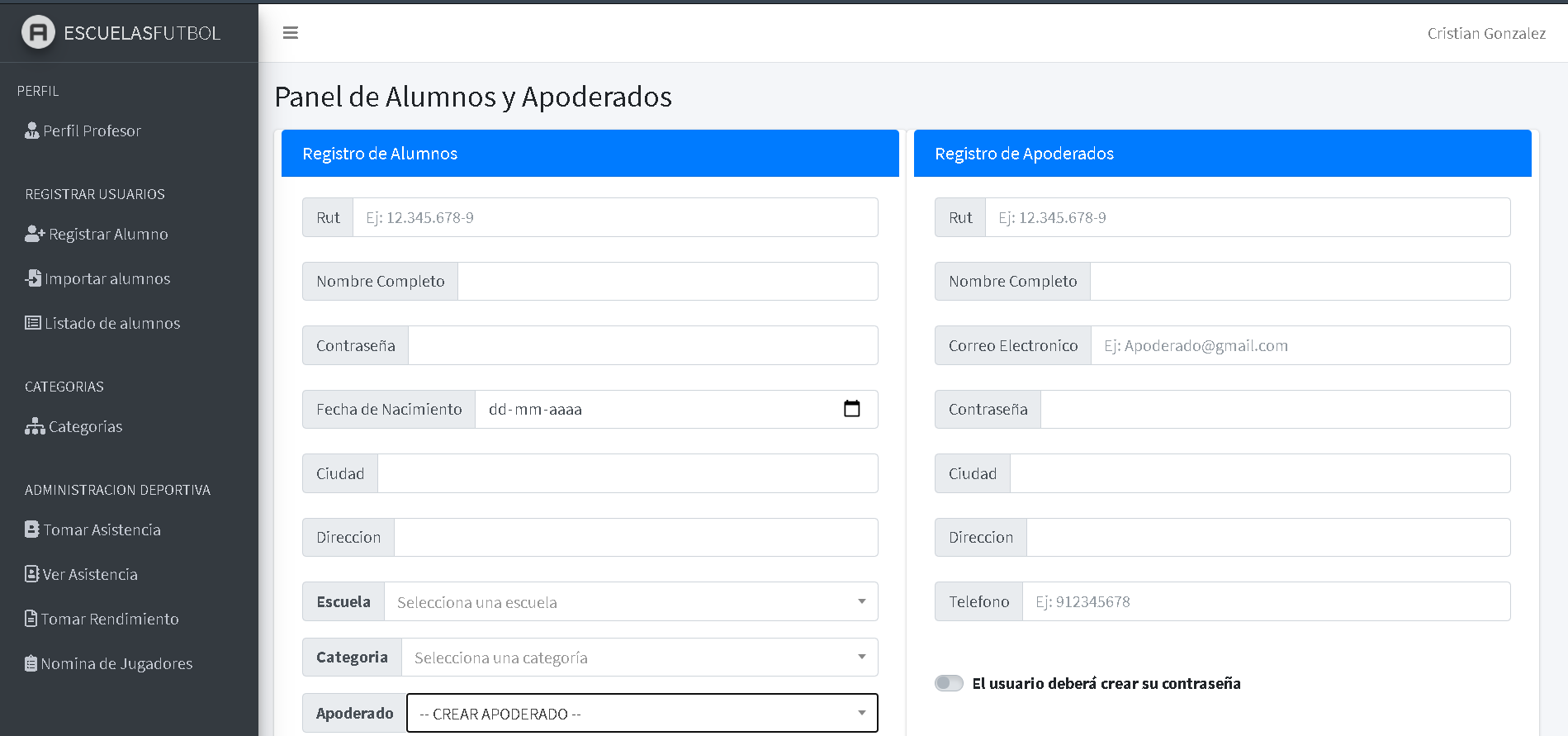




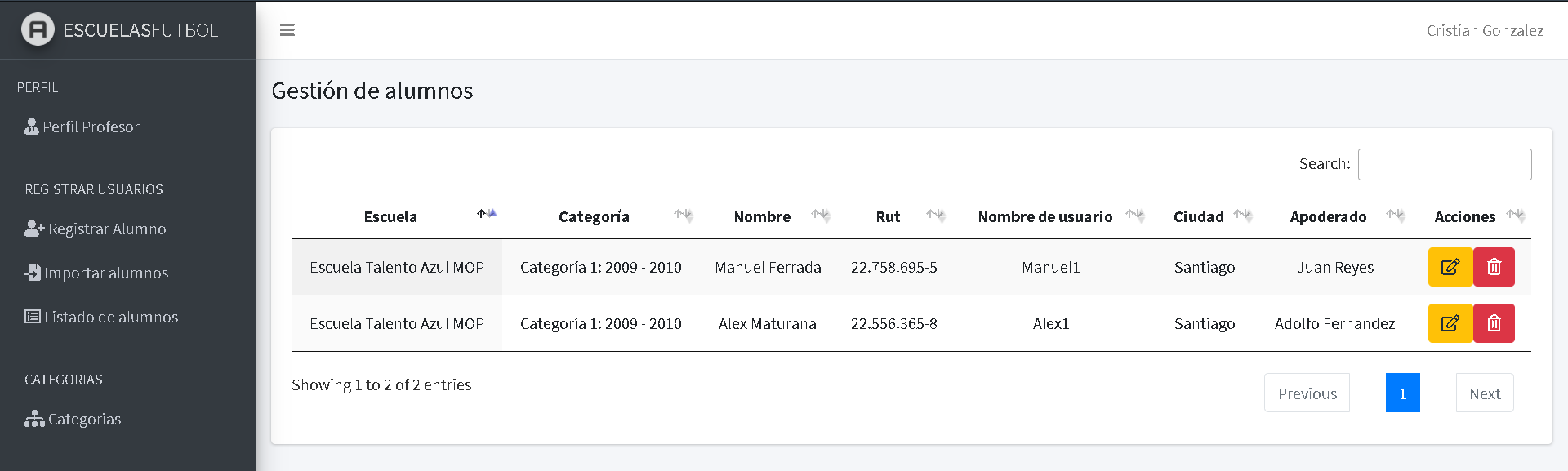
### Profesor/Perfil



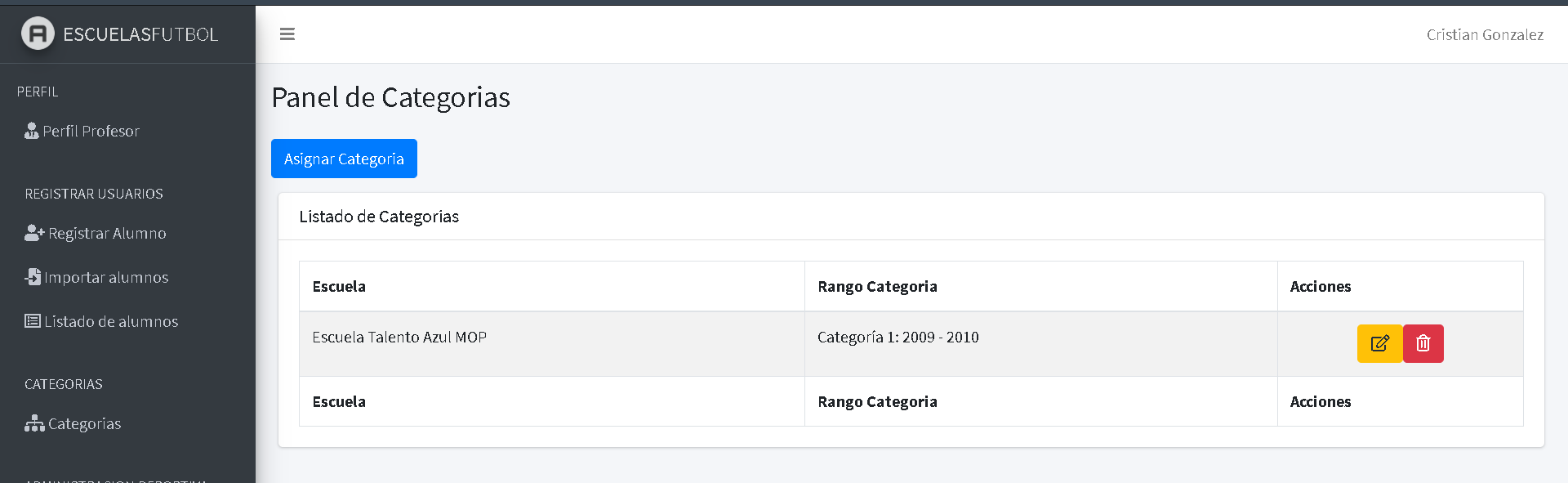
### Profesor/Alumno



### Profesor/Lista Alumno

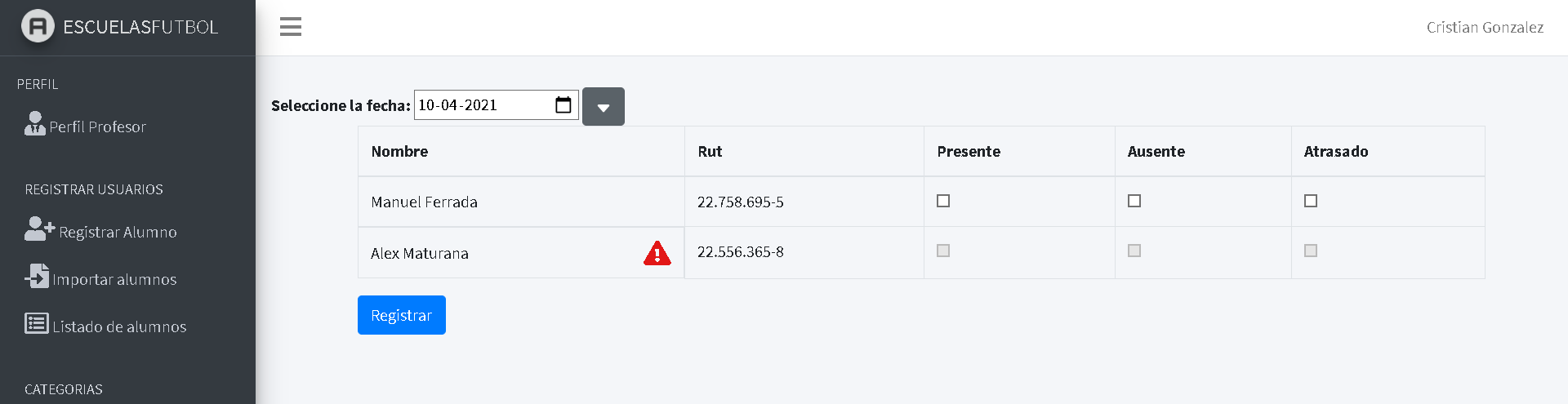


### Profesor/Categorías

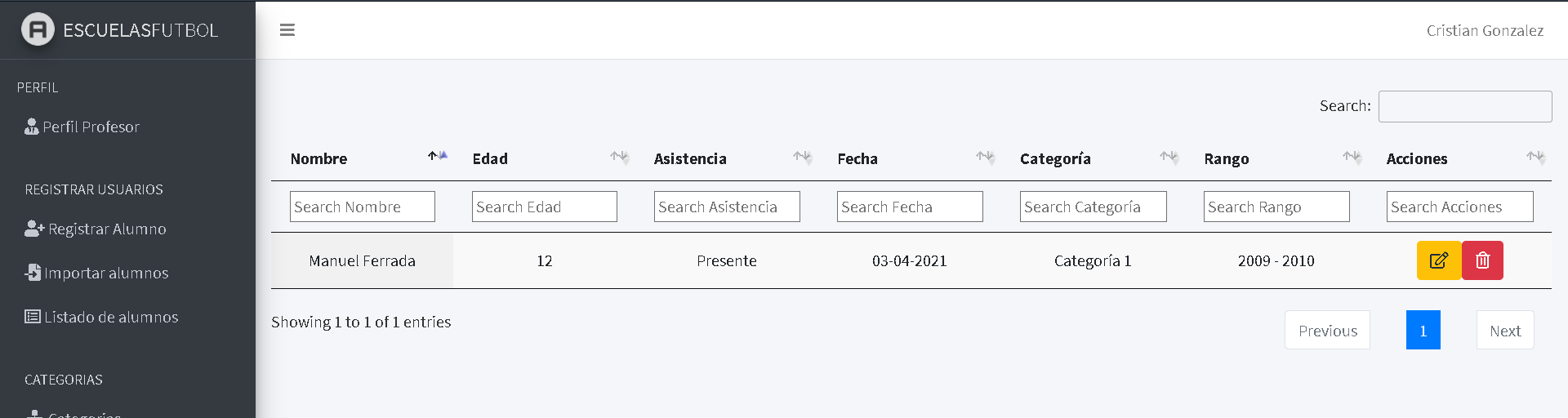


### Profesor/Tomar Asistencia

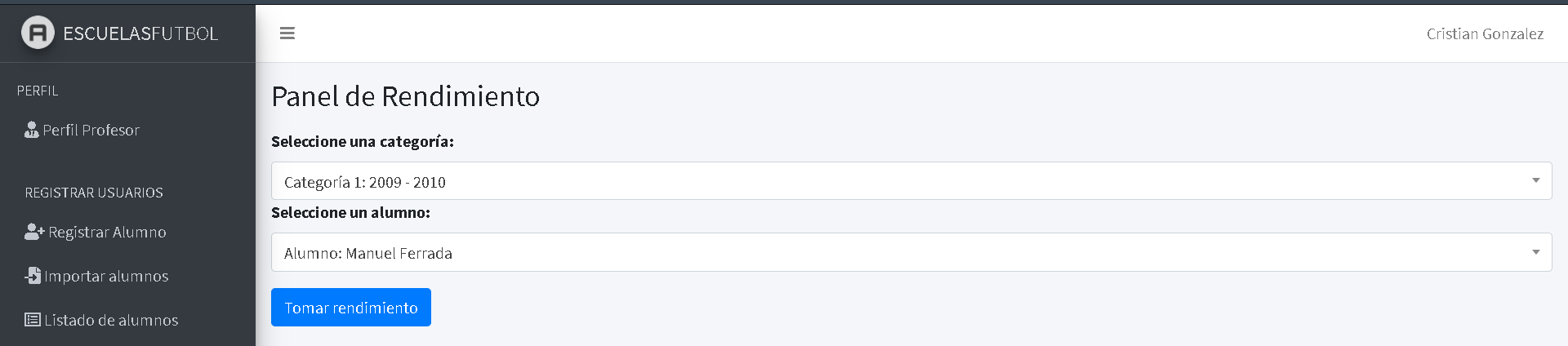
### 

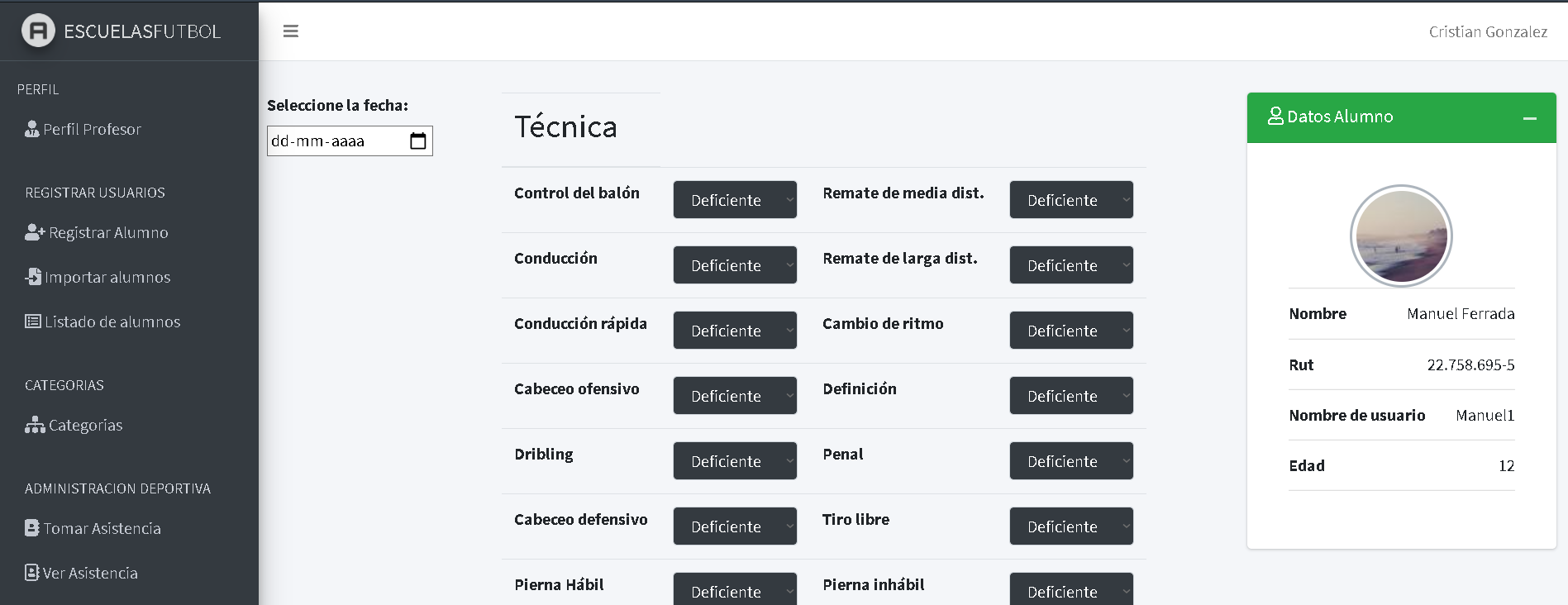


### Profesor/Ver Asistencia

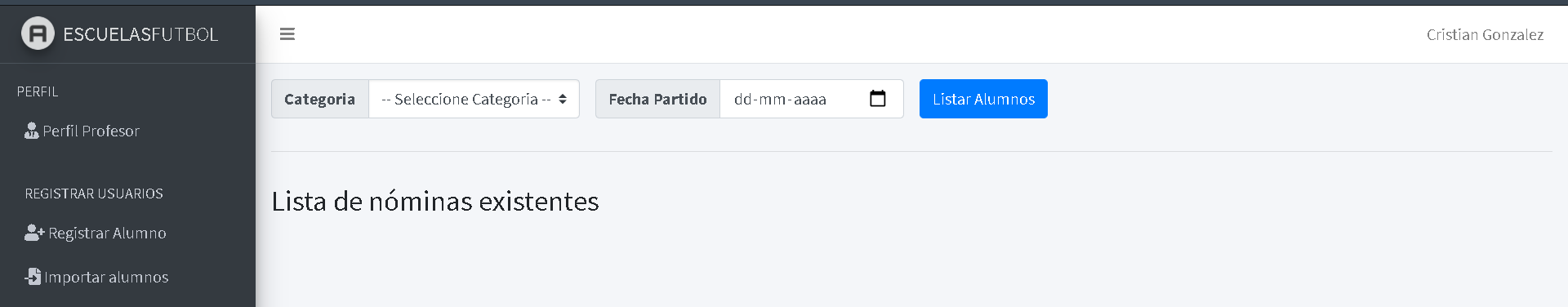


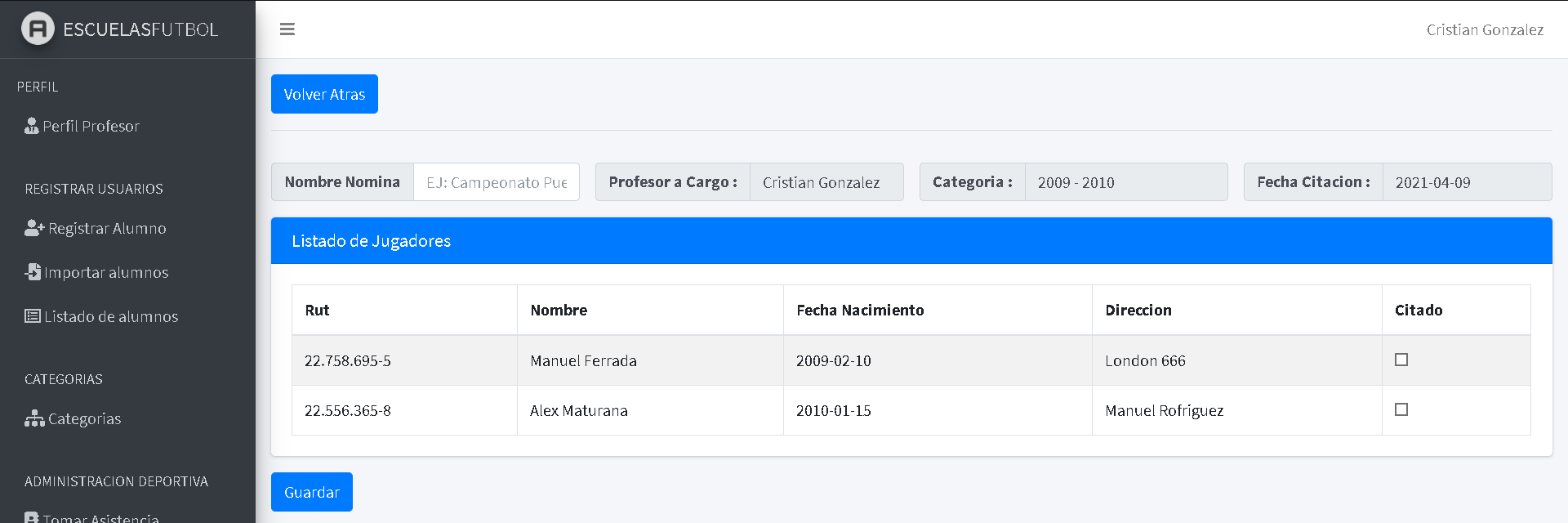
### Profesor/Tomar Rendimiento





### Profesor/Nomina Jugadores





### Alumno/Perfil

Interfaz de usuario gráfica, Aplicación, Tabla

Descripción generada automáticamente

### Apoderado/Perfil

