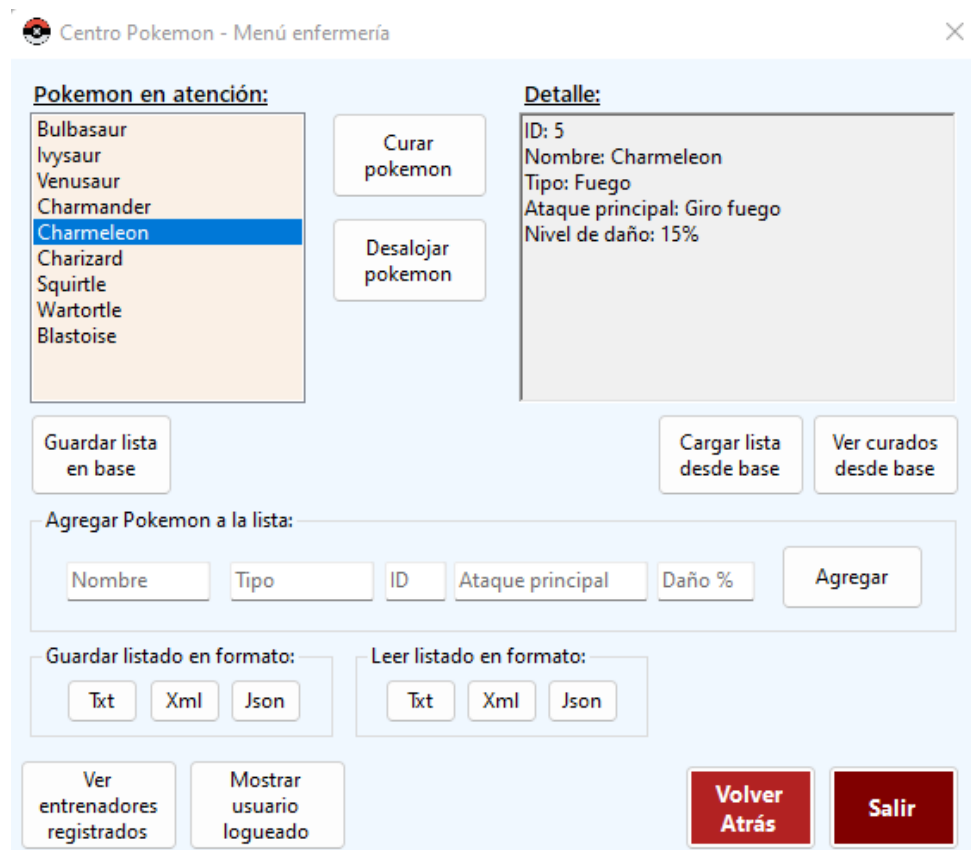


Instructivo – Centro Pokemon App

La app cuenta con un Login principal, donde ingresaremos como administrador o entrenador. Para acceder a todas las funcionalidades recomiendo ingresar como administrador, ya que podrás ingresar con el perfil de la Enfermera Joey, y así poder usar el sistema de atención y gestión que la app provee. Ingresando como entrenador, al momento actual solo podrás ver los pokemon alojados y sus detalles. El mismo consta de diferentes tipos de validaciones para asegurar un ingreso seguro y exitoso.



Una vez avanzado el Login, nos encontraremos con un menú principal, el cual diferirá en base al tipo de usuario el cual haya ingresado. El menú de enfermería:



En este menú, podremos ver la lista de pokemon alojados en el centro, y si presionamos click en algunos de ellos, podremos observar su detalle en la lista de la derecha.

Podremos agregar pokemon de manera manual a la lista, completando correctamente todos los campos necesarios para ello y luego presionando [Agregar].

A su vez podemos cargar pokemon desde una base de datos a través del botón [Cargar lista desde base], para realizar esto, primero hay que presionar [Guardar lista en base].

Tenemos el botón [Curar pokemon], que actualiza el daño del pokemon seleccionado a 0 (si su daño es mayor a 0) en la lista y base si se encuentra, y el [Desalojar pokemon] que elimina el pokemon de la lista (si el mismo se encuentra con daño 0, o sea curado) y de la base si se encuentra en la misma.

En caso de querer preservar la información, podemos a través de los botones del group “guardar listado en formato”, guardar un archivo txt, xml, o json, los mismos se guardarán en una carpeta en el escritorio. A su vez podemos leer esos archivos y visualizarlos en la lista de la derecha a través de los botones del group “Leer listado en formato”. Deberías haber guardado los archivos antes de intentar leerlos.

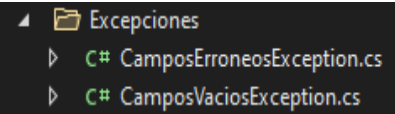
Por último tenemos el botón [Ver curados desde base] que realiza un filtro y nos trae los pokemon con daño 0 cargados en la base.

Después tenemos funciones accesorias como “Ver entrenadores registrados”, que visualizará qué usuarios están registrados en el Centro. Y tenemos el botón “Mostrar usuario logueado”, que mostrará qué usuario está usando la app.

Y como toda app necesita, un botón para ir una ventana atrás, o para salir completamente de la app.

Temas aplicados:

Excepciones: Se crearon dos excepciones, se usan en el Login para verificar el correcto ingreso de datos. También se aplican en diferentes partes del código para que el mismo no rompa.



```
Excepciones
├── C# CamposErroneosException.cs
└── C# CamposVaciosException.cs
```

Métodos de extensión: Extendemos un string, el cual en este caso será el nombre de un pokemon seleccionado, aplicará el método curar y hará un update del daño en la base.

```
public static string CurarPokemon(this string strpokemon)
{
    foreach (Pokemon pokemon in Pokemon.ListaPokemon)
    {
        if (strpokemon == pokemon.nombre)
        {
            pokemon.danio = 0;
            PokemonAccesoDatos.UpdateDanioCurar(strpokemon);
            return $"{pokemon.nombre} ha sido curado y su daño es 0";
        }
    }
    return "No se encontró el pokemon";
}
```

```
private void btnDesalojarPokemon_Click(object sender, EventArgs e)
{
    if (this.lstPokemon.SelectedItem is not null)
    {
        string pokemonSeleccionado = (string)this.lstPokemon.SelectedItem;
        if (PokemonAccesoDatos.EliminarPokemonDeBase(pokemonSeleccionado) && desalojarDelCentro(pokemonSeleccionado))
        {
            MessageBox.Show($"Se desalojó a {pokemonSeleccionado}, se quitó de la lista y de la base si se encontraba en ella.", "Desalojo exitoso");
            MostrarPokemonEnListaPokemon();
        }
        else
        {
            MessageBox.Show("El pokemon aún no se encuentra curado.", "Elegi un pokemon sano", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
    }
    else
    {
        MessageBox.Show("No se seleccionó ningún pokemon.", "Elegi un pokemon a desalojar", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

Test unitarios: Se prueba el método de extensión nombrado antes, y un método que prueba la carga manual de un pokemon.

```
9 //Pruebo método Carga de datos manual(AgregarPokemonManual)
10 [TestMethod]
11 //0 referencias
12 public void ProbarCargaDeDatosManualNuevoPokemon()
13 {
14     //Act
15     bool resultado = Pokemon.AgregarPokemonManualALaLista("Chikorita", "Planta", 152, "Látigo Cepa", 33);
16     //Assert
17     Assert.IsTrue(resultado);
18 }
19
20 //Pruebo método de extensión(CurarPokemon)
21 [TestMethod]
22 //0 referencias
23 public void ProbarCurarPokemon()
24 {
25     //Arrange
26     string pokemon = "Charmander";
27     string resultadoEsperado = "Charmander ha sido curado y su daño es 0";
28     //Act
29     string resultado = pokemon.CurarPokemon();
30     //Assert
31     Assert.AreEqual(resultadoEsperado, resultado);
32 }
33
34
35
36 }
```

Explorador de pruebas

2 2 0

Prueba	Duración	Rasgos	Mensaje de error
Test (2)	474 ms		
Test (2)	474 ms		
PokemonTest (2)	474 ms		
ProbarCargaDeDatosManualNuevoPokemon	25 ms		
ProbarCurarPokemon	449 ms		

Interfaces: Creamos una interfaz con el método MostrarDato, el cual se aplica en diferentes clases no relacionadas.

```
8
9 public interface IDatos
10 {
11     /// <summary>
12     /// Función que mostrará un dato
13     /// </summary>
14     /// <returns>Retornará un string</returns>
15     string MostrarDato();
16 }
```

Implementaciones de "MostrarDato"

Toda la solución | Agrupar por: solo Proyecto

Código

Entidades (3)

- string Administrador.MostrarDato()
- string Entrenador.MostrarDato()
- string Pokemon.MostrarDato()

Archivos y serialización: Los usamos para guardar la lista en un archivo (txt, xml, json) y también para leer esos mismos archivos guardados.

EscribirJsonLista(List<Pokemon> datos)	LeerJson()
EscribirTxt()	LeerTxt(string archivo)
EscribirXml(List<Pokemon> pokemon)	LeerXml()

Base de datos:

Tenemos la base de datos CentroPokemon con la tabla PokemonAlojados. En la clase PokemonAccesoDatos se trabaja con su implementación, y así poder traer sus elementos a la lista, agregarlos y visualizarlos. Se utilizó select, insert, delete, y update.

```
DeletePokemon(string nombrePokemon)
EliminarPokemonDeBase(string nombrePokemon)
GuardarListaEnBase()
GuardarPokemonEnBase(Pokemon pokemon)
Leer()
LeerCurados()
PokemonAccesoDatos()
UpdateDanioCurar(string nombrePokemon)
```

Delegados: Guardamos el método DesalojarPokemon en el atributo desalojarDelCentro y de esa manera trabajamos con delegados.

```
public delegate bool DesalojarDelCentro(string nombre);
```

```
Pokemon.DesalojarDelCentro desalojarDelCentro = Pokemon.DesalojarPokemon;
```

```
private void btnDesalojarPokemon_Click(object sender, EventArgs e)
{
    if (this.lstPokemon.SelectedItem is not null)
    {
        string pokemonSeleccionado = (string)this.lstPokemon.SelectedItem;
        if (PokemonAccesoDatos.EliminarPokemonDeBase(pokemonSeleccionado) && desalojarDelCentro(pokemonSeleccionado))
        {
            MessageBox.Show($"Se desalojó a {pokemonSeleccionado}, se quitó de la lista y de la base si se encontraba en ella.", "Desalojo exitoso",
                MostrarPokemonEnListaPokemon());
        }
        else
        {
            MessageBox.Show("El pokemon aún no se encuentra curado.", "Elegí un pokemon sano", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
    }
    else
    {
        MessageBox.Show("No se seleccionó ningún pokemon.", "Elegí un pokemon a desalojar", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```