

## Parcial Hotel. Emmanuel Martín.

El programa simula el manejo de registros de un hotel por un usuario para la atención a un huésped a hospedarse en el hotel.

El usuario puede tanto registrar huéspedes, habitaciones, reservaciones. A su vez puede modificar y eliminar datos de los mismos.

Puede también, seguir un registro de las cuentas de las reservaciones de los huéspedes con su registro en JSON y seguir un tracking de la actividad del usuario en el programa.

Adjunto imágenes del programa con explicaciones de funcionamiento. El orden de como ejecutar el programa es indistinto, pero si ciertos datos o funciones podrán realizarse en base a previos registros por el usuario.

Por ejemplo, sin huéspedes registrados, no se podrá generar reservaciones, ni modificar o eliminar estos datos.

### 1- INICIO DE PROGRAMA

The screenshot shows the main interface of the 'Hotel Don Jose' reservation system. At the top, there is a header with the title 'BIENVENIDO A SISTEMA DE RESERVA HOTEL DON JOSE'. Below the header, there is a row of buttons: 'Generar Reservacion', 'Registrar Nuevo Huesped al sistema', 'Registrar Nueva habitacion al sistema', 'Ver Huespedes Registrados', 'Ver Reservaciones', 'Modificar Elimnar Reservaciones', 'Modificar Elimnar Huespedes', and 'Modificar Elimnar Habitaciones'. Below this row, there is a button 'VER INFORME DE CUENTA DESDE JSON'. Below that, there is a message 'No hay nada para mostrar' with a red dot and a dropdown arrow. Below the message, there is a button 'VER TRACKING DE USUARIO'. At the bottom, there is a dropdown menu. The interface is clean and modern, with a light gray background and blue accents.

En este FORM, el usuario puede elegir tanto como ver huéspedes o reservaciones en el sistema, reflejadas en el DataGrid.

Puede elegir generar una reservación, registrar huéspedes o habitaciones.

Puede modificar o eliminar reservaciones, huéspedes, habitaciones.

A su vez, puede ver los informes en JSON sobre las cuentas de las reservaciones de huéspedes, como el tracking de la actividad de usuario en el programa.

**Acá usé tanto tema de Base de datos para las entidades huéspedes, habitaciones y reservaciones. En tema base de datos, utilizo Generic, Interface para intentar implementar el patrón de diseño de repositorio.**

**Uso delegado y evento para generar el tracking del usuario en el programa.**

**Método de extensión para ordenar los huéspedes por Apellido. También hay otros órdenes en la clase de extensión, pero se utiliza el mencionado**

**Use test para verificar validaciones y que se obtengan correctamente huéspedes por su dni desde la base de datos.**

Generé excepciones a lo largo del programa, que son capturadas en catch por Exception y lanzan el mensaje respectivo.

**NOMBRE BASE DE DATOS:**  
**Hotel-Segundo-Parcial-Labo.**  
Dejé en el repo el backUp de la base datos.

Dejo el path del JSON generado.  
`string path = Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "billings.json");`

2- RESERVACIONES

Reservas

Huespedes en Sistema

Jorge Luis Borges - DNI: 12345678

Datos Habitacion

Habitaciones Disponibles

101 - Simple: US\$50

Datos Reservacion

CheckInMonday, November 20,

CheckInMonday, November 20,

GENERAR RESERVA

VER INFORME DE CUENTA DESDE JSON

En esta sección. Si es que hay usuarios registrados en el sistema y habitaciones disponibles, el usuario puede generar reservaciones, eligiendo la habitación y las fechas para ese huésped. A su vez verá reflejado los datos de la cuenta de las reservaciones hechas.

**Acá usé Serialización y archivos para generar el informe JSON**

3- Registro de huéspedes

Menu Resgitro Huesped en Sistema

Datos Huesped

DNI

Nombre

Apellido

Nro Telefono

Guardar

Huespedes

	DNI	Nombre	Apellido	Nro Telefono
▶	25647853	Hanna	Arendt	1134564356
	12345678	Jorge Luis	Borges	1234567324
	18954123	Herman	Hesse	1234567899

En esta sección, podrá ver los huéspedes ya registrados en el sistema a través del datagrid. A su vez rellenando los campos correspondientes, podrá registrar un nuevo huésped al sistema, siempre y cuando este no exista en el sistema.

4- Modificación - borrado Huéspedes

Menu Resgitro Huesped en Sistema

Datos Huesped

Nombre

Hanna

Apellido

Arendt

Nro Telefono

1134564356

Eliminar

Modificar

Huespedes. Doble click sobre el huesped que quiera seleccionar.

	DNI	Nombre	Apellido	Nro Telefono
▶	25647853	Hanna	Arendt	1134564356
	12345678	Jorge Luis	Borges	1234567324
	18954123	Herman	Hesse	1234567899

En esta sección haciendo doble click sobre el huésped que quiera, podrá modificar sus datos. Mientras que si el huésped está seleccionado, podrá eliminarlo. Esto generará una cadena de otros efectos, dependiendo de ese huésped. Por ejemplo.

Si elimina un huésped con una reserva hecha, esta reserva automáticamente será eliminada del sistema y la habitación ocupada pasará a estar disponible.

5- HABITACIONES

Habitaciones

Datos Habitacion

Numero

Tipo

Simple

Guardar

Habitaciones

	Numero	Tipo
▶	101	Simple
	102	Simple
	103	Simple
	104	Simple
	105	Simple
	106	Simple
	107	Simple
	108	Simple
	109	Simple
	110	Simple

En esta sección verá las habitaciones en el sistema y a su vez podrá registrar nuevas al sistema.

6- Modificación - borrado habitaciones.

Habitaciones

Datos Habitacion

Numero

101

Tipo

Simple

Eliminar

Modificar

Habitaciones. Doble click sobre la habitacion que quiera seleccionar.

	Numero	Tipo
▶	101	Simple
	102	Simple
	103	Simple
	104	Simple
	105	Simple
	106	Simple
	107	Simple
	108	Simple
	109	Simple
	110	Simple

Podrá modificar el número o el tipo de habitación. Solo verá las habitaciones disponibles, es decir, que no tengan reservaciones hechas.

7- Modificación - borrado Reservas.

Menu Edicion Reservas

Reservacion

CheckIn

Monday , November : ▾

CheckOut

Monday , November : ▾

Habitacion

101

Guardar

Eliminar

Reservas

	DNI	CheckIn	CheckOut	Nro Habitacion
▶	12345678	20-Nov-23	20-Nov-23	101

Igual que con las anteriores entidades, la reservación seleccionada podrá ser eliminada. En ese caso, la habitación correspondiente pasará a estar disponible nuevamente y el huésped no tendrá reservas a su nombre.

Doble click sobre la reservación para modificar sus datos.

## CÓDIGO CON IMPLEMENTACIONES

Entities\SQLLogic\IDataBaseGenericRepository.cs

```
/// <summary>
/// CLASE ABSTRACTA DE LA CUAL HEREDAN TODOS LOS REPOSITORIOS DE LA BASE DE DATOS
/// EN ESTA CLASE SE DEFINEN LOS METODOS GENERICOS QUE SE USARAN EN TODOS LOS REPOSITORIOS
/// TEMA DE PARCIAL: INTERFACES, TIPO GENERICO Y METODOS ASINCRONOS (ASYNC Y AWAIT) MUTI-HILO
/// </summary>
/// <typeparam name="T"></typeparam>
12 references
public interface IDataBaseGenericRepository<T>
{
    6 references
    Task<List<T>> GetAll();
    6 references
    Task Add(T entity);
    6 references
    Task Update(T entity);
    6 references
    Task Delete(int id);
    7 references
    Task<T> GetById(int id);
}
```

Entities\Handlers\GuestRepository.cs

```
/// <summary>
/// Clase que se encarga de la logica de la base de datos de los huéspedes
/// </summary>
3 references
public class GuestRepository : IDataBaseGenericRepository<Guest>
{
    private readonly ContextDb _contextDb;

    1 reference
    public GuestRepository(ContextDb contextDb)
    {
        _contextDb = contextDb;
    }

    /// <summary>
    /// Agrega un huésped a la base de datos
    /// </summary>
    /// <param name="guest"></param>
    /// <returns></returns>
    4 references
    public async Task Add(Guest guest)
    {
        try
        {
            string query = "INSERT INTO Huespedes (Dni, Nombre, Apellido, NumeroTelefono)" +
                           "values(@dni, @name, @lastName, @phoneNumber)";
            using (var command = await _contextDb.CreateCommand(query))
            {
                command.Parameters.AddWithValue("dni", guest.Dni);
                command.Parameters.AddWithValue("name", guest.Name);
                command.Parameters.AddWithValue("lastName", guest.LastName);
                command.Parameters.AddWithValue("phoneNumber", guest.PhoneNumber);
                await _contextDb.ExecuteNonQuery(command);
            }
        }
    }
}
```

## Entities\Serialization\JSONSerialization.cs

```

3 references
public class JSONSerialization : JSONManagment
{
    /// <summary>
    /// Metodo para serializar una cuenta JSOn
    /// </summary>
    /// <param name="billing"></param>
    /// <exception cref="NotSerializeJsonException"></exception>
    0 references
    public static void SerializeBillings(Billing billing)
    {
        try
        {
            string json = Serialize(billing);
            string path = Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "billings.json");
            File.WriteAllText(path, json);
        }
        catch (Exception ex)
        {
            throw new NotSerializeJsonException("Error", ex.Message);
        }
    }
}

```

## Entities\SQLLogic\ContextDb.cs

```

11 references
public class ContextDb : ConnectionDataBase
{
    /// <summary>
    /// Abre la conexión con la base de datos y crea un comando
    /// </summary>
    /// <param name="query"></param>
    /// <returns></returns>
    15 references
    public async Task<SqlCommand> CreateCommand(string query)
    {
        await this.Open();
        var command = new SqlCommand(query, this._connection);
        return command;
    }

    /// <summary>
    /// Encapsula el ExecuteReaderAsync y crea la dataTable
    /// </summary>
    /// <param name="c"></param>
    /// <returns></returns>
    6 references
    public async Task<DataTable> ExecuteReader(SqlCommand c)
    {
        var reader = await c.ExecuteReaderAsync();
        var dataTable = new DataTable();
        dataTable.Load(reader);
        reader.Close();

        return dataTable;
    }
}

```

## UIHotel\BaseFormTrack.cs

```

5 references
public class BaseFormTrack : Form
{
    public delegate void UserActionTracker(string action);
    public event UserActionTracker ?OnUserTracker;
    /// <summary>
    /// METODO PARA DISPARAR EL EVENTO DE TRACKER
    /// </summary>
    /// <param name="action"></param>
    11 references
    protected void TriggerUserTracker(string action)
    {
        OnUserTracker?.Invoke(action);
    }
}

```

```

public static class GuestsExtensions
{
    /// <summary>
    /// Ordena los huéspedes por apellido
    /// </summary>
    /// <param name="guests"></param>
    /// <returns></returns>
    2 references
    public static List<Guest> OrderGuestByLastName(this List<Guest> guests)
    {
        var sortedGuests = new List<Guest>(guests);
        guests.Sort((x, y) => string.Compare(x.LastName, y.LastName));
        return guests;
    }

    /// <summary>
    /// Ordena los huéspedes por DNI
    /// </summary>
    /// <param name="guests"></param>
    /// <returns></returns>
    0 references
    public static List<Guest> OrderGuestByDNI(this List<Guest> guests)
    {
        var sortedGuests = new List<Guest>(guests);
        guests.Sort((x, y) => string.Compare(x.Dni.ToString(), y.Dni.ToString()));
        return guests;
    }
}

```

```

/// <summary>
/// Evento que se ejecuta al hacer click en el boton de "Huespedes Registrados"
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 reference
private async void btnRegisteredGuests_Click(object sender, EventArgs e)
{
    this.dgvMainData.DataSource = null;
    var guests = await _guestController.GetAllGuests();
    guests.OrderGuestByLastName();
    this.dgvMainData.DataSource = guests;
    this.dgvMainData.Columns[0].HeaderText = "DNI";
    this.dgvMainData.Columns[1].HeaderText = "Nombre";
    this.dgvMainData.Columns[2].HeaderText = "Apellido";
    this.dgvMainData.Columns[3].HeaderText = "Nro Telefono";
    this.dgvMainData.Columns[4].Visible = false;

    this.TriggerUserTracker($"Usuario ve Huespedes registrados en sistema. {DateTime.Now}");
}

```