```
using BussinessLayer;
using Entities;
using System;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Windows.Forms;

namespace Gym
{
    public partial class Planes : Form
    {
        #region Instancias
        //Capa de negocio
        private readonly BussinessPersonas _bussinesPersonas;
        private readonly BussinessPlanes _bussinessPlanes;
        private readonly BussinesClientes _bussinesClientes;
        private readonly BussinessAsistencia _bussinessAsistencia;
        private readonly BussinessPlanesAsignados _bussinesPlanesAsignados;
        private readonly BussinessJornadas _bussinessJornadas;

        //Entities
        private readonly Entities.Clientes _clientes;
        private readonly Entities.Planes _planes;
        private readonly Entities.Asistencias _asistencias;
        private readonly Entities.Planes_Asignados _planesAsignados;
        private readonly Jornadas_Planes _jornadasPlanes;
        private Personas _personas;


        //Clases internas
        private readonly MetodosGenerales _metodosGenerales;
        private readonly Restricciones _restricciones;

        #endregion

        #region Variables
        private string desde;
        private string hasta;
        private int personaLogueada;
        private string buscar;
        private DataSet DsClienteDatos;
        private DataSet DsPlanesAsignados;
        private DataSet DsPlanes;
        private DataTable DtPlanesAsignados;
        private DataTable DtJornadaDePlan;
        private DataTable DtProfesores;
        private DataTable DtPlanes;
        private bool camposVacios = false;
        private bool datosVacios = false;
        private int cliente_ID;
        private int planSeleccionado;
        private int contador;
        private bool todosChk;
        private int pTodosId = -1;
        private int pLunesId = -1;
        private int pMartesId = -1;
        private int pMiercolesId = -1;
        private int pJuevesId = -1;
        private int pViernesId = -1;
        private int pSabadoId = -1;
```

```
        private int idJornada;
        private int check;
        private bool esAlta = true;
        private bool cargarJornada;
        private int idPlanAEditar;
        private string hora;
        private bool contenidoIncorrecto = false;
        private bool duplicidad = false;


        #endregion

        #region Loading
        public Planes(int idPersonaLog)
        {
            InitializeComponent();

            _bussinessPlanes = new BussinessPlanes();
            _bussinesClientes = new BussinessClientes();
            _bussinesPersonas = new BussinessPersonas();
            _bussinessJornadas = new BussinessJornadas();
            _bussinessAsistencia = new BussinessAsistencia();
            _planesAsignados = new Entities.Planes_Asignados();
            _bussinesPlanesAsignados = new BussinessPlanesAsignados();

            _personas = new Personas();
            _planes = new Entities.Planes();
            _clientes = new Entities.Clientes();
            _jornadasPlanes = new Jornadas_Planes();
            _asistencias = new Entities.Asistencias();

            _restricciones = new Restricciones();
            _metodosGenerales = new MetodosGenerales();

            personaLogueada = idPersonaLog;
            btnAsignarPlan.Enabled = false;

            BuscarPlanes();
            BuscarProfesores();
            CargarPlanes();
        }

        #endregion

        #region Encapsulamiento

        private void CargarPlanes()
        {
            dtgvPlanes.Rows.Clear();
            DsPlanes = _bussinessPlanes.GetPlanesActuales(buscar);
            buscar = string.Empty;

            if (DsPlanes.Tables[0].Rows.Count > 0)
            {
                foreach (DataRow dr in DsPlanes.Tables[0].Rows)
                {
                    dtgvPlanes.Rows.Add(dr[0].ToString(), dr[1], dr[2], dr[3]);
                }
            }
        }
```

```
        private void BuscarProfesores()
        {
            DtProfesores = _bussinesPersonas.BuscaProfesores(_personas);
            cmbProfesores.DataSource = DtProfesores;
            cmbProfesores.DisplayMember = "Nombre";
            cmbProfesores.ValueMember = "Empleado_ID";
        }

        private void BuscarPlanes()
        {
            DtPlanes = _bussinessPlanes.GetPlanes(_planes);
            cmbPlanesActivos.DataSource = DtPlanes;
            cmbPlanesActivos.DisplayMember = "Nombre";
            cmbPlanesActivos.ValueMember = "Plan_ID";
        }

        private void AcomodarDatos()
        {
            if (DsClienteDatos.Tables[0].Rows.Count > 0)
            {
                foreach (DataRow dr in DsClienteDatos.Tables[0].Rows)
                {
                    cliente_ID = int.Parse(dr[1].ToString());
                    lblNombre.Text += dr[2].ToString() + " " + dr[3].ToString();
                    lblDocumento.Text += dr[4].ToString();
                    lblTelefono.Text += dr[5].ToString();
                    lblMail.Text += dr[6].ToString();
                    break;
                }
            }
            else
            {
                MessageBox.Show("No hay clientes con el documento ingresado. " +
                    "Por favor, intente de nuevo, o haga primero el registro del cliente.",
                    "Datos no encontrados");
            }
        }
        private void MostrarCliente()
        {
            DsClienteDatos = _bussinesPersonas.GetPersonaPlan(buscar, _personas);
            AcomodarDatos();
            DsPlanesAsignados = _bussinesPlanesAsignados.BuscarPlanesAsignados(cliente_ID);
            AcomodarPlanesAsignados();
            btnAsignarPlan.Enabled = true;
        }
        private void AcomodarPlanesAsignados()
        {
            int i = 0;
            foreach (DataRow dr in DsPlanesAsignados.Tables[0].Rows)
            {
                lblPlanesAsignadosCliente.Text += dr[0].ToString();
                i++;
                if ((i + 1) <= DsPlanesAsignados.Tables[0].Rows.Count)
                {
                    lblPlanesAsignadosCliente.Text += ", ";
                }
            }
        }

        private void AsigarlePlanAlCliente()
        {
```

```
            _planesAsignados.Plan_ID = Convert.ToInt32(cmbPlanesActivos.SelectedValue);
            _planesAsignados.Empleado_ID = personaLogueada;
            _planesAsignados.Cliente_ID = cliente_ID;
            _planesAsignados.Fecha_Inscripcion = DateTime.Now;
            _planesAsignados.Estado = "A";
            _bussinesPlanesAsignados.AsginarPlanAlCliente(_planesAsignados);

            //Acá hacemos la resta de cupos.
            _planes.Plan_ID = Convert.ToInt32(cmbPlanesActivos.SelectedValue);
            _bussinessPlanes.EditarCupoRestante(_planes);
        }

        private void RevisarCamposVacios()
        {
            foreach (Control ctrl in gpDatosPlan.Controls)
            {
                if (ctrl is TextBox box)
                {
                    TextBox t;
                    t = box;
                    if (string.IsNullOrEmpty(t.Text))
                    {
                        camposVacios = true;
                        break;
                    }
                }
                else
                {
                    camposVacios = false;
                }
            }
        }

        private void RegistrarNuevoPlan()
        {
            _planes.Persona_ID = personaLogueada;
            _planes.Nombre = txtNombrePlan.Text;
            _planes.Importe_Plan = Convert.ToDecimal(txtCostoMensual.Text);
            _planes.Empleado_ID = Convert.ToInt32(cmbProfesores.SelectedValue);
            _planes.Duracion = Convert.ToInt32(txtDuracion.Text);
            _planes.Cupo_Total = Convert.ToInt32(txtCupoTotal.Text);
            _planes.Cupo_Restante = Convert.ToInt32(txtCupoTotal.Text);
            _planes.Fecha_Inicio = dtpFechaInicio.Value;
            _planes.Fecha_Alta_Plan = DateTime.Now;
            _planes.Estado = "A";
            _bussinessPlanes.RegistrarNuevoPlan(_planes);
        }

        private void AsignacionHoras()
        {
            int contadorDesde = desde.Length;
            int contadorHasta = hasta.Length;

            if (contadorDesde <= 5 && contadorHasta <= 5)
            {
                _jornadasPlanes.Desde_Hora = _metodosGenerales.ConvertirHoraAfecha(desde);
                _jornadasPlanes.Hasta_Hora = _metodosGenerales.ConvertirHoraAfecha(hasta);
            }
            else
            {
                MessageBox.Show("El campo de horas, está mal registrado. Desde registrarse con
```

```
este formato: ##:##", "Formato incorrecto", MessageBoxButtons.OK, MessageBoxIcon.Warning);
            }
        }

        private void EditarPlanes()
        {
            _planes.Plan_ID = idPlanAEditar;
            _planes.Nombre = txtNombrePlan.Text;
            _planes.Importe_Plan = Convert.ToDecimal(txtCostoMensual.Text);
            _planes.Duracion = Convert.ToInt32(txtDuracion.Text);
            _planes.Cupo_Total = Convert.ToInt32(txtCupoTotal.Text);
            _planes.Fecha_Inicio = dtpFechaInicio.Value;
            _planes.Empleado_ID = Convert.ToInt32(cmbProfesores.SelectedValue);

            _bussinessPlanes.EditarPlan(_planes);
        }

        private void GuardarJornada()
        {
            foreach (Control chk in this.gbJornadaPlanes.Controls)
            {
                if (chk is CheckBox box)
                {
                    CheckBox c;
                    c = box;
                    //Y luego de cada checkbox le asignamos el horario.
                    //Lunes
                    if (c.Checked)
                    {
                        switch (c.Text)
                        {
                            case "Lunes":
                                _jornadasPlanes.Dia = "Lunes";
                                _jornadasPlanes.Jornada_Plan_ID = pLunesId;
                                desde = txtDesdeLunes.Text.ToString();
                                hasta = txtHastaLunes.Text.ToString();
                                AsignacionHoras();
                                break;
                            case "Martes":
                                _jornadasPlanes.Dia = "Martes";
                                _jornadasPlanes.Jornada_Plan_ID = pMartesId;
                                desde = txtDesdeMartes.Text.ToString();
                                hasta = txtHastaMartes.Text.ToString();
                                AsignacionHoras();
                                break;
                            case "Miercoles":
                                _jornadasPlanes.Dia = "Miercoles";
                                _jornadasPlanes.Jornada_Plan_ID = pMiercolesId;
                                desde = txtDesdeMiercoles.Text.ToString();
                                hasta = txtHastaMiercoles.Text.ToString();
                                AsignacionHoras();
                                break;
                            case "Jueves":
                                _jornadasPlanes.Dia = "Jueves";
                                _jornadasPlanes.Jornada_Plan_ID = pJuevesId;
                                desde = txtDesdeJueves.Text.ToString();
                                hasta = txtHastaJueves.Text.ToString();
                                AsignacionHoras();
                                break;
                            case "Viernes":
                                _jornadasPlanes.Dia = "Viernes";
```

```
                            _jornadasPlanes.Jornada_Plan_ID = pViernesId;
                            desde = txtDesdeViernes.Text.ToString();
                            hasta = txtHastaViernes.Text.ToString();
                            AsignacionHoras();
                            break;
                        case "Sabado":
                            _jornadasPlanes.Dia = "Sabado";
                            _jornadasPlanes.Jornada_Plan_ID = pSabadoId;
                            desde = txtDesdeSabado.Text.ToString();
                            hasta = txtHastaSabado.Text.ToString();
                            AsignacionHoras();
                            break;
                    }
                    _jornadasPlanes.Estado = "A";
                    _jornadasPlanes.Plan_ID = idJornada;
                    if (_jornadasPlanes.Plan_ID != -1)
                    {
                        _bussinessJornadas.EditarJornadaPlan(_jornadasPlanes);
                    }
                    else
                    {
                        _bussinessJornadas.AltaJornadaPlan(_jornadasPlanes);
                    }
                    desde = string.Empty;
                    hasta = string.Empty;
                }
            }
        }
    }
    private void JornadasManagement()
    {
        //Acá editamos
        if (cargarJornada)
        {
            GuardarJornada();
        }
        else
        {
            //Acá creamos:

            //Ahora tenemos 2 opciones.
            // 1 es que se haya seleccionado el check "todos"
            if (todosChk)
            {
                _jornadasPlanes.Plan_ID = idJornada;
                _jornadasPlanes.Dia = "Todos";
                _jornadasPlanes.Estado = "A";
                desde = txtDesdeLunes.Text.ToString();
                hasta = txtHastaLunes.Text.ToString();
                AsignacionHoras();

                _bussinessJornadas.AltaJornadaPlan(_jornadasPlanes);
            }
            else
            {
                //Y la otra es que se hayan seleccionado varios dias
                //Vamos a verificar los checkbox que se hayan seleccionado
                GuardarJornada();

            }
        }
```

```
        }

        private void GetJornadaDePlan()
        {
            bool esEmpleado = false;
            bool darBaja = false;
            int plan_ID = Convert.ToInt32(cmbPlanesActivos.SelectedValue);
            Jornadas jn = new Jornadas(plan_ID, esEmpleado, darBaja);
            jn.ShowDialog();
        }
        private void RevisarTodosChk()
        {
            foreach (Control chk in gbJornadaPlanes.Controls)
            {
                if (chk is CheckBox box)
                {
                    CheckBox c;
                    c = box;
                    if (c.Checked)
                    {
                        contador++;
                    }
                }
            }
        }
        private void ChkDias()
        {
            RevisarTodosChk();
            if (chkTodos.Checked)
            {
                chkTodos.Checked = false;
                todosChk = false;
            }
            else
            {
                if (contador > 5 && !chkTodos.Checked)
                {
                    ActivarTodoChk();
                }
            }
        }
        private void ActivarTodoChk()
        {
            if (!chkTodos.Checked && contador < 7 && !todosChk)
            {
                foreach (Control chk in gbJornadaPlanes.Controls)
                {
                    if (chk is CheckBox box)
                    {
                        CheckBox c;
                        c = box;
                        c.Checked = true;
                    }
                }
                todosChk = true;
            }
            else
            {
                if (todosChk)
                {
                    //Si el chk que dice "Seleccionar todos" está seleccionado
```

```
                //Entonces todos los otros chk se van a seleccionar
                //Caso contrario, se les quitará el chkeck
                foreach (Control chk in this.gbJornadaPlanes.Controls)
                {
                    if (chk is CheckBox box)
                    {
                        CheckBox c;
                        c = box;
                        //Deshabilito el check en cada checkbox
                        c.Checked = false;
                    }
                }
                todosChk = false;
            }
            else
            {
                foreach (Control chk in this.gbJornadaPlanes.Controls)
                {
                    if (chk is CheckBox box)
                    {
                        CheckBox c;
                        c = box;
                        c.Checked = true;
                    }
                }
                todosChk = true;
            }
        }
    }
    private void VerificarChk()
    {
        switch (check)
        {
            case 0:
                ActivarTodoChk();
                break;
            case 1:
                ChkDias();
                break;
            case 2:
                ChkDias();
                break;
            case 3:
                ChkDias();
                break;
            case 4:
                ChkDias();
                break;
            case 5:
                ChkDias();
                break;
            case 6:
                ChkDias();
                break;
        }
        contador = 0;
    }
    private void ResetControls()
    {
        lblCostoMensual.Text = "Costo mensual: $";
        lblCuposRestantes.Text = "0";
```

```
            lblCuposTotales.Text = "0";
        }
        private void BuscarDatosPlan()
        {
            ResetControls();
            _planes.Plan_ID = Convert.ToInt32(cmbPlanesActivos.SelectedValue);
            _bussinessPlanes.GetDatoPlan(_planes);
            planSeleccionado = _planes.Plan_ID;
            lblCostoMensual.Text += _planes.Importe_Plan.ToString();
            lblCuposRestantes.Text = _planes.Cupo_Restante.ToString();
            lblCuposTotales.Text = _planes.Cupo_Total.ToString();
        }

        private void ResetControlsCliente()
        {
            lblNombre.Text = "Nombre: ";
            lblDocumento.Text = "DNI: ";
            lblTelefono.Text = "Telefono: ";
            lblMail.Text = "Mail: ";
            lblPlanesAsignadosCliente.Text = string.Empty;
        }

        private void ResetControlsAltaPlan()
        {
            foreach (Control ctrl in gpDatosPlan.Controls)
            {
                if (ctrl is TextBox box)
                {
                    TextBox txt;
                    txt = box;
                    txt.Text = string.Empty;
                }

                if (ctrl is CheckBox chk)
                {
                    CheckBox c;
                    c = chk;
                    c.Checked = false;
                }
            }
            foreach (Control ctrl in gbJornadaPlanes.Controls)
            {
                if (ctrl is TextBox box)
                {
                    TextBox txt;
                    txt = box;
                    txt.Text = "hh:mm";
                    txt.ForeColor = Color.DimGray;
                }

                if (ctrl is CheckBox chk)
                {
                    CheckBox c;
                    c = chk;
                    c.Checked = false;
                }
            }
        }

        private void ComprobarDuplicidadDePlan()
        {
```

```
            int idplan = Convert.ToInt32(cmbPlanesActivos.SelectedValue);
            duplicidad = _bussinesPlanesAsignados.BuscarDuplicidad(idplan, cliente_ID);
        }

        #endregion

        #region Eventos Keypress
        private void txtBuscarClase_KeyPress(object sender, KeyPressEventArgs e)
        {
            buscar = txtBuscarClase.Text;
            if (e.KeyChar == Convert.ToChar(Keys.Enter))
            {
                CargarPlanes();
            }
        }
        private void txtBuscarCliente_KeyPress(object sender, KeyPressEventArgs e)
        {
            buscar = txtBuscarCliente.Text;
            _restricciones.SoloNumeros(e, buscar);
            if (e.KeyChar == Convert.ToChar(Keys.Enter))
            {
                if (!datosVacios)
                {
                    if (string.IsNullOrEmpty(buscar))
                    {
                        MessageBox.Show("Tiene que buscar con un número de documento válido.
No se adminten campos vacíos.",
                            "Gestión en proceso", MessageBoxButtons.OKCancel);
                    }
                    else
                    {
                        ResetControlsCliente();
                        MostrarCliente();
                        datosVacios = true;
                    }
                }
                else
                {
                    DialogResult result = MessageBox.Show("Tiene un cliente cargado. ¿Quiere
interrumpir la gestión y buscar un cliente nuevo?",
                        "Gestión en proceso", MessageBoxButtons.OKCancel);
                    if (result == DialogResult.OK)
                    {
                        if (string.IsNullOrEmpty(buscar))
                        {
                            MessageBox.Show("Tiene que buscar con un número de documento
válido. No se adminten campos vacíos.",
                                "Gestión en proceso", MessageBoxButtons.OKCancel);
                        }
                        else
                        {
                            ResetControlsCliente();
                            MostrarCliente();
                            buscar = string.Empty;
                        }
                    }
                }
            }
        }

        #endregion
```

```
        #region Alta de Planes

        private void btnAltaPlan_Click(object sender, EventArgs e)
        {
            RevisarCamposVacios();
            VerificarFormatoHora();
            if (camposVacios)
            {
                MessageBox.Show("Para registrar un nuevo plan, no deben haber campos vacíos",
                    "Campos incompletos");
            }
            else
            {
                if (esAlta)
                {
                    RegistrarNuevoPlan();
                    _bussinessPlanes.GetLastID(_planes);
                    idPlanAEditar = _planes.Plan_ID;
                }
                else
                {
                    EditarPlanes();
                }

                JornadasManagement();
                MessageBox.Show("El plan se ha guardado con éxito!",
                    "Registro de Planes");

                //Una vez dado de alta, actualizamos la lista.
                CargarPlanes();
                ResetControlsAltaPlan();
                esAlta = true;
            }
        }

        private void VerificarFormatoHora()
        {

            foreach (Control ctrl in gbJornadaPlanes.Controls)
            {
                if (ctrl is TextBox box)
                {
                    TextBox t;
                    t = box;
                    if (t.Text.Length < 5)
                    {
                        contenidoIncorrecto = true; ;
                        break;
                    }
                    else
                    {
                        contenidoIncorrecto = false;
                    }
                }
            }
        }

        #endregion

        #region Focus Textbox
        private void txtBuscarCliente_Enter(object sender, EventArgs e)
```

```
        {
            if (txtBuscarCliente.Text == "DNI")
            {
                txtBuscarCliente.Text = string.Empty;
                txtBuscarCliente.ForeColor = Color.Black;
            }
        }

        private void txtBuscarCliente_Leave(object sender, EventArgs e)
        {
            if (string.IsNullOrEmpty(txtBuscarCliente.Text))
            {
                txtBuscarCliente.Text = "DNI";
                txtBuscarCliente.ForeColor = Color.DimGray;
            }
        }


        private void txtBuscarClase_Enter(object sender, EventArgs e)
        {
            if (txtBuscarClase.Text == "Buscar Clase")
            {
                txtBuscarClase.Text = string.Empty;
                txtBuscarClase.ForeColor = Color.Black;
            }
        }

        private void txtBuscarClase_Leave(object sender, EventArgs e)
        {
            if (string.IsNullOrEmpty(txtBuscarClase.Text))
            {
                txtBuscarClase.Text = "Buscar Clase";
                txtBuscarClase.ForeColor = Color.DimGray;
            }
        }

        #endregion

        #region Asignación de planes a un cliente
        private void btnAsignarPlan_Click(object sender, EventArgs e)
        {
            if (camposVacios)
            {
                MessageBox.Show("No hay cliente seleccionado para asignarle un plan." +
                    "Por favor, busque el cliente primero, y luego asigne el plan.");
            }
            else
            {
                ComprobarDuplicidadDePlan();
                if (duplicidad)
                {
                    MessageBox.Show("Este plan ya está asignado al cliente. Por favor,
seleccione otro plan", "Duplicidad encontrada");
                }
                else
                {
                    if (Convert.ToInt32(lblCuposRestantes.Text) == 0 &&
Convert.ToInt32(lblCuposTotales.Text) > 0)
                    {
                        MessageBox.Show("No hay cupos disponibles para esta clase.");
                    }
                    else
```

```
                    {
                        //Asignarle el plan.
                        AsigarlePlanAlCliente();

                        //Actualizar planes
                        cmbPlanesActivos_SelectionChangeCommitted(sender, e);

                        string lblnombre = lblNombre.Text;
                        string nombre = lblnombre.Substring(8, lblnombre.Length - 8);
                        string plan = Convert.ToString(cmbPlanesActivos.Text);

                        MessageBox.Show($"Se asignó correctamente el plan: {plan}, para el
alumno: {nombre}", "Asignación de Planes");
                    }
                }
            }
        }
        private void lblVerJornadas_LinkClicked(object sender, LinkLabelLinkClickedEventArgs
e)
        {
            if (cmbProfesores.Items.Count > 0)
            {
                //traemos la hornada del plan seleccionado.
                GetJornadaDePlan();
            }
            else
            {
                MessageBox.Show("Debe seleccionar primero un plan para poder buscar los
horarios.", "Horarios de Planes", MessageBoxButtons.OK);
            }
        }
        private void cmbPlanesActivos_SelectionChangeCommitted(object sender, EventArgs e)
        {
            //Me busca el precio, cupo total y restante.
            BuscarDatosPlan();
            //Busca los horarios
        }

        #endregion

        #region Selección de checks
        private void chkTodos_Click(object sender, EventArgs e)
        {
            check = 0;
            VerificarChk();
        }
        private void chkLunes_Click(object sender, EventArgs e)
        {
            check = 1;
            VerificarChk();
        }
        private void chkMartes_Click(object sender, EventArgs e)
        {
            check = 2;
            VerificarChk();
            if (chkMartes.Checked && !chkTodos.Checked)
            {
                txtDesdeMartes.Enabled = true;
                txtHastaMartes.Enabled = true;
            }
            else
            {
```

```
            txtDesdeMartes.Enabled = false;
            txtHastaMartes.Enabled = false;
        }
    }
    private void chkMiercoles_Click(object sender, EventArgs e)
    {
        check = 3;
        VerificarChk();
        if (chkMiercoles.Checked && !chkTodos.Checked)
        {
            txtDesdeMiercoles.Enabled = true;
            txtHastaMiercoles.Enabled = true;
        }
        else
        {
            txtDesdeMiercoles.Enabled = false;
            txtHastaMiercoles.Enabled = false;
        }
    }
    private void chkJueves_Click(object sender, EventArgs e)
    {
        check = 4;
        VerificarChk();
        if (chkJueves.Checked && !chkTodos.Checked)
        {
            txtDesdeJueves.Enabled = true;
            txtHastaJueves.Enabled = true;
        }
        else
        {
            txtHastaJueves.Enabled = false;
            txtDesdeJueves.Enabled = false;
        }
    }
    private void chkViernes_Click(object sender, EventArgs e)
    {
        check = 5;
        VerificarChk();
        if (chkViernes.Checked && !chkTodos.Checked)
        {
            txtDesdeViernes.Enabled = true;
            txtHastaViernes.Enabled = true;
        }
        else
        {
            txtDesdeViernes.Enabled = false;
            txtHastaViernes.Enabled = false;
        }
    }
    private void chkSabado_Click(object sender, EventArgs e)
    {
        check = 6;
        VerificarChk();
        if (chkSabado.Checked && !chkTodos.Checked)
        {
            txtDesdeSabado.Enabled = true;
            txtHastaSabado.Enabled = true;
        }
        else
        {
            txtDesdeSabado.Enabled = false;
```

```
            txtHastaSabado.Enabled = false;
        }
    }

    #endregion

    #region TextBox de Hora

    //Lunes
    private void txtDesdeLunes_Enter(object sender, EventArgs e)
    {
        if (txtDesdeLunes.Text == "hh:mm")
        {
            txtDesdeLunes.Text = string.Empty;
            txtDesdeLunes.ForeColor = Color.Black;
        }
    }
    private void txtDesdeLunes_Leave(object sender, EventArgs e)
    {
        //Por el lado contrario, cuando se quita el foco
        //del control, lo volvemos a su estado y color inicial

        if (string.IsNullOrEmpty(txtDesdeLunes.Text))
        {
            txtDesdeLunes.Text = "hh:mm";
            txtDesdeLunes.ForeColor = Color.DimGray;
        }
    }
    private void txtHastaLunes_Enter(object sender, EventArgs e)
    {
        if (txtHastaLunes.Text == "hh:mm")
        {
            txtHastaLunes.Text = string.Empty;
            txtHastaLunes.ForeColor = Color.Black;
        }
    }
    private void txtHastaLunes_Leave(object sender, EventArgs e)
    {
        //Por el lado contrario, cuando se quita el foco
        //del control, lo volvemos a su estado y color inicial

        if (string.IsNullOrEmpty(txtHastaLunes.Text))
        {
            txtHastaLunes.Text = "hh:mm";
            txtHastaLunes.ForeColor = Color.DimGray;
        }
    }

    //Martes
    private void txtDesdeMartes_Enter(object sender, EventArgs e)
    {
        if (txtDesdeMartes.Text == "hh:mm")
        {
            txtDesdeMartes.Text = string.Empty;
            txtDesdeMartes.ForeColor = Color.Black;
        }
    }
    private void txtDesdeMartes_Leave(object sender, EventArgs e)
    {
        //Por el lado contrario, cuando se quita el foco
        //del control, lo volvemos a su estado y color inicial
```

```
        if (string.IsNullOrEmpty(txtDesdeMartes.Text))
        {
            txtDesdeMartes.Text = "hh:mm";
            txtDesdeMartes.ForeColor = Color.DimGray;
        }
    }
    private void txtHastaMartes_Enter(object sender, EventArgs e)
    {
        if (txtHastaMartes.Text == "hh:mm")
        {
            txtHastaMartes.Text = string.Empty;
            txtHastaMartes.ForeColor = Color.Black;
        }
    }
    private void txtHastaMartes_Leave(object sender, EventArgs e)
    {
        //Por el lado contrario, cuando se quita el foco
        //del control, lo volvemos a su estado y color inicial

        if (string.IsNullOrEmpty(txtHastaMartes.Text))
        {
            txtHastaMartes.Text = "hh:mm";
            txtHastaMartes.ForeColor = Color.DimGray;
        }
    }

    //Miércoles
    private void txtDesdeMiercoles_Enter(object sender, EventArgs e)
    {
        if (txtDesdeMiercoles.Text == "hh:mm")
        {
            txtDesdeMiercoles.Text = string.Empty;
            txtDesdeMiercoles.ForeColor = Color.Black;
        }
    }
    private void txtDesdeMiercoles_Leave(object sender, EventArgs e)
    {
        //Por el lado contrario, cuando se quita el foco
        //del control, lo volvemos a su estado y color inicial

        if (string.IsNullOrEmpty(txtDesdeMiercoles.Text))
        {
            txtDesdeMiercoles.Text = "hh:mm";
            txtDesdeMiercoles.ForeColor = Color.DimGray;
        }
    }
    private void txtHastaMiercoles_Enter(object sender, EventArgs e)
    {
        if (txtHastaMiercoles.Text == "hh:mm")
        {
            txtHastaMiercoles.Text = string.Empty;
            txtHastaMiercoles.ForeColor = Color.Black;
        }
    }
    private void txtHastaMiercoles_Leave(object sender, EventArgs e)
    {
        //Por el lado contrario, cuando se quita el foco
        //del control, lo volvemos a su estado y color inicial

        if (string.IsNullOrEmpty(txtHastaMiercoles.Text))
```

```
        {
            txtHastaMiercoles.Text = "hh:mm";
            txtHastaMiercoles.ForeColor = Color.DimGray;
        }
    }

    //Jueves
    private void txtDesdeJueves_Enter(object sender, EventArgs e)
    {
        if (txtDesdeJueves.Text == "hh:mm")
        {
            txtDesdeJueves.Text = string.Empty;
            txtDesdeJueves.ForeColor = Color.Black;
        }
    }
    private void txtDesdeJueves_Leave(object sender, EventArgs e)
    {
        //Por el lado contrario, cuando se quita el foco
        //del control, lo volvemos a su estado y color inicial

        if (string.IsNullOrEmpty(txtDesdeJueves.Text))
        {
            txtDesdeJueves.Text = "hh:mm";
            txtDesdeJueves.ForeColor = Color.DimGray;
        }
    }
    private void txtHastaJueves_Enter(object sender, EventArgs e)
    {
        if (txtHastaJueves.Text == "hh:mm")
        {
            txtHastaJueves.Text = string.Empty;
            txtHastaJueves.ForeColor = Color.Black;
        }
    }
    private void txtHastaJueves_Leave(object sender, EventArgs e)
    {
        //Por el lado contrario, cuando se quita el foco
        //del control, lo volvemos a su estado y color inicial

        if (string.IsNullOrEmpty(txtHastaJueves.Text))
        {
            txtHastaJueves.Text = "hh:mm";
            txtHastaJueves.ForeColor = Color.DimGray;
        }
    }

    //Viernes
    private void txtDesdeViernes_Enter(object sender, EventArgs e)
    {
        if (txtDesdeViernes.Text == "hh:mm")
        {
            txtDesdeViernes.Text = string.Empty;
            txtDesdeViernes.ForeColor = Color.Black;
        }
    }
    private void txtDesdeViernes_Leave(object sender, EventArgs e)
    {
        //Por el lado contrario, cuando se quita el foco
        //del control, lo volvemos a su estado y color inicial

        if (string.IsNullOrEmpty(txtDesdeViernes.Text))
```

```
        {
            txtDesdeViernes.Text = "hh:mm";
            txtDesdeViernes.ForeColor = Color.DimGray;
        }
    }
    private void txtHastaViernes_Enter(object sender, EventArgs e)
    {
        if (txtHastaViernes.Text == "hh:mm")
        {
            txtHastaViernes.Text = string.Empty;
            txtHastaViernes.ForeColor = Color.Black;
        }
    }
    private void txtHastaViernes_Leave(object sender, EventArgs e)
    {
        //Por el lado contrario, cuando se quita el foco
        //del control, lo volvemos a su estado y color inicial

        if (string.IsNullOrEmpty(txtHastaViernes.Text))
        {
            txtHastaViernes.Text = "hh:mm";
            txtHastaViernes.ForeColor = Color.DimGray;
        }
    }

    //Sábado
    private void txtDesdeSabado_Enter(object sender, EventArgs e)
    {
        if (txtDesdeSabado.Text == "hh:mm")
        {
            txtDesdeSabado.Text = string.Empty;
            txtDesdeSabado.ForeColor = Color.Black;
        }
    }
    private void txtDesdeSabado_Leave(object sender, EventArgs e)
    {
        //Por el lado contrario, cuando se quita el foco
        //del control, lo volvemos a su estado y color inicial

        if (string.IsNullOrEmpty(txtDesdeSabado.Text))
        {
            txtDesdeSabado.Text = "hh:mm";
            txtDesdeSabado.ForeColor = Color.DimGray;
        }
    }
    private void txtHastaSabado_Enter(object sender, EventArgs e)
    {
        if (txtHastaSabado.Text == "hh:mm")
        {
            txtHastaSabado.Text = string.Empty;
            txtHastaSabado.ForeColor = Color.Black;
        }
    }
    private void txtHastaSabado_Leave(object sender, EventArgs e)
    {
        //Por el lado contrario, cuando se quita el foco
        //del control, lo volvemos a su estado y color inicial

        if (string.IsNullOrEmpty(txtHastaSabado.Text))
        {
            txtHastaSabado.Text = "hh:mm";
```

```
            txtHastaSabado.ForeColor = Color.DimGray;
        }
    }


    private string AgregarColon(string texto)
    {
        //verificamos si el texto ingresado hasta el momento tiene 2 caracteres
        //si es así, se le agregan los 2 puntos, para evitar ponerlo a cada momento
        //cuando están poniendo los horarios.
        if (texto.Length == 2)
        {
            texto += ":";
        }

        return texto;
    }

    private void txtDesdeLunes_KeyUp(object sender, KeyEventArgs e)
    {
        hora = txtDesdeLunes.Text;
        txtDesdeLunes.Text = AgregarColon(hora);
        txtDesdeLunes.Select(txtDesdeLunes.Text.Length, 0);
    }

    private void txtHastaLunes_KeyUp(object sender, KeyEventArgs e)
    {
        hora = txtHastaLunes.Text;
        txtHastaLunes.Text = AgregarColon(hora);
        txtHastaLunes.Select(txtHastaLunes.Text.Length, 0);
    }

    private void txtDesdeMartes_KeyUp(object sender, KeyEventArgs e)
    {
        hora = txtDesdeMartes.Text;
        txtDesdeMartes.Text = AgregarColon(hora);
        txtDesdeMartes.Select(txtDesdeMartes.Text.Length, 0);
    }

    private void txtHastaMartes_KeyUp(object sender, KeyEventArgs e)
    {
        hora = txtHastaMartes.Text;
        txtHastaMartes.Text = AgregarColon(hora);
        txtHastaMartes.Select(txtHastaMartes.Text.Length, 0);
    }

    private void txtDesdeMiercoles_KeyUp(object sender, KeyEventArgs e)
    {
        hora = txtDesdeMiercoles.Text;
        txtDesdeMiercoles.Text = AgregarColon(hora);
        txtDesdeMiercoles.Select(txtDesdeMiercoles.Text.Length, 0);
    }

    private void txtHastaMiercoles_KeyUp(object sender, KeyEventArgs e)
    {
        hora = txtHastaMiercoles.Text;
        txtHastaMiercoles.Text = AgregarColon(hora);
        txtHastaMiercoles.Select(txtHastaMiercoles.Text.Length, 0);
    }

    private void txtDesdeJueves_KeyUp(object sender, KeyEventArgs e)
```

```
        {
            hora = txtDesdeJueves.Text;
            txtDesdeJueves.Text = AgregarColon(hora);
            txtDesdeJueves.Select(txtDesdeJueves.Text.Length, 0);
        }

        private void txtHastaJueves_KeyUp(object sender, KeyEventArgs e)
        {
            hora = txtHastaJueves.Text;
            txtHastaJueves.Text = AgregarColon(hora);
            txtHastaJueves.Select(txtHastaJueves.Text.Length, 0);
        }

        private void txtDesdeViernes_KeyUp(object sender, KeyEventArgs e)
        {
            hora = txtDesdeViernes.Text;
            txtDesdeViernes.Text = AgregarColon(hora);
            txtDesdeViernes.Select(txtDesdeViernes.Text.Length, 0);
        }

        private void txtHastaViernes_KeyUp(object sender, KeyEventArgs e)
        {
            hora = txtHastaViernes.Text;
            txtHastaViernes.Text = AgregarColon(hora);
            txtHastaViernes.Select(txtHastaViernes.Text.Length, 0);
        }

        private void txtDesdeSabado_KeyUp(object sender, KeyEventArgs e)
        {
            hora = txtDesdeSabado.Text;
            txtDesdeSabado.Text = AgregarColon(hora);
            txtDesdeSabado.Select(txtDesdeSabado.Text.Length, 0);
        }

        private void txtHastaSabado_KeyUp(object sender, KeyEventArgs e)
        {
            hora = txtHastaSabado.Text;
            txtHastaSabado.Text = AgregarColon(hora);
            txtHastaSabado.Select(txtHastaSabado.Text.Length, 0);
        }

        private void txtDesdeLunes_KeyPress(object sender, KeyPressEventArgs e)
        {
            _restricciones.Horarios(e);
        }

        private void txtHastaLunes_KeyPress(object sender, KeyPressEventArgs e)
        {
            _restricciones.Horarios(e);
        }

        private void txtDesdeMartes_KeyPress(object sender, KeyPressEventArgs e)
        {
            _restricciones.Horarios(e);
        }

        private void txtHastaMartes_KeyPress(object sender, KeyPressEventArgs e)
        {
            _restricciones.Horarios(e);
        }
```

```
        private void txtDesdeMiercoles_KeyPress(object sender, KeyPressEventArgs e)
        {
            _restricciones.Horarios(e);
        }

        private void txtHastaMiercoles_KeyPress(object sender, KeyPressEventArgs e)
        {
            _restricciones.Horarios(e);
        }

        private void txtDesdeJueves_KeyPress(object sender, KeyPressEventArgs e)
        {
            _restricciones.Horarios(e);
        }

        private void txtHastaJueves_KeyPress(object sender, KeyPressEventArgs e)
        {
            _restricciones.Horarios(e);
        }

        private void txtDesdeViernes_KeyPress(object sender, KeyPressEventArgs e)
        {
            _restricciones.Horarios(e);
        }

        private void txtHastaViernes_KeyPress(object sender, KeyPressEventArgs e)
        {
            _restricciones.Horarios(e);
        }

        private void txtDesdeSabado_KeyPress(object sender, KeyPressEventArgs e)
        {
            _restricciones.Horarios(e);
        }

        private void txtHastaSabado_KeyPress(object sender, KeyPressEventArgs e)
        {
            _restricciones.Horarios(e);
        }

        #endregion

        #region Eventos de botones
        private void btnEliminarPlanes_Click(object sender, EventArgs e)
        {
            string lim = lblPlanesAsignadosCliente.Text;
            EditarPlanesDeCliente ep = new EditarPlanesDeCliente(lim);
            ep.ShowDialog();
        }

        private void btnEditarPlan_Click(object sender, EventArgs e)
        {
            esAlta = false;
            idPlanAEditar = Convert.ToInt32(dtgvPlanes.CurrentRow.Cells[0].Value);
            CargarPlanParaEdicion();
            GetJornadaPlan();
        }

        private void CargarPlanParaEdicion()
        {
            //traigo el plan con el ID
```

```
            _planes.Plan_ID = Convert.ToInt32(dtgvPlanes.CurrentRow.Cells[0].Value);
            _bussinessPlanes.GetPlanUnico(_planes);

            //Vamos a asignar los datos del plan en cada cuadro de texto.
            idPlanAEditar = _planes.Plan_ID;
            txtNombrePlan.Text = _planes.Nombre.ToString();
            txtCostoMensual.Text = _planes.Importe_Plan.ToString();
            txtDuracion.Text = _planes.Duracion.ToString();
            txtCupoTotal.Text = _planes.Cupo_Total.ToString();
            dtpFechaInicio.Value = _planes.Fecha_Inicio;
            cmbProfesores.SelectedValue = _planes.Empleado_ID;
        }


        private void GetJornadaPlan()
        {
            _jornadasPlanes.Plan_ID = idPlanAEditar;
            //Una vez asignado, busco la jornada
            //Y la asigno a un datatable, para poder manejar el contenido.
            DtJornadaDePlan = _bussinessJornadas.GetJornadaPlan(_jornadasPlanes);

            if (DtJornadaDePlan.Rows.Count > 0)
            {
                cargarJornada = true;
                //cargo la jornada en los textbox correspondientes
                CargarJornada();
            }
            else
            {
                cargarJornada = false;
            }
        }

        private void CargarJornada()
        {
            foreach (DataRow dr in DtJornadaDePlan.Rows)
            {
                //Vamos a switchear cada vuelta para que vaya
                //colocando los datos cada vuelta donde corresponda.
                switch (dr[2].ToString())
                {
                    case "Todos":
                        chkTodos.Checked = true;
                        //Este es el Id de la jornada que se carga
                        pTodosId = Convert.ToInt32(dr[0]);
                        txtDesdeLunes.Text =
Convert.ToString(_metodosGenerales.ConvertirfechaAhora(dr[3].ToString()));
                        txtHastaLunes.Text =
Convert.ToString(_metodosGenerales.ConvertirfechaAhora(dr[4].ToString()));
                        txtDesdeLunes.ForeColor = Color.Black;
                        txtHastaLunes.ForeColor = Color.Black;
                        VerificarChk();
                        break;
                    case "Lunes":
                        txtDesdeLunes.Text =
Convert.ToString(_metodosGenerales.ConvertirfechaAhora(dr[3].ToString()));
Convert.ToString(dr[3]);
                        txtHastaLunes.Text =
Convert.ToString(_metodosGenerales.ConvertirfechaAhora(dr[4].ToString()));
                        txtDesdeLunes.ForeColor = Color.Black;
                        txtHastaLunes.ForeColor = Color.Black;
                        pLunesId = Convert.ToInt32(dr[0]);
```

```
                        chkLunes.Checked = true;
                        break;
                    case "Martes":
                        txtDesdeMartes.Text =
Convert.ToString(_metodosGenerales.ConvertirfechaAhora(dr[3].ToString()));
                        txtHastaMartes.Text =
Convert.ToString(_metodosGenerales.ConvertirfechaAhora(dr[4].ToString()));
                        txtDesdeMartes.ForeColor = Color.Black;
                        txtHastaMartes.ForeColor = Color.Black;
                        pMartesId = Convert.ToInt32(dr[0]);
                        chkMartes.Checked = true;
                        break;
                    case "Miercoles":
                        txtDesdeMiercoles.Text =
Convert.ToString(_metodosGenerales.ConvertirfechaAhora(dr[3].ToString()));
                        txtHastaMiercoles.Text =
Convert.ToString(_metodosGenerales.ConvertirfechaAhora(dr[4].ToString()));
                        txtDesdeMiercoles.ForeColor = Color.Black;
                        txtHastaMiercoles.ForeColor = Color.Black;
                        pMiercolesId = Convert.ToInt32(dr[0]);
                        chkMiercoles.Checked = true;
                        break;
                    case "Jueves":
                        txtDesdeJueves.Text =
Convert.ToString(_metodosGenerales.ConvertirfechaAhora(dr[3].ToString()));
                        txtHastaJueves.Text =
Convert.ToString(_metodosGenerales.ConvertirfechaAhora(dr[4].ToString()));
                        txtDesdeJueves.ForeColor = Color.Black;
                        txtHastaJueves.ForeColor = Color.Black;
                        pJuevesId = Convert.ToInt32(dr[0]);
                        chkJueves.Checked = true;
                        break;
                    case "Viernes":
                        txtDesdeViernes.Text =
Convert.ToString(_metodosGenerales.ConvertirfechaAhora(dr[3].ToString()));
                        txtHastaViernes.Text =
Convert.ToString(_metodosGenerales.ConvertirfechaAhora(dr[4].ToString()));
                        txtDesdeViernes.ForeColor = Color.Black;
                        txtHastaViernes.ForeColor = Color.Black;
                        pViernesId = Convert.ToInt32(dr[0]);
                        chkViernes.Checked = true;
                        break;
                    case "Sabado":
                        txtDesdeSabado.Text =
Convert.ToString(_metodosGenerales.ConvertirfechaAhora(dr[3].ToString()));
                        txtHastaSabado.Text =
Convert.ToString(_metodosGenerales.ConvertirfechaAhora(dr[4].ToString()));
                        txtDesdeSabado.ForeColor = Color.Black;
                        txtHastaSabado.ForeColor = Color.Black;
                        pSabadoId = Convert.ToInt32(dr[0]);
                        chkSabado.Checked = true;
                        break;
                }
            }
        }

        private void btnEliminarPlan_Click(object sender, EventArgs e)
        {
            DialogResult result = MessageBox.Show("¿Está seguro de querer eliminar el plan? Si
es así, presione 'Aceptar'",
                "Atención! Eliminación de Plan", MessageBoxButtons.OKCancel);
            if (result == DialogResult.OK)
            {
```

```
                _planes.Plan_ID = Convert.ToInt32(dtgvPlanes.CurrentRow.Cells[0].Value);
                _bussinessPlanes.EliminarPlan(_planes);

                MessageBox.Show("¿Está seguro de querer eliminar el plan?",
                "Atención! Eliminación de Plan", MessageBoxButtons.OKCancel);
            }
        }
        #endregion

    }
}
```