Documentación Técnica

Proyecto Compra de Boletos en Linea

Daniel A. Sequeira Gaitan-Fabian Arredondo Arce

Universidad Internacional San Isidro Labrador

(ISB-09) PROGRAMACIÓN I

Ing. Kenneth L. Aguilar Salas

05 julio 2023

Compra-de-Boletos-en-Linea

Definición General

Desarrollo de venta de boletos automatizada, que inicia mostrando los horarios disponibles y mediante un menú de seleeción le permite al usuario, elegir su horario de conveniencia, o de lo contrario salir. Al elegir horario, muestra proceso de registro de usuario, solicitando datos necesarios los cuales se almacenarán para su posterior impresión estructurados como factura. El sistema valida si el tipo de datos ingresado es acorde al formato establecido, para finalmente muestra todos los clientes que viajan en el horario.

Requerimientos

Proyecto desarrollado mediante aplicación en consola en lenguaje C#, con Visual Studio Community 2019, NetCore3.1, este proyecto fue desarrollado en entorno estudiantil universitario, en su versión original y unica, no forma parte de otros desarrollos al ser propiamente un proyecto universitario.

Procedimiento Instalación y Prueba

Aplicación en consola la cual puede ser ejecutada en visual studio, Net Core 3.1, no requiere instalación adicional al ser una aplicación en consola mas allá de los medios mencionados donde debe ejecutarse.

Arquitectura del proyecto ----------

Cuenta con 3 clases llamadas Program/MAIN, Datos y Validaciones, siendo estas contenedores de código interrelacionado para su funcionamiento, siendo Main el principal a pertir del cual el sistema inicia su ejecución que contiene propiamente código relacionado al menu principal de donde a través de métodos llamados a ejecución realiza las operaciones solicitadas, la calse datos se comporta como almacen de datos de clientes existentes como ingresados por teclado, tarifas, horarios y facturación. Finalmente la clase Validaciones contiene metodos que definen el proceso a seguir de acuerdo a desicios tomadas en pantalla por el usuario, desde definir la disponibilidad de un horario como si se desea realizar una facturación o no. Seguidamente se explicara operación realizada por el código en sus partes.

Menun Principal

Inicialmente mediante la Instancia Validaciones horarios = new Validaciones(); se llama a ejecucuión al metodo Validar Horarios() (almacenado en validaciones) desde Main, el cual funciona de la siguiente forma. Se establece un pequeño arreglo(comparador) de tipo int que contiene números que se comparan con números aleatorios ingresados al arreglo horario1 (o segun corresponda) mediante la clase Random cualquiera = new Random(); que genera números aleatorios en el intérvalo Next(0, 2) siendo en esta caso entre 0 y 2 (0 incluido, 2 excluido) lo enterior usando un bucle for que recorre los indices del arreglo horario y lo rellena de números aleatorios según el intérvalo; posteriormente mediante if se comparan indices de ambos arreglos determinando si horario esta disponible o lleno.

```
public void ValidarHorarios()
    {
        int[] comparador = { 1,0,1 };
        Random cualquiera = new Random();
        bool horarioNodisponible;

        for (int i = 0; i < horarios.horario1.Length; i++)
        {
            horarios.horario1[i] = cualquiera.Next(0, 2);

            if (comparador[1] == horarios.horario1[1] )
            {
                Console.WriteLine("1---09:00h " + " Horario Lleno");
                horarioNodisponible = false;

                if (horarioNodisponible)
                {
                    Console.WriteLine("Este horario está lleno, seleccione otro");
                }
                break;
            else
            {
                Console.WriteLine("1---09:00h " + "Horario Disponible");
                horarioNodisponible = true;

                break;
            }
```

Posteriormente el Menú está estructurado en estructura condicional switch la cual por su sintaxis, evalua opción seleccionada por el usuario y en cada una de estas ejecuta el código que se inserte entre case y break, donde case (1) es un equivalente a if nos dice que el usuario selecionó opción 1 (ingresada por teclado y almacenada en variable opción) y de acuerdo a esta pasa a ejecución case (1)

2 o 3, según corresponda. con un break; que detiene el ciclo, en este caso se llama al metodo registro clientes(previa realización de instancia Datos registro = new Datos();) que solicita datos y rellena un arreglo cliente11.

```
Datos registro = new Datos();//
switch (opcion)
{
    case (1):
     Console.Clear();
                        Console.WriteLine("***Registro de Clientes***");
                        registro.RegistroCliente();
        break;
```

Metodo Registro Clientes

Se utilizó el cilo Do While, para que ejecute el código en el que solicita datos al usuario los cuales se ingresarán por teclado, posteriormente valida si el usuario dejó el espacio solicitado en blanco, escribe cedula o correo en formato incorrecto. validados por if (string.IsNullOrEmpty(cliente11[0]))///Si los datos tipo string son nulos o vacíos en el indice 0 del arreglo cliente11. lance elmensaje de error y mediante el while repite el proceso "Digite su nombre" si el usuario ingresa mal los datos o no los ingresa.

```
    public void RegistroCliente()
   {

        Console.WriteLine("Digite su nombre");
        do
        {
            cliente11[0] = Console.ReadLine();
            if (string.IsNullOrEmpty(cliente11[0]))
            {
                Console.WriteLine("Este campo no puede estar vacio, por favor intente de nue
            }
        } while (string.IsNullOrEmpty(cliente11[0]));
```

Validar cédula

En la opción de solicitud cédula e email llama a ejecución un método de validación para el formato correcto de los mismos. El método ValidarCedula: Compara si la cadena de texto ingresada cedula coincide con el patrón definido por la expresión regular @"^\d{9}$" (Define una variable llamada patron que contiene una expresión regular cadena de 9 digitos) return Regex.IsMatch(cedula, patron);Método IsMatch de la clase Regex valida si la cadena de texto almacenada en la variable cedula coincide con el patrón definido anteriormente.

```
public static bool ValidarCedula(string cedula)
```

```
    {
        string patron = @"^\d{9}$";
        return Regex.IsMatch(cedula, patron);
    }
```

El método ValidarEmail

El metodo que valida si el usuario ingresa de manera correcta su correo, la expresión regular @"^\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*$" (valida si cadena coincide con formato dirección email.) IsMatch(email, patron); (Verifica si la cadena de texto almacenada en la variable email coincide con el patrón definido anteriormente).

```
public static bool ValidarEmail(string email)
    {
        string patron = @"^\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*$"; //

        return Regex.IsMatch(email, patron);
    }
```

EL metodo validar Horarios

Método que establece si horario está lleno o no aleatoriamente, el Arreglo (comparador) que almacena datos tipo entero utilizado para comparaciones. El metodo Random genera numeros aleatorios que rellenan el arreglo horario1 (o en su caso horario2...). luego Bucle For recorre los indices del arreglo horario 1 uno a uno, rellena los indices con numeros aleatorios (Random) donde 0 está incluido y 3 excluido (cualquiera.Next(0, 3);). luego prosigue la comparación de indices [] entre arreglos según coincidencia o no determina la impresión Horario LLeno o Disponible

```
  public void ValidarHorarios()
    {
        int[] comparador = { 1,0,2 };
        Random cualquiera = new Random();
        bool horarioNodisponible = false;

        for (int i = 0; i < horarios.horario1.Length; i++)
        {
            horarios.horario1[i] = cualquiera.Next(0, 3);

            for (int j = 0; j < comparador.Length; j++)
            {
                if (comparador[0] == horarios.horario1[2] && !horarioNodisponible)
                {
                    Console.WriteLine("1---09:00h " + " Horario Lleno");
                    horarioNodisponible = true;
```

```
        }
    }
}
if (!horarioNodisponible)
{
    Console.WriteLine("1---09:00h " + "Horario Disponible");
}
```