Imports System.IO

Imports IPAddressRange

Imports System.Net

Imports System.Numerics

Imports IPNetwork2

Imports System.Threading.Tasks

Imports IPAddressExtensions.IPAddressExtensions

Imports System.Net.IPNetwork

'Imports IPNetwork2

'Imports IPAddressRange

'Imports System.Net

'Imports System.Numerics

'Imports System.Threading.Tasks

'Imports IPAddressExtensions.IPAddressExtensions

Public Class Form1

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click

Dim openFileDialog As New OpenFileDialog()

openFileDialog.Filter = "CSV Files (\*.csv)|\*.csv"

openFileDialog.Multiselect = False

If openFileDialog.ShowDialog() = DialogResult.OK Then

Dim dt As New DataTable()

Dim lines As String() = File.ReadAllLines(openFileDialog.FileName)

If lines.Length > 0 Then

Dim columns As String() = lines(0).Split(","c)

For Each column As String In columns

dt.Columns.Add(column.Trim("""")) ' Elimina las comillas dobles de los nombres de las columnas si las hay

Next

For i As Integer = 1 To lines.Length - 1

Dim rowData As String() = lines(i).Split(","c)

Dim row As DataRow = dt.NewRow()

For j As Integer = 0 To dt.Columns.Count - 1

If j < rowData.Length Then

row(j) = rowData(j).Trim("""") ' Elimina las comillas dobles de los datos de las celdas si las hay

Else

row(j) = DBNull.Value

End If

Next

dt.Rows.Add(row)

Next

If dt IsNot Nothing AndAlso dt.Rows.Count > 0 Then

' Llenar el ComboBox con sedes únicas

ComboBoxSede.Items.Clear()

ComboBoxSede.Items.Add("Todas las sedes")

Dim sedes As List(Of String) = dt.AsEnumerable().Where(Function(r) r("Sede") IsNot DBNull.Value).Select(Function(r) r.Field(Of String)("Sede")).Distinct().OrderBy(Function(s) s).ToList()

For Each sede As String In sedes

If Not String.IsNullOrWhiteSpace(sede) Then ' Agregar esta línea para verificar que la sede no sea una cadena vacía o de espacios en blanco

ComboBoxSede.Items.Add(sede)

End If

Next

ComboBoxSede.SelectedIndex = 0 ' Seleccionar "Todas las sedes" por defecto

End If

Label1.Text = $"Registros cargados: {dt.Rows.Count}"

DataGridView1.DataSource = dt

DataGridView2.DataSource = FiltrarColumnas(dt) ' Mostrar toda la información sin filtrar en DataGridView2

End If

' Añadir esta línea al final del evento Button1\_Click

AplicarFiltro()

End If

End Sub

Private Function FiltrarColumnas(dt As DataTable) As DataTable

Dim dtFiltrado As New DataTable()

Dim columnNames As String() = {"Network", "Comment", "Sede", "País", "VLAN", "Ubicación física dentro de la sede", "IPAM Utilization"}

For Each columnName As String In columnNames

If dt.Columns.Contains(columnName) Then

dtFiltrado.Columns.Add(dt.Columns(columnName).ColumnName)

End If

Next

' Ordena las redes antes de devolver el resultado

dtFiltrado.DefaultView.Sort = "Network ASC"

dtFiltrado = dtFiltrado.DefaultView.ToTable()

dtFiltrado = dt.DefaultView.ToTable(False, dtFiltrado.Columns.Cast(Of DataColumn).Select(Function(c) c.ColumnName).ToArray())

Return dtFiltrado

End Function

Private Sub TextBox1\_KeyDown(sender As Object, e As KeyEventArgs) Handles TextBox1.KeyDown

If e.KeyCode = Keys.Enter Then

Dim dt As DataTable = CType(DataGridView1.DataSource, DataTable)

Dim dv As New DataView(dt)

Dim filterExpression As New List(Of String)

If dt.Columns.Contains("Network") Then filterExpression.Add($"[Network] LIKE '%{TextBox1.Text}%'")

If dt.Columns.Contains("Comment") Then filterExpression.Add($"[Comment] LIKE '%{TextBox1.Text}%'")

If dt.Columns.Contains("Sede") Then filterExpression.Add($"Sede LIKE '%{TextBox1.Text}%'")

If dt.Columns.Contains("País") Then filterExpression.Add($"[País] LIKE '%{TextBox1.Text}%'")

If dt.Columns.Contains("VLAN") Then filterExpression.Add($"[VLAN] LIKE '%{TextBox1.Text}%'")

If dt.Columns.Contains("Ubicación física dentro de la sede") Then filterExpression.Add($"[Ubicación física dentro de la sede] LIKE '%{TextBox1.Text}%'")

If dt.Columns.Contains("IPAM Utilization") Then filterExpression.Add($"[IPAM Utilization] LIKE '%{TextBox1.Text}%'")

dv.RowFilter = String.Join(" OR ", filterExpression)

DataGridView2.DataSource = FiltrarColumnas(dv.ToTable())

Label2.Text = $"Última búsqueda: {TextBox1.Text}"

Label3.Text = $"Registros encontrados: {DataGridView2.Rows.Count - 1}"

Label7.Text = $"Registros encontrados: {DataGridView2.Rows.Count - 1}"

TextBox1.Clear()

e.SuppressKeyPress = True

' Elimina la línea "AplicarFiltro()" desde aquí

TextBox1.Clear()

e.SuppressKeyPress = True

End If

End Sub

Private Sub AplicarFiltro()

If DataGridView1.DataSource IsNot Nothing AndAlso TypeOf DataGridView1.DataSource Is DataTable Then

Dim dt As DataTable = CType(DataGridView1.DataSource, DataTable)

Dim filtro As String = ""

' Filtrar por Sede

If ComboBoxSede.SelectedIndex > 0 Then

filtro = $"Sede = '{ComboBoxSede.SelectedItem}'"

End If

' Filtrar por palabras clave en la columna "Comment"

Dim palabrasClave As String() = {"ToIP", "Interconexión", "Wi-Fi", "WiFi", "Interconexion"}

Dim filtroPalabrasClave As String = String.Join(" AND ", palabrasClave.Select(Function(p) $"Comment NOT LIKE '%{p}%'"))

If String.IsNullOrEmpty(filtro) Then

filtro = filtroPalabrasClave

Else

filtro = filtro & " AND " & filtroPalabrasClave

End If

' Filtrar por contenido de TextBox1

Dim filterExpression As New List(Of String)

If dt.Columns.Contains("Network") Then filterExpression.Add($"[Network] LIKE '%{TextBox1.Text}%'")

If dt.Columns.Contains("Comment") Then filterExpression.Add($"[Comment] LIKE '%{TextBox1.Text}%'")

If dt.Columns.Contains("Sede") Then filterExpression.Add($"Sede LIKE '%{TextBox1.Text}%'")

If dt.Columns.Contains("País") Then filterExpression.Add($"[País] LIKE '%{TextBox1.Text}%'")

If dt.Columns.Contains("VLAN") Then filterExpression.Add($"[VLAN] LIKE '%{TextBox1.Text}%'")

If dt.Columns.Contains("Ubicación física dentro de la sede") Then filterExpression.Add($"[Ubicación física dentro de la sede] LIKE '%{TextBox1.Text}%'")

If dt.Columns.Contains("IPAM Utilization") Then filterExpression.Add($"[IPAM Utilization] LIKE '%{TextBox1.Text}%'")

If filterExpression.Any() Then

filtro = filtro & " AND (" & String.Join(" OR ", filterExpression) & ")"

End If

dt.DefaultView.RowFilter = filtro

DataGridView2.DataSource = FiltrarColumnas(dt.DefaultView.ToTable())

Label2.Text = $"Última búsqueda: {TextBox1.Text}"

Label3.Text = $"Registros encontrados: {DataGridView2.Rows.Count - 1}"

TextBox1.Clear()

End If

End Sub

Private Sub ComboBoxSede\_SelectedIndexChanged(sender As Object, e As EventArgs) Handles ComboBoxSede.SelectedIndexChanged

AplicarFiltro()

End Sub

Private Sub ButtonBuscarRed\_Click(sender As Object, e As EventArgs) Handles ButtonBuscarRed.Click

If ComboBoxSede.SelectedIndex <= 0 Then

MessageBox.Show("Seleccione una sede para buscar redes disponibles.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

Return

End If

ListViewRedesDisponibles.Items.Clear()

ListViewRedesDisponibles.Columns.Clear()

' Añadir columnas al ListView

ListViewRedesDisponibles.Columns.Add("Network")

ListViewRedesDisponibles.Columns.Add("Comment")

ListViewRedesDisponibles.Columns.Add("Sede")

ListViewRedesDisponibles.Columns.Add("País")

ListViewRedesDisponibles.Columns.Add("VLAN")

ListViewRedesDisponibles.Columns.Add("Ubicación física dentro de la sede")

ListViewRedesDisponibles.Columns.Add("IPAM Utilization")

' Configurar el estilo del ListView

ListViewRedesDisponibles.View = View.Details

ListViewRedesDisponibles.GridLines = True

ListViewRedesDisponibles.FullRowSelect = True

' Llenar el ListView con datos filtrados

Dim dt As DataTable = CType(DataGridView1.DataSource, DataTable)

Dim dv As New DataView(dt)

dv.RowFilter = $"Sede = '{ComboBoxSede.SelectedItem}'"

Dim dtFiltrado As DataTable = FiltrarColumnas(dv.ToTable())

For Each row As DataRow In dtFiltrado.Rows

Dim item As New ListViewItem(row.ItemArray.Select(Function(r) r.ToString()).ToArray())

ListViewRedesDisponibles.Items.Add(item)

Next

' Ajustar el ancho de las columnas al contenido

For Each column As ColumnHeader In ListViewRedesDisponibles.Columns

column.Width = -2

Next

' Actualizar Label4

Label4.Text = $"Información cargada para la sede: {ComboBoxSede.SelectedItem}"

Label8.Text = $"Registros encontrados: {ListViewRedesDisponibles.Items.Count}"

' Cambiar a la pestaña TabPage2

TabControl1.SelectedTab = TabPage2

End Sub

Private Sub ListViewRedesDisponibles\_MouseDoubleClick(sender As Object, e As MouseEventArgs) Handles ListViewRedesDisponibles.MouseDoubleClick

If ListViewRedesDisponibles.SelectedItems.Count > 0 Then

Dim selectedNetwork As String = ListViewRedesDisponibles.SelectedItems(0).Text

Clipboard.SetText(selectedNetwork)

End If

End Sub

Private Sub ButtonAbrirCalculadora\_Click\_1(sender As Object, e As EventArgs) Handles ButtonAbrirCalculadora.Click

Dim calculadoraSubredes As New FormSubnetCalculato()

calculadoraSubredes.ShowDialog()

End Sub

'Buscar Ip disponible

Private Async Sub ButtonBuscarIPsDisponibles\_Click(sender As Object, e As EventArgs) Handles ButtonBuscarIPsDisponibles.Click

' ... (El resto del código se mantiene igual)

Dim ipLibre As String = EncontrarEspacioLibreEntreRedes()

MessageBox.Show("IP libre encontrada: " & ipLibre)

' Deshabilita el botón durante la búsqueda

ButtonBuscarIPsDisponibles.Enabled = False

' Muestra el ProgressBar

ProgressBar1.Visible = True

ProgressBar1.Value = 0

' Realiza la búsqueda de IPs disponibles

Await Task.Run(Sub() BuscarIPsDisponibles())

' Oculta el ProgressBar

ProgressBar1.Visible = False

' Habilita el botón después de la búsqueda

ButtonBuscarIPsDisponibles.Enabled = True

End Sub

Private Function ToBigInteger(ipAddress As IPAddress) As BigInteger

Dim bytes As Byte() = ipAddress.GetAddressBytes()

Array.Reverse(bytes)

Return New BigInteger(bytes.Concat({CByte(0)}).ToArray())

End Function

Private Function ToIPAddress(ipBigInteger As BigInteger) As IPAddress

Dim bytes As Byte() = ipBigInteger.ToByteArray()

Array.Resize(bytes, 4)

Array.Reverse(bytes)

Return New IPAddress(bytes)

End Function

Private ipDisponibles As New List(Of String)()

Private Sub BuscarIPsDisponibles()

If String.IsNullOrEmpty(TextBoxIPInicial.Text) OrElse String.IsNullOrEmpty(TextBoxIPFinal.Text) Then

MessageBox.Show("Por favor, ingrese una IP inicial y final válidas antes de buscar IPs disponibles.")

Return

End If

Dim ipInicial As IPAddress = IPAddress.Parse(TextBoxIPInicial.Text)

Dim ipFinal As IPAddress = IPAddress.Parse(TextBoxIPFinal.Text)

Dim ipsDisponibles As New List(Of IPAddress)

For i As BigInteger = ToBigInteger(ipInicial) To ToBigInteger(ipFinal)

ipsDisponibles.Add(ToIPAddress(i))

Next

ProgressBar1.Invoke(Sub() ProgressBar1.Maximum = ipsDisponibles.Count)

ProgressBar1.Invoke(Sub() ProgressBar1.Value = 0)

ListViewIPsDisponibles.Items.Clear()

Dim totalItems As Integer = ipsDisponibles.Count

For index As Integer = 0 To ipsDisponibles.Count - 1

Dim ip As IPAddress = ipsDisponibles(index)

If Not IPEncontradaEnDataGridView(ip) Then

Dim row As DataGridViewRow = DataGridView2.Rows.Cast(Of DataGridViewRow)().FirstOrDefault(Function(r) r.Cells("Network").Value IsNot Nothing AndAlso r.Cells("Network").Value.ToString().Split("/")(0) = ip.ToString())

If row IsNot Nothing Then

Dim comentario As String = row.Cells("Comment").Value.ToString()

If Not ContienePalabrasClave(comentario) Then

ListViewIPsDisponibles.Invoke(Sub() ListViewIPsDisponibles.Items.Add(ip.ToString()))

End If

End If

End If

' Actualiza el ProgressBar y LabelProgressBarPercentage dentro del bucle

Dim progressPercentage As Integer = CInt(((index + 1) / totalItems) \* 100)

If progressPercentage > ProgressBar1.Maximum Then

progressPercentage = ProgressBar1.Maximum

End If

ProgressBar1.Invoke(Sub() ProgressBar1.Value = progressPercentage)

LabelProgressBarPercentage.Invoke(Sub() LabelProgressBarPercentage.Text = $"{progressPercentage}%")

Next

' Muestra la cantidad de IPs disponibles en el LabelIPsDisponibles

LabelIPsDisponibles.Text = "IPs disponibles: " & ListViewIPsDisponibles.Items.Count.ToString()

End Sub

Private Function IPEncontradaEnDataGridView(ip As String) As Boolean

For Each row As DataGridViewRow In DataGridView2.Rows

If Not row.IsNewRow AndAlso row.Cells("Network").Value.ToString().Split("/")(0) = ip Then

Return True

End If

Next

Return False

End Function

Private Function CompareIPs(ip1 As IPAddress, ip2 As IPAddress) As Integer

Dim ip1BigInt As BigInteger = ToBigInteger(ip1)

Dim ip2BigInt As BigInteger = ToBigInteger(ip2)

Return ip1BigInt.CompareTo(ip2BigInt)

End Function

Private Function EncontrarEspacioLibreEntreRedes() As String

Dim ipLibre As String = ""

' Filtrar las filas según las palabras clave en la columna "Comment"

Dim rowsFiltradas = DataGridView2.Rows.Cast(Of DataGridViewRow)().Where(Function(row) Not row.IsNewRow AndAlso Not ContienePalabrasClave(row.Cells("Comment").Value.ToString())).ToList()

' Ordenar las filas filtradas por IP

rowsFiltradas.Sort(Function(row1, row2) CompareIPs(IPAddress.Parse(row1.Cells("Network").Value.ToString().Split("/")(0)), IPAddress.Parse(row2.Cells("Network").Value.ToString().Split("/")(0))))

' Buscar espacio libre entre redes

For i As Integer = 0 To rowsFiltradas.Count - 2

Dim cidrActual As String = rowsFiltradas(i).Cells("Network").Value.ToString()

Dim cidrSiguiente As String = rowsFiltradas(i + 1).Cells("Network").Value.ToString()

Dim ipNetworkActual As IPNetwork = IPNetwork.Parse(cidrActual)

Dim ipNetworkSiguiente As IPNetwork = IPNetwork.Parse(cidrSiguiente)

' Comprobar si hay espacio libre entre las dos redes

If Integer.Parse(ipNetworkActual.LastUsable.ToString()) + 1 <> Integer.Parse(ipNetworkSiguiente.FirstUsable.ToString()) Then

ipLibre = Integer.Parse(ipNetworkActual.LastUsable.ToString()) + 1

Exit For

End If

Next

If String.IsNullOrEmpty(ipLibre) Then

' Si no se encontró espacio libre, devolver la siguiente IP después de la última en la lista

Dim ultimaIp As String = rowsFiltradas(rowsFiltradas.Count - 1).Cells("Network").Value.ToString()

Dim ultimaIpNetwork As IPNetwork = IPNetwork.Parse(ultimaIp)

ipLibre = Integer.Parse(ultimaIpNetwork.LastUsable.ToString()) + 1

End If

Return ipLibre

End Function

' Diseño de la litviweipdisponible

Private Sub Form1\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

' Configurar ProgressBar1

ProgressBar1.Minimum = 0

ProgressBar1.Maximum = 100

ProgressBar1.Value = 0

ProgressBar1.Visible = False

' Configurar ListViewIPsDisponibles

ListViewIPsDisponibles.View = View.Details

ListViewIPsDisponibles.GridLines = True

ListViewIPsDisponibles.FullRowSelect = True

ListViewIPsDisponibles.MultiSelect = False

' Agregar columnas al ListViewIPsDisponibles

ListViewIPsDisponibles.Columns.Add("IP", 150)

ListViewIPsDisponibles.Columns.Add("CIDR", 100)

' Puedes agregar más columnas si es necesario

' Cambiar el estilo de vista a Details

ListViewIPsDisponibles.View = View.Details

' Agregar una columna

Dim columnHeader As New ColumnHeader()

columnHeader.Text = "IPs Disponibles"

columnHeader.Width = 150

ListViewIPsDisponibles.Columns.Add(columnHeader)

' Agrega la función de autocompletar al ComboBox

Dim autoComplete As New AutoCompleteStringCollection()

autoComplete.AddRange(ComboBoxSede.Items.Cast(Of String).ToArray())

ComboBoxSede.AutoCompleteMode = AutoCompleteMode.SuggestAppend

ComboBoxSede.AutoCompleteSource = AutoCompleteSource.CustomSource

ComboBoxSede.AutoCompleteCustomSource = autoComplete

ComboBoxSede.DropDownStyle = ComboBoxStyle.DropDownList

End Sub

' boton de filtrar IPs

Private Sub ButtonFiltrarIPs\_Click(sender As Object, e As EventArgs) Handles ButtonFiltrarIPs.Click

ProgressBar1.Visible = True

FiltrarIPs(TextBoxFiltro.Text)

ProgressBar1.Visible = False

End Sub

Private Sub FiltrarIPs(filtro As String)

ListViewIPsDisponibles.Items.Clear()

For Each ip As String In ipDisponibles

If ip.StartsWith(filtro) Then

ListViewIPsDisponibles.Items.Add(ip)

End If

Next

End Sub

Private Sub ListViewIPsDisponibles\_DoubleClick(sender As Object, e As EventArgs) Handles ListViewIPsDisponibles.DoubleClick

If ListViewIPsDisponibles.SelectedItems.Count > 0 Then

Dim ipSeleccionada As String = ListViewIPsDisponibles.SelectedItems(0).Text

Dim ipDisponible As String = EncontrarEspacioLibreEntreRedes()

' Aquí puedes mostrar la IP disponible en un cuadro de mensaje o en otro control, como un TextBox

MessageBox.Show($"IP seleccionada: {ipSeleccionada}{Environment.NewLine}IP disponible: {ipDisponible}")

End If

End Sub

Private Function IPEncontradaEnDataGridView(ip As IPAddress) As Boolean

For Each row As DataGridViewRow In DataGridView2.Rows

If Not row.IsNewRow Then

Dim ipActual As IPAddress = IPAddress.Parse(row.Cells("Network").Value.ToString().Split("/")(0))

Dim cidr As Integer = Integer.Parse(row.Cells("Network").Value.ToString().Split("/")(1))

' Convertir el objeto IPAddress a cadena antes de pasarlo a la función Parse()

Dim ipActualString As String = ipActual.ToString()

Dim ipNetwork As IPNetwork = IPNetwork.Parse(ipActualString, CByte(cidr))

If ipNetwork.Contains(ip) Then

Return True

End If

End If

Next

Return False

End Function

Private Function ContienePalabrasClave(comentario As String) As Boolean

If comentario Is Nothing Then

Return False

End If

Dim palabrasClave As String() = {"ToIP", "Interconexión", "Wi-Fi", "WiFi", "Interconexion"}

For Each palabra As String In palabrasClave

If comentario.Contains(palabra) Then

Return True

End If

Next

Return False

End Function

Private Sub ListViewRedesDisponibles\_SelectedIndexChanged(sender As Object, e As EventArgs) Handles ListViewRedesDisponibles.SelectedIndexChanged

If ListViewRedesDisponibles.SelectedItems.Count > 0 Then

Dim selectedItem As ListViewItem = ListViewRedesDisponibles.SelectedItems(0)

Dim cidr As String = $"{selectedItem.SubItems(0).Text}/{selectedItem.SubItems(1).Text}"

Dim ipNetwork As IPNetwork = IPNetwork.Parse(cidr)

TextBoxIPInicial.Text = ipNetwork.FirstUsable.ToString()

TextBoxIPFinal.Text = ipNetwork.LastUsable.ToString()

End If

End Sub

Private Sub ButtonReset\_Click(sender As Object, e As EventArgs) Handles ButtonReset.Click

ProgressBar1.Value = 0

LabelProgressBarPercentage.Text = "0%"

LabelIPsDisponibles.Text = "IPs disponibles:"

ListViewIPsDisponibles.Items.Clear()

ButtonBuscarIPsDisponibles.Enabled = True

End Sub

Private Function IPAddressNotInDatabase(ip As IPAddress, rows As List(Of DataGridViewRow)) As Boolean

For Each row As DataGridViewRow In rows

Dim existingIP As IPAddress = IPAddress.Parse(row.Cells("Network").Value.ToString().Split("/")(0))

If ip.Equals(existingIP) Then

Return False

End If

Next

Return True

End Function

End Class