Imports BAL

Imports DAL

Imports System

Imports System.Reflection

Imports System.Windows.Forms

Imports System.ComponentModel

Imports System.Collections.Generic

Imports System.Drawing

Public Class RescheduleDailyDisplay

    Private screenOffset As Point

    Private SourceClinician As String

    Private sourcerowindex As Integer

    Private destgridSourceClinician As String

    Private DestSourceStartTime As String

    Private tempstudent As String

    Private SourceStartTime As String

    Private trigger As Integer

    Private activeGrid As Integer = 0

    Private storedDestination As New List(Of Storage)

    Private storeoriginalTime As New List(Of OriginalTime)

    Dim destorigrowindex As Integer

    Dim destorigcolindex As Integer

REM setup the Source GridView Control

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

                Dim HomeDisplaydate As Date

        MonthCalendar1.SetDate(HomeDisplay.MonthCalendar1.SelectionStart())

        HomeDisplaydate = MonthCalendar1.SelectionStart

        HomeScreen(HomeDisplaydate)

        Dim newColumnWidthSource As Integer

        Dim newColumnWidthDestination As Integer

        Dim changeColumnSize As New StoreGridViewColumnWidth

        newColumnWidthSource = changeColumnSize.ReturnColumnWidth(2, 1)

        TextBox1.Text = newColumnWidthSource.ToString

        For x = 1 To DataGridView1.Columns.Count - 1

            DataGridView1.Columns(x).Width = newColumnWidthSource

        Next

        newColumnWidthDestination = changeColumnSize.ReturnColumnWidth(2, 2)

        TextBox1.Text = newColumnWidthSource.ToString

        For x = 1 To DataGridView4.Columns.Count - 1

            DataGridView4.Columns(x).Width = newColumnWidthDestination

        Next

        TextBox2.Text = newColumnWidthDestination.ToString

        With Me.DataGridView1

            .SelectionMode = DataGridViewSelectionMode.RowHeaderSelect

            .MultiSelect = False

        End With

        With Me.DataGridView4

            .SelectionMode = DataGridViewSelectionMode.RowHeaderSelect

            .MultiSelect = False

        End With

    End Sub

    Public Function HomeScreen(ByVal Startupdate As Date)

        Try

            Dim clinicianinfo As New Clinicians

            Dim convertApostrophe As New nameOperation

            Dim dsclinicians As New DataSet

REM inactive clinicans are clinician who have left the institute and  will not be coming back in the future.

REM Active clinicians are those clinicians who are currently working  or are not working for an extended peiod of time.

REM Return all active Clinicians. Passing TRUE value looks for all  active clinicians and returns all active clinicians.

REM If one were to choose false it would return all inactive  clinicians

            dsclinicians = clinicianinfo.GetClinicianInfo(True)

            Dim dtClinicians As DataTable = dsclinicians.Tables("Clinician")

            Dim row As DataRow

REM Populate each column headers of the DataGridView control with all the Clinicians that have returned as active in the Source and Destination GridViews.

For Each row In dtClinicians.Rows

                clinicianLastName = row("LastName")

                clinicianFirstName = row("FirstName")

Dim clinicianFullName As String = clinicianLastName.Trim & ", " & clinicianFirstName.Trim

            clinicianFullName = convertApostrophe.executeName(clinicianFullName, 0)

            DataGridView1.Columns.Add(clinicianFullName, clinicianFullName)

            DataGridView4.Columns.Add(clinicianFullName, clinicianFullName)

            Next

            DisplaySetup(Startupdate)

            DataGridView1.SelectionMode = DataGridViewSelectionMode.CellSelect

            DataGridView1.AllowUserToResizeColumns = True

            DataGridView4.AllowUserToResizeColumns = True

            Dim z As Integer = 0

            Dim x As Integer = 0

            z = DataGridView4.ColumnCount - 1

            For x = 0 To z

          DataGridView4.Columns(x).SortMode = DataGridViewColumnSortMode.NotSortable

            Next

        Catch ex As Exception

            Throw ex

        End Try

        Return Nothing

    End Function

 REM remove all the columns to reset the Source sceen Display

    Public Function RemoveColumns()

        Me.DataGridView1.Columns.Clear()

        Return Nothing

    End Function

REM Setup the column that list all 25 time intervals Between 7:30 AM to 6:00 PM

REM Dataset ds will store all the students in their respective time slots.

REM Dataset ds will store all the students in their respective time slots.

REM Get Selected Calendar value from the CalendarControl

REM The line below is necessary in order to attain other attributes associated with each student such as classroom, and Campus information

REM Return each clincian

REM Iterate through each Clinician listed in each header column of the DataGrid

REM Check to see if the current Clinician matches the Clinician name Listed in the Header Column of the DataGrid control

REM iterate through all the rows/labeled time intervals

REM Store current time calue from the first column of the GridView Control

REM Get the first student name at this time interval that is stored in the dataset

REM Check to see if clinician is off at the particular time interval by testing for an empty string.

REM Otherwise convert the students name to an thier identification number and place him in the respective cell in the Gridview

REM get studentid

REM return additional information of the student

REM Check Proposed(not rescheduled) hours that are hour 1 or 2

REM Check rescheduled hours that are hour 1 or 2

REM Determine the Color which corresponds to the students Status Reschedule/Transfer,Proposed,

REM Also Determine the students Attendance,No Show/Absent then set the respective cell color

REM Also check if there will be a meeting, or Testing

REM reset the studentid and location

REM Check for the first two hours of the current student. If the firsttwo hours are found the flag it inthevariable called start hour

REM Check to see if the first two hours are triggered. If so then  color the cell white.

REM check to see if this is a different student from the last  iteration.

REM Otherwise the location will show up in every cell/timeslot that  corresponds to the student in the current iteration

rather than once.

REM Also, check to see if student name appears on every hour that is scheduled.

REM Store the student name so the current student in the iteration only appears once.

    Public Sub DisplaySetup(ByVal CurrentDate As Date)

REM Returns all Students that are scheduled for the currently selected date,

REM and places them into thier respective time slots in the  DataGridView Control.

        Dim convertname As INameConversion = New StudentNameconversion

        Try

            Dim Student As String = String.Empty

            Dim display As IDisplaySetup = New DisplayModule

            Dim Stat As New Schedule

REM Setup the column that list all 25 time intervals between 7:30 AM to 6:00 PM

            DisplayTemplate()

            Dim ds As New DataSet

REM The GridView control will mirror the dataset.

REM Dataset ds  will store all the students in their respective  time slots.

            ds = display.mainDisplaySchedule(CurrentDate)

            Dim dt As DataTable = ds.Tables("ScheduleDisplayScreen")

            Dim ds2 As New DataSet

            Dim ds3 As New DataSet

            Dim ds5 As New DataSet

            Dim studentid As String

            Dim nextname As String = String.Empty

            Dim location As String = String.Empty

REM Get Selected Calendar value from the CalendarControl

            Dim startdate As DateTime = MonthCalendar1.SelectionStart

            Dim FinalDate As DateTime = MonthCalendar1.SelectionStart

            Dim markout As String = Nothing

REM The line below is necessary in order to attain other attributes associated with each student such as classroom, and Campus information

            ds2 = Stat.GetSchedule(startdate, FinalDate)

            Dim dt2 As DataTable = ds2.Tables("MainSchedule")

            Dim xname As Integer = 0

            Dim b As Integer

            Dim Status As String = String.Empty

            Dim Subject As String = String.Empty

            Dim present As String = String.Empty

            Dim Clinician As String = String.Empty

            Dim ClinicianHeader As String = String.Empty

            Dim countid As String = Nothing = String.Empty

REM Return each clincian

            For Each rw In dt.Rows

                Clinician = rw("Clinician")

                Dim numberofColumns As Integer = DataGridView1.ColumnCount

                Dim headercount As Integer = 0

REM Iterate through each Clinician listed in each header column of the

                For headercount = 1 To numberofColumns

                    ClinicianHeader = DataGridView1.Columns(headercount).HeaderText

REM Check to see if the current Clinician matches the Clinician name  Listed in the Header Column of the DataGrid control

                    If ClinicianHeader = Clinician Then

                        Dim a As Integer = 0

                        Dim timein As String = String.Empty

                        Dim t1 As DateTime

                        Dim query As String = String.Empty

REM iterate through all the rows/labeled time intervals

                        For a = 1 To 24

                            DataGridView4.Rows(a).Cells(headercount).Style.BackColor = Color.AntiqueWhite

REM Store current time calue from the first column of the GridView  Control

                            timein = DataGridView1.Rows(a).Cells(0).Value

                            t1 = Convert.ToDateTime(timein.Trim).ToShortTimeString

REM Get the first student name at this time interval that is stored in the dataset

                            Student = rw(a).ToString

REM Check to see if clinician is off at the particular time interval by testing for an empty string.

REM Otherwise convert the students name to an thier identification number and place him in the respective cell in the Gridview

                            If Student = "           " & "OUT" Then

                                DataGridView1.Rows(a).Cells(headercount).Style.BackColor = Color.Green

                                markout = markout + 1

                                If markout > 1 Then

                                    Student = [String].Empty

                                End If

                            ElseIf Student <> String.Empty Then

                                studentid = convertname.convertToId(Student)

                                Dim timestamp As String = String.Empty

                                timestamp = Convert.ToDateTime("1900-01-01 " & timein)

                                Dim time1 As DateTime

                                Dim time2 As DateTime

                                query = "Studentid='" & studentid.Trim & "' AND Timein <='" & timestamp & "' AND TimeOut >='" & timestamp & "'"

REM return additional information of the student

                                Dim foundrow() As DataRow = dt2.Select(query)

                                b = 0

                                Do While b <= foundrow.Length - 1

                                    Status = foundrow(b)("status")

                                    countid = foundrow(b)("Count")

                                    present = foundrow(b)("Attendance")

                                    time1 = foundrow(b)("TimeIn")

                                    time2 = foundrow(b)("TimeOut")

                                    ds5 = Stat.GetClassroomData(countid.Trim)

                                    Dim dt5 As DataTable = ds5.Tables("Classroom")

                                    Dim subjectrow As DataRow

                                    For Each subjectrow In dt5.Rows

                                        Subject = subjectrow("Subject")

                                        location = subjectrow("Campus")

                                    Next

'Check for the first two hours of the current student. If the first two hours are found the flag it inthevariable called start hour

                                    Dim starthr As String = String.Empty

                                    Dim gethr As String = String.Empty

'Check Proposed(not rescheduled) hours that are hour 1 or 2

                                    Dim query2 = "Studentid='" & studentid.Trim & "' AND Timein <='" & timestamp & "' AND TimeOut >='" & timestamp & "'"

'Check rescheduled hours that are hour 1 or 2

                                    Dim e As Integer = 0

REM Determine the Color which corresponds to the students Status Reschedule/Transfer, Proposed.

REM Also, Determine the students Attendance,No Show/Absent then set the respective cell color

REM Also check if there will be a meeting, or Testing

    If present.Trim = "Absent" And (timestamp >= time1 Or timestamp <= time2) Then

                                        DataGridView1.Rows(a).Cells(headercount).Style.BackColor = Color.Red

               ElseIf present.Trim = "Completed" Then

                                        DataGridView1.Rows(a).Cells(headercount).Style.BackColor = Color.Blue

                                    ElseIf Status.Trim = "Transfer" Then

                                        DataGridView1.Rows(a).Cells(headercount).Style.BackColor = Color.Azure

                                    ElseIf Status.Trim = "Proposed" And Subject.Trim = "Testing" Then

                                        DataGridView1.Rows(a).Cells(headercount).Style.BackColor = Color.Gray

                                    ElseIf Status.Trim = "Proposed" And Subject.Trim = "Meeting" Then

                                        DataGridView1.Rows(a).Cells(headercount).Style.BackColor = Color.Orange

                                    ElseIf present.Trim = "Proposed" And Subject.Trim = "Start" Then

                                        DataGridView1.Rows(a).Cells(headercount).Style.BackColor = Color.White

                                    ElseIf present.Trim = "Proposed" And (timestamp >= time1 Or timestamp <= time2) Then

                                        DataGridView1.Rows(a).Cells(headercount).Style.BackColor = Color.Yellow

                                    End If

'Check to see if the first two hours are triggered. If so then color the cell white.

                                    b = b + 1

                                Loop

                            ElseIf Student = [String].Empty Then

REM reset the studentid and location to an empty string

                                studentid = [String].Empty

                                location = [String].Empty

                                DataGridView1.Rows(a).Cells(headercount).Value = [String].Empty

                                DataGridView1.Rows(a).Cells(headercount).Style.BackColor = Color.AntiqueWhite

                            End If

REM check to see if this is a different student from the last iteration.

REM Otherwise the location will show up in every cell/timeslot that corresponds to the student in the current iteration rather than once.

                            If nextname <> Student Or xname < b Then

                                xname = xname + 1

REM Find out the Location and abreviate it, so it will be concatenated to the name

                                If location.Trim = "NorthWest" Or location.Trim = "Northwest" Then

                                    location = "NW="

                                ElseIf location.Trim = "Main" Then

                                    location = "M="

                                Else

                                    location = [String].Empty

                                End If

REM Concatenate the Location with the student name

                                DataGridView1.Rows(a).Cells(headercount).Value = location & Student

                                DataGridView1.Columns(headercount).DefaultCellStyle.Font = New Font("Times NewRoman", 8, FontStyle.Regular)

REM store the student name so the current student in the iteration  only appears once.

                                nextname = Student

                            Else

                                xname = 0

                            End If

                        Next

                        Exit For

                    Else

                    End If

                Next

            Next

REM Update Rescheduling, attendance display

            DataGridView1.AllowDrop = True

            DataGridView4.AllowDrop = True

            Dim x As Integer = 0

            Dim y As Integer = 0

            y = DataGridView1.ColumnCount - 1

            For x = 0 To y

                DataGridView1.Columns(x).SortMode = DataGridViewColumnSortMode.NotSortable

            Next

        Catch ex As Exception

            Throw ex

        End Try

    End Sub

    Public Sub DisplayTemplate()

REM This is a Template which Populates the all 26 rows in the Firstcolumn of the DataGrid Control with all the time values Between 7:30 AM to 6:00 PM.

REM set the Font size and Color

        DataGridView1.Columns(0).DefaultCellStyle.Font = New Font("Times NewRoman", 10, FontStyle.Regular)

        DataGridView4.Columns(0).DefaultCellStyle.Font = New Font("Times NewRoman", 10, FontStyle.Regular)

        Dim i As Integer

        Dim timestamp As Array = {"Hour", "7:30 AM", "8:00 AM", "8:30 AM", "9:00 AM", "9:30 AM", "10:00 AM", "10:30 AM", "11:00 AM", "11:30 AM", "12:00 PM", "12:30 PM",

                                  "1:00 PM", "1:30 PM", "2:00 PM", "2:30 PM", "3:00 PM", "3:30 PM", "4:00 PM", "4:30 PM", "5:00 PM", "5:30 PM", "6:00 PM",

                                 "6:30 PM", "7:00 PM", "7:30 PM"}

REM Add each row into the DataGrid Control through interation

        For i = 0 To 25

            Dim dgvRow As New DataGridViewRow

            Dim dgvCell As DataGridViewCell

            dgvCell = New DataGridViewTextBoxCell()

            dgvCell.Value = timestamp(i)

            dgvRow.Cells.Add(dgvCell)

            DataGridView1.Rows.Add(dgvRow)

            DataGridView1.Rows(i).Cells(0).Style.BackColor = Color.WhiteSmoke

        Next

REM Add each row into the DataGrid Control through interation

        For i = 0 To 25

            Dim dgvRowDestination As New DataGridViewRow

            Dim dgvCellDestination As DataGridViewCell

            dgvCellDestination = New DataGridViewTextBoxCell()

            dgvCellDestination.Value = timestamp(i)

            dgvRowDestination.Cells.Add(dgvCellDestination)

            DataGridView4.Rows.Add(dgvRowDestination)

            DataGridView4.Rows(i).Cells(0).Style.BackColor = Color.Beige

        Next

        DataGridView4.Rows(0).ReadOnly = True

        DataGridView4.Rows(25).ReadOnly = True

    End Sub

    Public Sub ClearDestinationScreenentries()

REM Wipe out all Generic Collections with a single day

        Dim p As Storage

        Dim q As Integer = 0

        Dim maxcount As Integer = storedDestination.Count - 1

        For q = maxcount To 0 Step -1

            p = CType(storedDestination(q), Storage)

            storedDestination.Remove(p)

        Next

    End Sub

REM Remove every Row in the GridView Control

    Public Sub Removerows(ByVal Schedule\_Date As Date)

        Dim w As Integer = DataGridView1.Rows.Count

        If w > 0 Then

            For b = 0 To DataGridView1.RowCount - 1

                DataGridView1.Rows.RemoveAt(0)

            Next

        End If

        Dim x As Integer = DataGridView4.Rows.Count

        If x > 0 Then

            For b = 0 To DataGridView4.RowCount - 1

                DataGridView4.Rows.RemoveAt(0)

            Next

        End If

        DisplaySetup(Schedule\_Date)

    End Sub

    Private Sub ClinicianManagerToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ClinicianManagerToolStripMenuItem.Click

        ClinicianConsole.Show()

        ClinicianConsole.Focus()

    End Sub

    Private Sub FormStyleToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles FormStyleToolStripMenuItem.Click

        OfficeSchedulePrintOut.Show()

    End Sub

    Private Sub CloseToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles CloseToolStripMenuItem.Click

        Me.Close()

    End Sub

    Private Sub FormStyleToolStripMenuItem1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)

        OfficeSchedulePrintOut.Show()

    End Sub

    Private Sub ToolStripButton2\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ToolStripButton2.Click

        ClinicianConsole.Show()

        ClinicianConsole.Focus()

    End Sub

    Private Sub ToolStripButton1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ToolStripButton1.Click

        StudentManager.Show()

    End Sub

    Private Sub StudentCalendarToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)

        StudentCalendar.Show()

        StudentCalendar.Focus()

    End Sub

    Private Sub StudentManagerToolStripMenuItem1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles StudentManagerToolStripMenuItem1.Click

        StudentManager.Show()

    End Sub

Private Sub StudentReportListingToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles StudentReportListingToolStripMenuItem.Click

    End Sub

    Private Sub StudentManagerToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles StudentManagerToolStripMenuItem.Click

        signin.Show()

        signin.Focus()

    End Sub

    Private Sub FormFormatToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles FormFormatToolStripMenuItem.Click

        StudentCalendar.Show()

    End Sub

    Private Sub Button1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)

    End Sub

    Public Sub InitialSelectionTransfer(ByVal colindex As Integer, ByVal rowindex As Integer, ByVal totaltime As Integer)

        Dim x As Integer = 0

        Do While x < totaltime

            x = x + 30

            If DataGridView4.Rows(rowindex).Cells(colindex).Style.BackColor = Color.Red Then

                DataGridView4.Rows(rowindex).Cells(colindex).Style.BackColor = Color.Yellow

            End If

            rowindex = rowindex + 1

        Loop

    End Sub

'Trigger for setting a clinician to bee scheduled out or in. Set cell to the appropriate color

    Private Sub DataGridView1\_CellMouseClick(ByVal sender As Object, ByVal e As System.Windows.Forms.DataGridViewCellMouseEventArgs) Handles DataGridView1.CellMouseClick

        Try

            Dim setOffDays As IscheduleClinicianoff = New ScanClinicianoffDays

            If DataGridView1.CurrentCell.Style.BackColor = Color.Green Then

                DataGridView1.CurrentCell.Style.BackColor = Color.AntiqueWhite

                DataGridView1.CurrentCell.Value = String.Empty

                setOffDays.scan()

                DataGridView1.ClearSelection()

            ElseIf DataGridView1.CurrentCell.Style.BackColor = Color.AntiqueWhite Then

                DataGridView1.CurrentCell.Style.BackColor = Color.Green

                setOffDays.scan()

                DataGridView1.ClearSelection()

            Else

                CaptureSourceGrid()

            End If

        Catch ex As Exception

        End Try

    End Sub

 'Get trigger in Source GridView

    Public Sub CaptureSourceGrid()

        If DataGridView1.CurrentCell.ColumnIndex < 1 Then

            Exit Sub

        End If

        If DataGridView1.CurrentCell.Style.BackColor = Color.LightGray Then

            DataGridView1.ClearSelection()

            trigger = -1

            Exit Sub

        End If

        trigger = 0

        activeGrid = 0

    End Sub

REM Determine the cell that was clicked in the source Gridview control

    Public Sub CaptureCell()

        Dim student As String = String.Empty

        Dim selfconflict As Boolean = False

        If DataGridView4.CurrentCell.ColumnIndex < 1 Then

            Exit Sub

        End If

        If DataGridView4.CurrentRow.Index = 0 Then

            Exit Sub

        ElseIf DataGridView4.CurrentRow.Index > 23 Then

            DataGridView1.ClearSelection()

            DataGridView4.ClearSelection()

            student = [String].Empty

            tempstudent = [String].Empty

            destorigcolindex = 0

            destorigrowindex = 0

            Exit Sub

        End If

        If trigger > -1 Then

            Dim DestinationStartTime As String = String.Empty

            Dim DestinationEndTime As String = String.Empty

            Dim colindex As Integer = 0

            Dim rowindex As Integer = 0

            Dim destcolindex As Integer = 0

            Dim timeslot As Integer

            Dim DestinationClinician As String

'Determine which grid is being triggered

            If activeGrid = 1 And trigger <= 1 Then

                If trigger = 1 Then

                    If Not tempstudent = String.Empty Then

                        DestinationStartTime = DataGridView4.CurrentRow.Cells(0).Value

                        Dim ClinicianColumnNumber As Integer = DataGridView4.CurrentCell.ColumnIndex

                        DestinationClinician = DataGridView4.Columns(ClinicianColumnNumber).HeaderText

                        MoveTimeSlot(tempstudent.Trim, destorigcolindex, destorigrowindex, DestinationClinician, DestinationStartTime)

                    Else

                        trigger = 0

                        DataGridView1.ClearSelection()

                        DataGridView4.ClearSelection()

                        student = [String].Empty

                        tempstudent = [String].Empty

                        destorigcolindex = 0

                        destorigrowindex = 0

                        Exit Sub

                    End If

                    trigger = 0

                    DataGridView1.ClearSelection()

                    DataGridView4.ClearSelection()

                    student = [String].Empty

                    tempstudent = [String].Empty

                    destorigcolindex = 0

                    destorigrowindex = 0

                Else

                    student = DataGridView4.CurrentCell.Value

                    If student <> String.Empty Then

                        tempstudent = student

                        destorigcolindex = DataGridView4.CurrentCell.ColumnIndex

                        destorigrowindex = DataGridView4.CurrentCell.RowIndex

                        DestSourceStartTime = DataGridView4.CurrentRow.Cells(0).Value.ToString()

                        destgridSourceClinician = DataGridView4.Columns(destorigcolindex).HeaderText

                        trigger = 1

                    Else

                        trigger = 0

                        activeGrid = 1

                        student = [String].Empty

                        destorigrowindex = 0

                        destorigcolindex = 0

                        Exit Sub

                    End If

                End If

                Exit Sub

'Catch the click in the source gridView

            ElseIf activeGrid = 0 And trigger = 0 Then

                student = DataGridView1.CurrentCell.Value

                rowindex = DataGridView1.CurrentCell.RowIndex

                colindex = DataGridView1.CurrentCell.ColumnIndex

                If student = [String].Empty Then

                    DataGridView1.ClearSelection()

                    DataGridView4.ClearSelection()

                    student = [String].Empty

                    Exit Sub

                End If

                destcolindex = DataGridView4.CurrentCell.ColumnIndex

                SourceClinician = DataGridView1.Columns(colindex).HeaderText

                SourceStartTime = DataGridView1.CurrentRow.Cells(0).Value.ToString()

                DestinationStartTime = DataGridView4.CurrentRow.Cells(0).Value.ToString()

                DestinationClinician = DataGridView4.Columns(destcolindex).HeaderText

                Dim Endtime As DateTime = Convert.ToDateTime(DestinationStartTime)

                timeslot = CalculateTimeSlot(student, SourceStartTime)

                selfconflict = selfConflictinDestinationGrid(timeslot)

                If selfconflict = True Then

                Else

StudentInitialClick(student.Trim, SourceStartTime.Trim, DestinationClinician.Trim, DestinationStartTime, DestinationClinician, rowindex, colindex)

                End If

                trigger = 0

                activeGrid = 1

                DataGridView1.ClearSelection()

                DataGridView4.ClearSelection()

                student = [String].Empty

                destorigrowindex = 0

                destorigcolindex = 0

            End If

        End If

    End Sub

'Collection of astudents data within the first cell which was clicked by the  user and store the values

    Public Sub StudentInitialClick(ByVal Studentname As String, ByVal SourceStarttime As String, ByVal schClinician As String, ByVal DestinationTimein As String, ByVal DestinedClinician As String, ByVal originalRow As Integer, originalCol As Integer)

        Dim convertName As INameConversion = New StudentNameconversion

        Dim studentfullname As String = String.Empty

        Dim GetData As New Schedule

        Dim Transactionid As String = String.Empty

        Dim studentinfo As New DataSet

        Dim time1, time2 As DateTime

        Dim timeconversion As DateTime

        Dim appointment As DateTime

        Dim starttime As String = String.Empty

        Dim endtime As String = String.Empty

        Dim totaltime As Integer = 0

        Dim DestinationTimeout As String = String.Empty

        Dim OriginaltimeIn As String = String.Empty

        Dim SourceEndtime As String = String.Empty

        Dim studentid As String = String.Empty

        Dim RemoveCampus() As String

        Dim rowindex, colindex As Integer

        RemoveCampus = Studentname.Split("=")

        studentfullname = RemoveCampus(1)

        Dim rowindex2 As Integer

        Dim colindex2 As Integer

        appointment = MonthCalendar1.SelectionStart

        rowindex2 = DataGridView1.CurrentCell.RowIndex

        colindex2 = DataGridView1.CurrentCell.ColumnIndex

        studentid = convertName.convertToId(studentfullname)

        studentinfo = GetData.ReturnStudentScheduleinfo(studentid.Trim, appointment, appointment)

        Dim dtStudentinfo As DataTable = studentinfo.Tables("MainSchedule")

        Dim t As String = "1900-01-01 " & SourceStarttime

        timeconversion = Convert.ToDateTime(t.Trim)

        Dim query As String = "TimeIn='" & timeconversion & "' AND Studentid='" & studentid.Trim & "'"

        Dim State As String = Nothing

        Dim findstudentinfo() As DataRow = dtStudentinfo.Select(query)

        Dim studentKey As Integer

        For z = 0 To findstudentinfo.Length - 1

            studentKey = findstudentinfo(z)("count")

            time1 = findstudentinfo(z)("Timein")

            starttime = time1.ToString("h:mm tt")

            time2 = findstudentinfo(z)("TimeOut")

            endtime = time2.ToString("h:mm tt")

            Transactionid = findstudentinfo(z)("Count")

            State = findstudentinfo(z)("Status")

        Next

        Dim subject As String = String.Empty

        Dim Location As String = String.Empty

        Dim newConflict As Boolean = False

        Dim ds As New DataSet

        ds = GetData.GetClassroomData(Transactionid.Trim)

        Dim dt As DataTable = ds.Tables("Classroom")

        Dim dr As DataRow

        For Each dr In dt.Rows

            Location = dr.Item("Campus")

            subject = dr.Item("Subject")

        Next

        Dim ts As TimeSpan

        Dim time3, time4 As DateTime

        Dim totalhours As Integer = 0

        ts = (time2 - time1)

        totaltime = ts.TotalMinutes

        totalhours = ts.Hours

        time3 = Convert.ToDateTime(DestinationTimein)

        time4 = time3.AddMinutes(totaltime)

        SourceEndtime = time2.ToString("h:mm tt")

        DestinationTimeout = time4.ToString("h:mm tt")

        If DestinationTimein = "7:00 PM" And totaltime > 30 Then

            Exit Sub

        ElseIf DestinationTimein = "7:30 PM" Then

            Exit Sub

        End If

        storeoriginalTime.Add(New OriginalTime(studentKey, studentfullname.Trim, schClinician.Trim, DestinationTimein, DestinationTimeout, originalRow, originalCol, totaltime, State, Location))

REM if Source Student move then Check the original location in the destination grid to see if there is someone in

REM conflict at the same time. If so then repaint the conflicting person in the second grid back to his origial color

        rowindex = DataGridView4.CurrentCell.RowIndex

        colindex = DataGridView4.CurrentCell.ColumnIndex

        InitialSelectionTransfer(colindex2, rowindex2, totaltime)

        Dim foundconflict As Boolean

        foundconflict = ConflictSearch(studentKey, Studentname.Trim, SourceStarttime.Trim, SourceEndtime.Trim, SourceClinician.Trim, DestinationTimein, DestinationTimeout, DestinedClinician, totaltime, Location.Trim, subject.Trim, State.Trim, rowindex, colindex)

    End Sub

REM Process the new selected location within the destination gridview

REM Check for conflicts with other students

    Public Function MoveTimeSlot(ByVal Student As String, ByVal x As Integer, ByVal y As Integer, ByVal destinationClinician As String, ByVal destinationStartTime As String)

        Try

            Dim DestinationEndTime As String = String.Empty

            Dim timeslot As Integer = 0

            Dim sourcestartTime As String = String.Empty

            Dim colindex As Integer

            Dim rowindex As Integer

            Dim location As String = String.Empty

            Dim subject As String = String.Empty

            Dim SourceEndtime As String = String.Empty

            Dim State As String = String.Empty

            Dim p As Storage

            Dim i As Integer = 0

            Dim newConflict As Boolean = False

            Dim originalrow, originalcol As Integer

            Dim originalClinician As String = String.Empty

            Dim Conflict As Boolean = False

            Dim studentKey As Integer

            Dim yindex, xindex As Integer

            yindex = DataGridView4.CurrentCell.RowIndex

            xindex = DataGridView4.CurrentCell.ColumnIndex

‘Determine the location of the student

            For i = 0 To storedDestination.Count - 1

                p = CType(storedDestination(i), Storage)

                If p.Name = Student And x = p.Col And y = p.Row Then

                    studentKey = p.key

                    timeslot = p.totaltime

                    sourcestartTime = p.DestinationTimeIn

                    SourceEndtime = p.DestinationTimeout

                    location = p.Location

                    State = p.Status

                    subject = p.Subject

                    colindex = p.Col

                    rowindex = p.Row

                    originalrow = p.OrigRow

                    originalcol = p.OrigCol

                    originalClinician = p.S\_Clinician

                    Conflict = p.Conflict

                    Dim Endtime As DateTime = Convert.ToDateTime(destinationStartTime)

                    DestinationEndTime = (Endtime.AddMinutes(timeslot)).ToString("h:mm tt")

                    If rowindex = 25 Or rowindex = 24 And timeslot > 30 Then

                        Return Nothing

                        Exit Function

                    End If

                    Dim selfconflict As Boolean = False

                    RemoveStudent(studentKey, Student.Trim, timeslot, originalrow, originalcol, originalClinician.Trim, colindex, rowindex)

                    selfconflict = ConflictSearch(studentKey, Student.Trim, sourcestartTime.Trim, SourceEndtime.Trim, originalClinician.Trim, destinationStartTime, DestinationEndTime, destinationClinician, timeslot, location.Trim, subject.Trim, State.Trim, originalcol, originalrow)

                    If selfconflict = True Then

                        storedDestination.Add(New Storage(studentKey, Student, originalClinician, p.S\_TimeIn, p.S\_TimeOut, destinationClinician, p.S\_TimeIn, p.S\_TimeOut, rowindex, colindex, originalrow, originalcol, timeslot, Conflict, location, subject, State))

                        AllocateDestinationstates(Student, timeslot, selfconflict, rowindex, colindex)

                    Else

                    End If

                    Exit For

                End If

            Next

        Catch ex As Exception

        End Try

        Return Nothing

    End Function

'Check for a conflict with another user within the same time slot. Check inside the generic collection

    Public Function ConflictSearch(ByVal studentKey As Integer, ByVal Studentname As String, ByVal SourceStarttime As String, ByVal SourceEndTime As String, ByVal schClinician As String, ByVal DestinationTimein As String, ByVal DestinationTimeout As String, ByVal DestinedClinician As String, ByVal totaltime As String, ByVal location As String, ByVal subject As String, ByVal State As String, ByVal originalcolindex As Integer, ByVal originalrowindex As Integer) As Boolean

        Try

            Dim intervals As IEvaluateDateTimeIntervals = New datetimeIntervalConversion

            Dim storedestinationtable As New ScheduleConfig

            Dim selfConflict As Boolean = False

            Dim foundConflict As New List(Of AutoSelectConflicts)

            Dim timeintervals As New ArrayList

            Dim starttime As String = String.Empty

            Dim endtime As String = String.Empty

            Dim StartDate As String

            Dim appointment As Date

            Dim convertname As New Schedule

            Dim studentfullname As String = String.Empty

            Dim Previousclinician As String = String.Empty

            Dim destinationdataset As New DataSet

            Dim rowindex, colindex As Integer

            Dim p As AutoSelectConflicts

            rowindex = DataGridView4.CurrentCell.RowIndex

            colindex = DataGridView4.CurrentCell.ColumnIndex

            appointment = MonthCalendar1.SelectionStart

            StartDate = appointment.ToString("M/dd/yyyy")

            Dim RemoveCampus() As String

            RemoveCampus = Studentname.Split("=")

            studentfullname = RemoveCampus(1)

REM Search for a Specific Student and return their data

            timeintervals = intervals.timeIntervals(DestinationTimein.Trim, DestinationTimeout.Trim)

            Dim scanForConflict As IReschedule = New ManuallyReschedulingTime

            foundConflict = scanForConflict.ClinicianAvailability(studentfullname, DestinedClinician, StartDate, StartDate, DestinationTimein, DestinationTimeout, timeintervals)

'Validate for conflicts for manual scheduling

            Dim person As Integer

            person = foundConflict.Count

            Dim conflictType As String = String.Empty

            Dim conflictDate As String = String.Empty

            Dim conflictTimeIn As String = String.Empty

            Dim conflictTimeOut As String = String.Empty

            Dim conflictwithstudent As String = String.Empty

            Dim conflictingStudent As String = String.Empty

            Dim tutor As String = String.Empty

            'There is 0 conflicts then store values

            Dim conflictrowindex As Integer = 0

            Dim conflictcolindex As Integer = 0

            conflictrowindex = DataGridView4.CurrentCell.RowIndex

            conflictcolindex = DataGridView4.CurrentCell.ColumnIndex

            If person > 0 Then

                For i = 0 To person - 1

                    p = CType(foundConflict(i), AutoSelectConflicts)

                    tutor = p.D\_Clinician

                    conflictType = p.ConflictType

                    conflictDate = p.ScheduledDate

                    conflictTimeIn = p.DestinationTimeIn

                    conflictTimeOut = p.DestinationTimeout

                    conflictwithstudent = p.ConflictType

                    conflictingStudent = p.Name

                    If conflictType = "student" Then

  If Not (DataGridView1.Rows(conflictrowindex).Cells(conflictcolindex).Style.BackColor = Color.LightGray) And conflictingStudent.Trim <> studentfullname.Trim Then

                            MsgBox("There is a schedule conflict with another student at this selected date and time!")

                            selfConflict = selfConflictinDestinationGrid(totaltime)

                            If selfConflict = False Then

                                storedDestination.Add(New Storage(studentKey, Studentname, schClinician, SourceStarttime, SourceEndTime, DestinedClinician, DestinationTimein, DestinationTimeout, rowindex, colindex, originalrowindex, originalcolindex, totaltime, False, location, subject, State))

                                AllocateSourcestates(Studentname, totaltime, True)

                                Return False

                                Exit Function

                            Else

                                Return True

                                Exit Function

                            End If

                        Else

                            storedDestination.Add(New Storage(studentKey, Studentname, schClinician, SourceStarttime, SourceEndTime, DestinedClinician, DestinationTimein, DestinationTimeout, rowindex, colindex, originalrowindex, originalcolindex, totaltime, False, location, subject, State))

                            AllocateSourcestates(Studentname, totaltime, False)

                            Return False

                            Exit Function

                        End If

                    ElseIf conflictType = "self" And DataGridView1.Rows(conflictrowindex).Cells(conflictcolindex).Style.BackColor <> Color.Gray And conflictingStudent.Trim = studentfullname.Trim Then

                        MsgBox(studentfullname & " is already scheduled on " & p.ScheduledDate & " from " & conflictTimeIn & " to " & conflictTimeOut & ".")

                        selfConflict = selfConflictinDestinationGrid(totaltime)

                        If selfConflict = False Then

                            storedDestination.Add(New Storage(studentKey, Studentname, schClinician, SourceStarttime, SourceEndTime, DestinedClinician, DestinationTimein, DestinationTimeout, rowindex, colindex, originalrowindex, originalcolindex, totaltime, False, location, subject, State))

                            AllocateSourcestates(Studentname, totaltime, True)

                            ChkConflictingTimeSlots(Studentname, totaltime)

                            Return False

                            Exit Function

                        Else

                            Return True

                            Exit Function

                        End If

                    ElseIf conflictType = "clinician" Then

                        MsgBox(DestinedClinician & " is scheduled to be off " & p.ScheduledDate & " from " & conflictTimeIn & " to " & conflictTimeOut & ".")

                        selfConflict = selfConflictinDestinationGrid(totaltime)

                        If selfConflict = False Then

                            storedDestination.Add(New Storage(studentKey, Studentname, schClinician, SourceStarttime, SourceEndTime, DestinedClinician, DestinationTimein, DestinationTimeout, rowindex, colindex, originalrowindex, originalcolindex, totaltime, True, location, subject, State))

                            AllocateSourcestates(Studentname, totaltime, True)

                            Return False

                            Exit Function

                        Else

                            Return True

                            Exit Function

                        End If

                    End If

                Next

            Else

                selfConflict = selfConflictinDestinationGrid(totaltime)

                If selfConflict = False Then

                    storedDestination.Add(New Storage(studentKey, Studentname, schClinician, SourceStarttime, SourceEndTime, DestinedClinician, DestinationTimein, DestinationTimeout, rowindex, colindex, originalrowindex, originalcolindex, totaltime, False, location, subject, State))

                    AllocateSourcestates(Studentname, totaltime, False)

                    Return False

                Else

                    Return True

                End If

            End If

        Catch ex As Exception

        End Try

        Return Nothing

    End Function

REM Check to see if student is in conflict with himeself by scanning the color of the destined cell

REM This is used for coloring the conflicting location

  Public Function selfConflictinDestinationGrid(ByVal timeslot As Integer) As Boolean

        Dim Conflict As Boolean = False

        Dim rowindex As Integer

        Dim colindex As Integer

        Dim x As Integer

        rowindex = DataGridView4.CurrentCell.RowIndex

        colindex = DataGridView4.CurrentCell.ColumnIndex

        Do While x < timeslot

            x = x + 30

            If DataGridView4.Rows(rowindex).Cells(colindex).Style.BackColor = Color.Red Then

                Conflict = True

                Exit Do

            ElseIf DataGridView4.Rows(rowindex).Cells(colindex).Style.BackColor = Color.Yellow Then

                Conflict = True

                Exit Do

            Else

                Conflict = False

            End If

            rowindex = rowindex + 1

        Loop

        Return Conflict

    End Function

REM Check to see if student is in conflict with another user by scanning the color of the destined cell for yellow or gray

REM This is used for coloring the conflicting location.

    Public Function ChkConflictingTimeSlots(ByVal Studentname As String, ByVal timeslot As Integer) As Boolean

        Dim student1 As String

        Dim conflict As Boolean

        Dim rowindex As Integer

        Dim colindex As Integer

        Dim x As Integer

        rowindex = DataGridView4.CurrentCell.RowIndex

        colindex = DataGridView4.CurrentCell.ColumnIndex

        student1 = DataGridView1.Rows(rowindex).Cells(colindex).Value

        Do While x < timeslot

            If (DataGridView1.Rows(rowindex).Cells(colindex).Style.BackColor = Color.LightGray Or DataGridView1.Rows(rowindex).Cells(colindex).Style.BackColor = Color.AntiqueWhite) Then

                For y = 0 To DataGridView4.ColumnCount - 1

                    If Studentname <> DataGridView4.CurrentRow.Cells(y).Value Or (y = colindex) Then

                        conflict = False

                        DataGridView4.Rows(rowindex).Cells(colindex).Style.BackColor = Color.Yellow

                    Else

                        DataGridView4.Rows(rowindex).Cells(colindex).Style.BackColor = Color.Red

                        conflict = True

                        Exit For

                    End If

                    If DataGridView1.Rows(rowindex).Cells(y).Value = Studentname And DataGridView1.Rows(rowindex).Cells(y).Style.BackColor = Color.Yellow Then

                        Do While x < timeslot

                            DataGridView4.Rows(rowindex).Cells(colindex).Style.BackColor = Color.Red

                            conflict = True

                            x = x + 30

                            rowindex = rowindex + 1

                        Loop

                        Return conflict

                        Exit Function

                    End If

                Next

            Else

            End If

            x = x + 30

            rowindex = rowindex + 1

        Loop

        Return conflict

    End Function

'Check to see if there is a conflict with another student in the destination GridView at a particular timeslot

    Public Function AllocateDestinationstates(ByVal Student As String, ByVal totaltime As Integer, ByVal Conflict As Boolean, ByVal destrow As Integer, ByVal destcol As Integer)

        Dim desrowindex As Integer

        DataGridView4.Rows(destrow).Cells(destcol).Value = Student

        For x = 30 To totaltime Step 30

            If desrowindex = 25 And totaltime > 30 Then

                Return Nothing

                Exit Function

            End If

            If Conflict = True Then

                DataGridView4.Rows(destrow).Cells(destcol).Style.BackColor = Color.Red

            ElseIf Conflict = False Then

                DataGridView4.Rows(destrow).Cells(destcol).Style.BackColor = Color.Yellow

            End If

            destrow = destrow + 1

        Next

        Return Nothing

    End Function

'Determine the exact number of hours the student is scheduled for.

Public Function CalculateTimeSlot(ByVal Studentname As String, ByVal SourceStarttime As String) As Integer

        Dim convertName As INameConversion = New StudentNameconversion

        Dim studentfullname As String = String.Empty

        Dim GetData As New Schedule

        Dim Transactionid As String = String.Empty

        Dim studentinfo As New DataSet

        Dim time1, time2 As DateTime

        Dim timeconversion As DateTime

        Dim appointment As DateTime

        Dim starttime As String = String.Empty

        Dim endtime As String = String.Empty

        Dim totaltime As Integer = 0

        Dim DestinationTimeout As String = String.Empty

        Dim OriginaltimeIn As String = String.Empty

        Dim SourceEndtime As String = String.Empty

        Dim studentid As String = String.Empty

        Dim RemoveCampus() As String

        If Studentname = [String].Empty Then

            Return True

            Exit Function

        End If

        RemoveCampus = Studentname.Split("=")

        studentfullname = RemoveCampus(1)

        Dim rowindex2 As Integer

        Dim colindex2 As Integer

        appointment = MonthCalendar1.SelectionStart

        rowindex2 = DataGridView1.CurrentCell.RowIndex

        colindex2 = DataGridView1.CurrentCell.ColumnIndex

        studentid = convertName.convertToId(studentfullname.Trim)

    studentinfo = GetData.ReturnStudentScheduleinfo(studentid.Trim, appointment, appointment)

        Dim dtStudentinfo As DataTable = studentinfo.Tables("MainSchedule")

        Dim t As String = "1900-01-01 " & SourceStarttime

        timeconversion = Convert.ToDateTime(t.Trim)

Dim query As String = "TimeIn='" & timeconversion & "' AND Studentid='" & studentid.Trim & "'"

        Dim State As String = Nothing

        Dim findstudentinfo() As DataRow = dtStudentinfo.Select(query)

        If findstudentinfo.Length = 0 Then

            Return True

            Exit Function

        End If

        For z = 0 To findstudentinfo.Length - 1

            time1 = findstudentinfo(z)("Timein")

            starttime = time1.ToString("h:mm tt")

            time2 = findstudentinfo(z)("TimeOut")

            endtime = time2.ToString("h:mm tt")

            Transactionid = findstudentinfo(z)("Count")

            State = findstudentinfo(z)("Status")

        Next

        Dim subject As String = Nothing

        Dim Location As String = Nothing

        Dim newConflict As Boolean = False

        Dim ds As New DataSet

        ds = GetData.GetClassroomData(Transactionid.Trim)

        Dim dt As DataTable = ds.Tables("Classroom")

        Dim dr As DataRow

        For Each dr In dt.Rows

            Location = dr.Item("Campus")

            subject = dr.Item("Subject")

        Next

        Dim ts As TimeSpan

        Dim totalhours As Integer = 0

        ts = (time2 - time1)

        totaltime = ts.TotalMinutes

        totalhours = ts.Hours

        Return totaltime

    End Function

    Private Sub DataGridView4\_CellMouseClick(ByVal sender As Object, ByVal e As System.Windows.Forms.DataGridViewCellMouseEventArgs) Handles DataGridView4.CellMouseClick

        Try

            CaptureCell()

        Catch ex As Exception

        End Try

    End Sub

'Trigger for resetting a student to their original location

    Private Sub DataGridView4\_CellMouseDoubleClick(ByVal sender As Object, ByVal e As System.Windows.Forms.DataGridViewCellMouseEventArgs) Handles DataGridView4.CellMouseDoubleClick

        If DataGridView4.CurrentCell.ColumnIndex < 1 Then

            Exit Sub

        End If

        Dim conflict As Boolean

        conflict = ResetStudent()

        trigger = 0

        DataGridView1.ClearSelection()

        DataGridView4.ClearSelection()

        tempstudent = [String].Empty

        destorigcolindex = 0

        destorigrowindex = 0

    End Sub

'Checks the source gridView if there is a conflict with another student at a  particular time slot. If so, then color the student ‘red’ in the  destination gridView.

    Public Function AllocateSourcestates(ByVal Student As String, ByVal totaltime As Integer, ByVal Conflict As Boolean)

        Dim rowindex, colindex, desrowindex, descolindex As Integer

        rowindex = DataGridView1.CurrentCell.RowIndex

        desrowindex = DataGridView4.CurrentCell.RowIndex

        descolindex = DataGridView4.CurrentCell.ColumnIndex

        colindex = DataGridView1.CurrentCell.ColumnIndex

        DataGridView4.Rows(desrowindex).Cells(descolindex).Value = Student

        For x = 30 To totaltime Step 30

            If desrowindex = 25 And totaltime > 30 Then

                Return Nothing

                Exit Function

            End If

            If Conflict = True Then

                DataGridView4.Rows(desrowindex).Cells(descolindex).Style.BackColor = Color.Red

                DataGridView1.Rows(rowindex).Cells(colindex).Style.BackColor = Color.LightGray

            ElseIf Conflict = False Then

                DataGridView4.Rows(desrowindex).Cells(descolindex).Style.BackColor = Color.Yellow

                DataGridView1.Rows(rowindex).Cells(colindex).Style.BackColor = Color.LightGray

            End If

            desrowindex = desrowindex + 1

            rowindex = rowindex + 1

        Next

        Return Nothing

    End Function

'Clears any trace of the student from their cell. This happens when the user moves the student to a new location within the destination  gridview.

    Public Function RemoveStudent(ByVal studentKey As Integer, ByVal Student As String, ByVal timeslot As Integer, ByVal originaly As Integer, ByVal originalx As Integer, ByVal SourceClinician As String, ByVal destorigcolindex As Integer, ByVal destorigrowindex As Integer)

        Dim p As Storage

        Dim i As Integer = 0

        For i = 0 To storedDestination.Count - 1

            p = CType(storedDestination(i), Storage)

            If p.Name = Student And p.OrigRow = originaly And p.OrigCol = originalx And studentKey = p.key Then

                storedDestination.Remove(p)

                For x = 30 To timeslot Step 30

                    DataGridView4.Rows(destorigrowindex).Cells(destorigcolindex).Value = [String].Empty

                    DataGridView4.Rows(destorigrowindex).Cells(destorigcolindex).Style.BackColor = Color.AntiqueWhite

                    destorigrowindex = destorigrowindex + 1

                Next

                Exit For

            End If

        Next

        Return Nothing

    End Function

'Displays the user to their original state within the source gridview.

    Public Sub RestoreStudentState(ByVal totaltime As Integer, ByVal state As String, ByVal rowindex As Integer, ByVal colindex As Integer)

        For x = 30 To totaltime Step 30

            If state = "Proposed" Then

                DataGridView1.Rows(rowindex).Cells(colindex).Style.BackColor = Color.Yellow

                rowindex = rowindex + 1

            ElseIf state = "Transfer" Then

                DataGridView1.Rows(rowindex).Cells(colindex).Style.BackColor = Color.Azure

                rowindex = rowindex + 1

            End If

        Next

        DataGridView4.ClearSelection()

    End Sub

‘Returns a student back to their original position within the source gridview.

REM this is done when the user double clicks on the students name in the destination gridview.

    Public Function ResetStudent() As Boolean

        Dim studentName As String = String.Empty

        Dim studentfullname As String = String.Empty

        Dim p As Storage

        Dim q As OriginalTime

        Dim timeslot As Integer

        Dim SourceEndTime As String

        Dim DestinationEndTime As String = String.Empty

        Dim DestinationStartTime As String = String.Empty

        Dim studentKey As Integer

        Dim location As String = String.Empty

        Dim resetsourcegridx As Integer

        Dim resetsourcegridy As Integer

        Dim State As String = String.Empty

        Dim totaltime As Integer

        Dim x, y As Integer

        Dim destinationClinician As String = String.Empty

        Dim studentid As String = String.Empty

        Dim RemoveCampus() As String

        studentName = DataGridView4.CurrentCell.Value

        If studentName = String.Empty Then

            Return True

            Exit Function

        End If

        RemoveCampus = studentName.Split("=")

        studentfullname = RemoveCampus(1)

        x = DataGridView4.CurrentCell.ColumnIndex

        y = DataGridView4.CurrentCell.RowIndex

        destinationClinician = DataGridView4.Columns(x).HeaderText

        DestinationStartTime = DataGridView4.CurrentRow.Cells(0).Value.ToString()

        Dim originalclinician As String

        For i = 0 To storedDestination.Count - 1

            p = CType(storedDestination(i), Storage)

            If p.Name = studentName.Trim And p.DestinationTimeIn = DestinationStartTime.Trim And p.D\_Clinician = destinationClinician.Trim Then

                studentKey = p.key

                For j = 0 To storeoriginalTime.Count - 1

                    q = CType(storeoriginalTime(j), OriginalTime)

                    If q.Name = studentfullname.Trim And q.key = studentKey Then

                        SourceStartTime = q.S\_TimeIn

                        SourceEndTime = q.S\_TimeOut

                        location = q.Location

                        State = q.Status

                        totaltime = q.totaltime

                        resetsourcegridy = q.OrigRow

                        resetsourcegridx = q.OrigCol

                        originalclinician = q.S\_Clinician

                        If y > 23 And timeslot > 30 Then

                            Return False

                            Exit Function

                        End If

                        Dim b As Integer

                        If DataGridView4.Rows(y).Cells(x).Value <> studentName Then

                            For a = 30 To totaltime Step 30

                                DataGridView4.Rows(resetsourcegridy + b).Cells(resetsourcegridx).Style.BackColor = Color.Red

                                b = b + 1

                            Next

                            MsgBox("There is a student that is proposed to be scheduled at this time.")

                        End If

                        Dim Endtime As DateTime = Convert.ToDateTime(DestinationStartTime)

                        DestinationEndTime = (Endtime.AddMinutes(totaltime)).ToString("h:mm tt")

                        RestoreStudentState(totaltime, State.Trim, resetsourcegridy, resetsourcegridx)

                        RemoveStudent(studentKey, studentName, totaltime, p.OrigRow, p.OrigCol, destinationClinician.Trim, x, y)

                        Return False

                        Exit Function

                    End If

                Next

            End If

        Next

        Return False

    End Function

    Private Sub DataGridView1\_Scroll(ByVal sender As Object, ByVal e As System.Windows.Forms.ScrollEventArgs) Handles DataGridView1.Scroll

        sourcerowindex = DataGridView1.FirstDisplayedScrollingRowIndex

    End Sub

    Private Sub MonthCalendar1\_DateSelected(ByVal sender As Object, ByVal e As System.Windows.Forms.DateRangeEventArgs) Handles MonthCalendar1.DateSelected

        Dim Schedule\_Date As Date

        ClearDestinationScreenentries()

        Schedule\_Date = MonthCalendar1.SelectionStart

        Removerows(Schedule\_Date)

    End Sub

    Private Sub Button3\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click

        REM Reset the entire Display

        Dim DonnotSave As Boolean = False

        Dim Schedule\_Date As Date

        DonnotSave = updateStudent()

        If DonnotSave = False Then

            ClearDestinationScreenentries()

            Schedule\_Date = MonthCalendar1.SelectionStart

            Removerows(Schedule\_Date)

        End If

        HomeDisplay.Removerows(Schedule\_Date)

    End Sub

'When student is updated it checks for any conflicts on the destionation  grid. If there are any, it does not allow the changes to take place unless the conflict is resolved by the user.

    Public Function ChkConflicts() As Boolean

        Dim y, x, q As Integer

        Dim p As Storage

        Dim maxcount As Integer = storedDestination.Count - 1

        For q = maxcount To 0 Step -1

            p = CType(storedDestination(q), Storage)

            y = p.Row

            x = p.Col

            If DataGridView4.Rows(y).Cells(x).Style.BackColor = Color.Red Then

                Return True

                Exit Function

            End If

        Next

        HomeDisplay.Refresh()

        Return False

    End Function

'Save all changes to the data store

    Public Function updateStudent() As Boolean

        Dim convertStudentName As INameConversion = New StudentNameconversion

        Dim convertClincianName As INameConversion = New ClinicianNameConversion

        Dim UpdateTimes As New CommitChanges

        Dim p As Storage

        Dim q As Integer = 0

        Dim convertname As New Schedule

        Dim studentfullname As String = [String].Empty

        Dim studentid As String = [String].Empty

        Dim StartTime As String = [String].Empty

        Dim EndTime As String = [String].Empty

        Dim schClinician As String = [String].Empty

        Dim studentkey As String = String.Empty

        Dim StartDate As String = [String].Empty

        Dim Location As String = [String].Empty

        Dim Subject As String = [String].Empty

        Dim oldTimein As String = [String].Empty

        Dim OldTimeout As String = [String].Empty

        Dim studentname As String = [String].Empty

        Dim clinicianid As String = [String].Empty

        Dim SourceDate As DateTime

        Dim conflict As Boolean = False

        Dim timeslot As Integer

        Dim origrowindex As Integer

        Dim origcolindex As Integer

        Dim x As Integer

        Dim y As Integer

        Dim update As Boolean = False

'Check for any conflict on the destination gridview

        update = ChkConflicts()

        If update = True Then

            MsgBox("You cannot save the following layout until some scheduling conflicts are resolved!")

            Return True

            Exit Function

        End If

        SourceDate = MonthCalendar1.SelectionStart

        StartDate = Convert.ToString(SourceDate)

        Dim prevClinician As String = String.Empty

        Dim maxcount As Integer = storedDestination.Count - 1

        For i = maxcount To 0 Step -1

            p = CType(storedDestination(i), Storage)

            studentkey = p.key

            studentname = p.Name

            schClinician = p.D\_Clinician

            prevClinician = p.S\_Clinician

            StartTime = p.DestinationTimeIn

            EndTime = p.DestinationTimeout

            oldTimein = p.S\_TimeIn

            OldTimeout = p.S\_TimeOut

            timeslot = p.totaltime

            Location = p.Location

            Subject = p.Subject

            y = p.Row

            x = p.Col

            origrowindex = p.OrigRow

            origcolindex = p.OrigCol

            Dim RemoveCampus() As String

            RemoveCampus = studentname.Split("=")

            studentfullname = RemoveCampus(1)

            studentid = convertStudentName.convertToId(studentfullname)

            clinicianid = convertClincianName.convertToId(schClinician)

            UpdateTimes.EditDailySchedule(studentkey, studentid.Trim, clinicianid.Trim, schClinician.Trim, StartDate.Trim, StartTime.Trim, EndTime.Trim, oldTimein.Trim, OldTimeout.Trim, Location, Subject)

REM Remove student from the destination display and from memory

            Dim p1 As Storage

            For j = 0 To storedDestination.Count - 1

                p1 = CType(storedDestination(j), Storage)

                If p1.Name = studentname And p1.OrigRow = origrowindex And p1.OrigCol = origcolindex And p1.S\_Clinician = prevClinician.Trim Then

                    storedDestination.Remove(p1)

                    For z = 30 To timeslot Step 30

                    DataGridView4.Rows(y).Cells(x).Value = [String].Empty

                    DataGridView4.Rows(y).Cells(x).Style.BackColor = Color.AntiqueWhite

                    Next

                End If

                Exit For

            Next

        Next

        Return False

    End Function

    Private Sub StudentScheduleToolStripMenuItem1\_Click(sender As System.Object, e As System.EventArgs) Handles StudentScheduleToolStripMenuItem1.Click

        StudentCalendar.Show()

    End Sub

    Private Sub DailyScheduleToolStripMenuItem1\_Click(sender As System.Object, e As System.EventArgs) Handles DailyScheduleToolStripMenuItem1.Click

        OfficeSchedulePrintOut.Show()

    End Sub

    Public Sub PrintOfficeSchedule()

        OfficeSchedulePrintOut.Show()

        OfficeSchedulePrintOut.DateTimePicker1.Value = MonthCalendar1.SelectionStart()

        OfficeSchedulePrintOut.DateTimePicker2.Value = MonthCalendar1.SelectionStart()

        OfficeSchedulePrintOut.Focus()

    End Sub

'Adjust column width in the source gridview control

    Private Sub Button1\_Click\_1(sender As System.Object, e As System.EventArgs) Handles Button1.Click

        If TextBox1.Text = String.Empty Then

            Exit Sub

        End If

        Dim newwidth As Integer

        newwidth = Convert.ToInt16(TextBox1.Text)

        If newwidth < 10 Then

            For x = 1 To DataGridView1.Columns.Count - 1

                DataGridView1.Columns(x).Width = 80

            Next

            newwidth = 80

        ElseIf newwidth > 150 Then

            For x = 1 To DataGridView1.Columns.Count - 1

                DataGridView1.Columns(x).Width = 80

            Next

            newwidth = 80

        Else

            For x = 1 To DataGridView1.Columns.Count - 1

                DataGridView1.Columns(x).Width = newwidth

            Next

        End If

        Dim changeColumnSize As New StoreGridViewColumnWidth

        newwidth = changeColumnSize.SaveColumnWidth(2, 1, newwidth)

        TextBox1.Text = newwidth.ToString

    End Sub

'Adjust column width in the destination gridview control

    Private Sub Button4\_Click(sender As System.Object, e As System.EventArgs) Handles Button4.Click

        If TextBox2.Text = String.Empty Then

            Exit Sub

        End If

        Dim newWidth As Integer

        newWidth = Convert.ToInt16(TextBox2.Text)

        If newWidth < 10 Then

            For x = 1 To DataGridView4.Columns.Count - 1

                DataGridView4.Columns(x).Width = 100

            Next

            newWidth = 100

            Exit Sub

        ElseIf newWidth > 150 Then

            For x = 1 To DataGridView4.Columns.Count - 1

                DataGridView4.Columns(x).Width = 100

            Next

            newWidth = 100

        Else

            For x = 1 To DataGridView4.Columns.Count - 1

                DataGridView4.Columns(x).Width = newWidth

            Next

        End If

        Dim changeColumnSize As New StoreGridViewColumnWidth

        newWidth = changeColumnSize.SaveColumnWidth(2, 2, newWidth)

        TextBox2.Text = newWidth.ToString

    End Sub

    Private Sub ToolStripButton3\_Click(sender As System.Object, e As System.EventArgs) Handles ToolStripButton3.Click

        signin.Show()

        signin.Focus()

    End Sub

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

        ClinicianCalendar.Show()

        ClinicianCalendar.Focus()

    End Sub

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

        ExportSchedule.Show()

        ExportSchedule.Focus()

    End Sub

End Class

'Used for scheduling a clinician out or in whenever a partcular time slot is clicked.

Interface IscheduleClinicianoff

    Function scan()

End Interface

'Remove the schedule for the entire day

'Scan entire column for the color green

'Schedule clinician at the specific time slot and save the values in the data store

Public Class ScanClinicianoffDays

    Inherits updateClinicianOutSchedule

    Implements IscheduleClinicianoff

    Public Function scan() Implements IscheduleClinicianoff.scan

        Dim cliniciannameConversion As INameConversion = New ClinicianNameConversion

        Dim scheduleClinician As IscheduleClinican = New alterSchedule

        Dim clinicianId As String = String.Empty

        Dim StartTime As String = String.Empty

        Dim Endtime As String = String.Empty

        Dim todaysdate As Date

        todaysdate = RescheduleDailyDisplay.MonthCalendar1.SelectionStart

        x = RescheduleDailyDisplay.DataGridView1.CurrentCell.ColumnIndex

        y = RescheduleDailyDisplay.DataGridView1.CurrentCell.RowIndex

        clinicianId = cliniciannameConversion.convertToId(RescheduleDailyDisplay.DataGridView4.Columns(x).HeaderText)

        Dim clinicianRow As Integer

        'Remove the schedule for the entire day

        scheduleClinician.RemoveDate(clinicianId, todaysdate)

        For clinicianRow = 0 To 25

            If RescheduleDailyDisplay.DataGridView1.Rows(clinicianRow).Cells(x).Style.BackColor = Color.Green Then

                StartTime = RescheduleDailyDisplay.DataGridView1.Rows(clinicianRow).Cells(0).Value

                Do While clinicianRow < 26

'Scan entire column for the color green

'Schedule clinician at the specific time slot and save the values in the data store

If Not (RescheduleDailyDisplay.DataGridView1.Rows(clinicianRow).Cells(x).Style.BackColor = Color.Green) Then

       Endtime = RescheduleDailyDisplay.DataGridView1.Rows(clinicianRow).Cells(0).Value

                     scheduledOff(clinicianId, todaysdate, StartTime.Trim, Endtime.Trim)

                        Exit Do

                    End If

                    clinicianRow = clinicianRow + 1

                Loop

            End If

        Next

        Return Nothing

    End Function

End Class

'Save clinician off schedule into the data store.

Public Class updateClinicianOutSchedule

    Public Function scheduledOff(ByVal clinicianId As String, ByVal currentDate As Date, ByVal StartTime As String, ByVal Endtime As String)

        Dim scheduleclinician As IscheduleClinican = New alterSchedule

        scheduleclinician.UpdateSchedule(clinicianId, currentDate, StartTime, Endtime)

        Return Nothing

    End Function

End Class