Criterion E: Product development

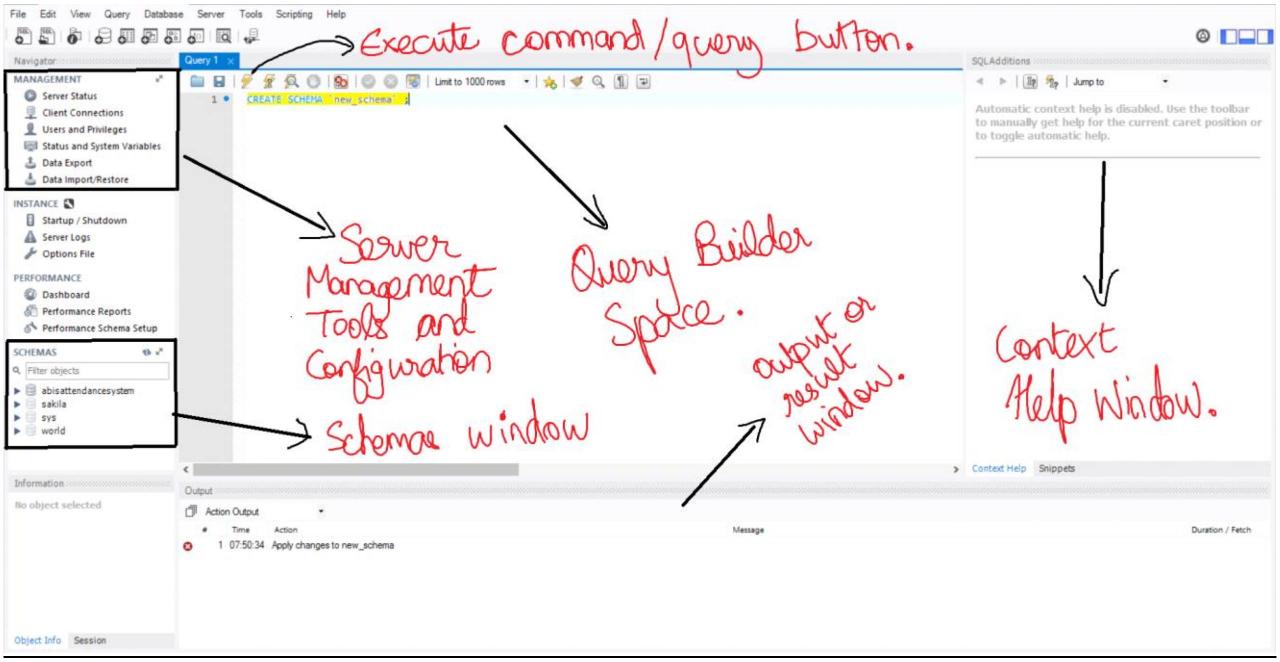
Complex techniques used to address the client's requirements:

- Integration of the database and program through:
  - If-then statements, and select case to perform events based on user input.
  - One dimensional arrays used in combination with loops to store multiple data from the database.
  - Procedures and Functions with Bypass parameters.
  - Queries to insert, view, and update records in the database.
  - Embedding the database into the Visual Basic (importing database resources), declaring connections, Try Catch Blocks, MySQLCommand and DataSet and DataTable Classes.
  - Displaying graphic charts by adding points in a series.

**Note: Visual Studio and MySQL Workbench are the sources used for this project.

MySQL Database Construction

After connecting to the SQL Server, the following screenshot below shows the resulting window.



```
1 CREATE SCHEMA `new_schema` ;
```

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

#	Time	Action	Message	Duration / Fetch
1	07:50:34	Apply changes to new_schema		

- *Query Builder Space:* This is where specific queries can be written to view, add, and filter data in the database. In MySQL, schema is a word which is synonymous to database. In the above screenshot, we perform the following query:

  CREATE SCHEMA 'abisattendancesystem' ;

- *Execute command/query button:* By clicking on this button, the typed query in the builder space is executed.

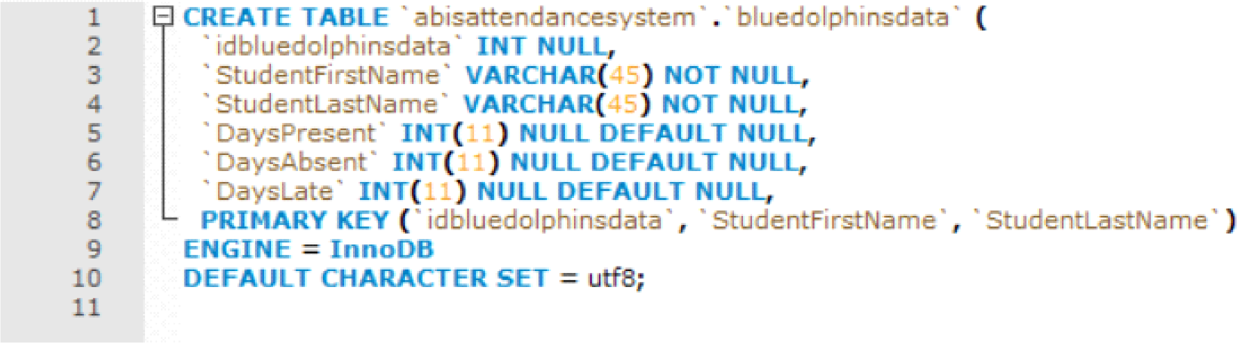
- *Server Management Tools and Configuration:* This is a panel which concerns itself with managing different client connections and privileges.

- *Schemas Window:* This is a panel which shows all of the schemas present in the workbench. By clicking the dropdown menu, user can get a list of the different tables, views, and datasets in the selected schema.

- *Output or Result Window:* This window serves to indicate whether the query has been successfully performed or not.

- *Context Help Window:* Whenever users are trying to search for a solution to some errors or problems in the database, they can use this window to search for some possible solutions on the MSDN Microsoft Website directly.

After creating the schema, we can now add DataTables to it which can be structured as follows in the query builder:



```
1   CREATE TABLE `abisattendancesystem`.`bluedolphinsdata` (
2     `idbluedolphinsdata` INT NULL,
3     `StudentFirstName` VARCHAR(45) NOT NULL,
4     `StudentLastName` VARCHAR(45) NOT NULL,
5     `DaysPresent` INT(11) NULL DEFAULT NULL,
6     `DaysAbsent` INT(11) NULL DEFAULT NULL,
7     `DaysLate` INT(11) NULL DEFAULT NULL,
8     PRIMARY KEY (`idbluedolphinsdata`, `StudentFirstName`, `StudentLastName`)
9   ENGINE = InnoDB
10  DEFAULT CHARACTER SET = utf8;
11
```

- Line 1 code is used to create the table "bluedolphinsdata" in the abisattendancesystem schema. The general structure can be defined as

  CREATE TABLE 'schema' . 'TableName'

- Data Types mentioned in the code are Integer and VarChar(45).

  - Integer: Data Type that accepts only positive and negative whole number values. This is appropriate for the day by day attendance count column, as number of days needs to be a whole number.

  - VarChar(45): A data type specific to MySQL where character input from the user is accepted but is limited to only 45 characters. This is appropriate for storing fields such as Students

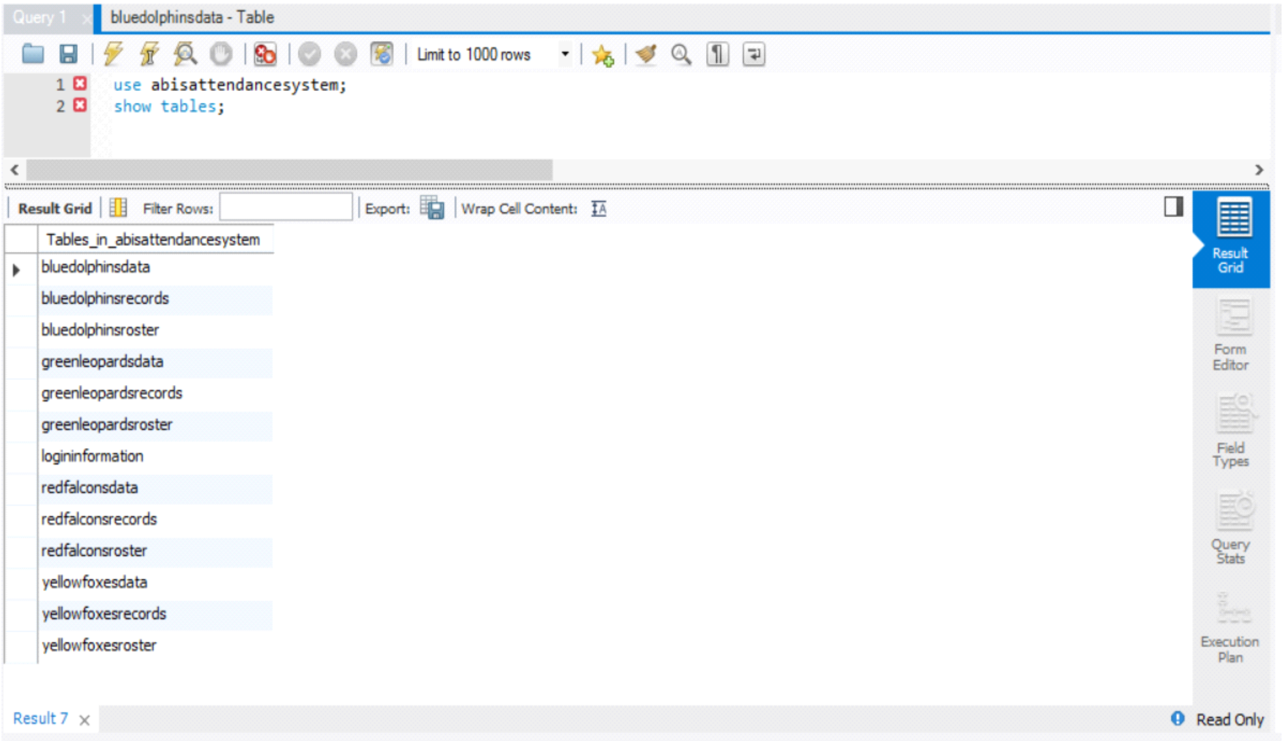
Names.

- On line 8, the primary key is defined and specific columns or fields that are marked as primary keys are mentioned. This takes the form of

  PRIMARY KEY ('column1' , 'column2' , 'column 3').

  In this case, the primary keys are the ID number, Student First Name, Student Last Name). This is to distinguish between the records in the database.

The same process mentioned above was used to create different tables in the schema. The screenshot below shows and addresses the hierarchical structure of the database:



```
use abisattendancesystem;
show tables;
```

Tables_in_abisattendancesystem
bluedolphinsdata
bluedolphinsrecords
bluedolphinsroster
greenleopardsdata
greenleopardsrecords
greenleopardsroster
logininformation
redfalconsdata
redfalconsrecords
redfalconsroster
yellowfoxesdata
yellowfoxesrecords
yellowfoxesroster

- The records table (such as bluedolphinsrecords) and roster tables are structured in the following manner for each of the teams (sample provided below is for blue team):
  - Records Table Structure:
    - idbluedolphinsrecords (DataType: Int).
    - Date (DataType: Date).
    - StudentFirstName (DataType: VarChar(45)).
    - StudentLastName (DataType: VarChar(45)).
    - Grade Level (DataType: Int).

- AttendanceRecorded (DataType: VarChar(45)).
- Comments (DataType: VarChar(45)).

- Roster Table Structure:
  - idbluedolphinsroster (DataType: Int).
  - StudentFirstName (DataType: VarChar(45)).
  - StudentLastName (DataType: VarChar(45)).
  - GradeLevel (DataType: Int).

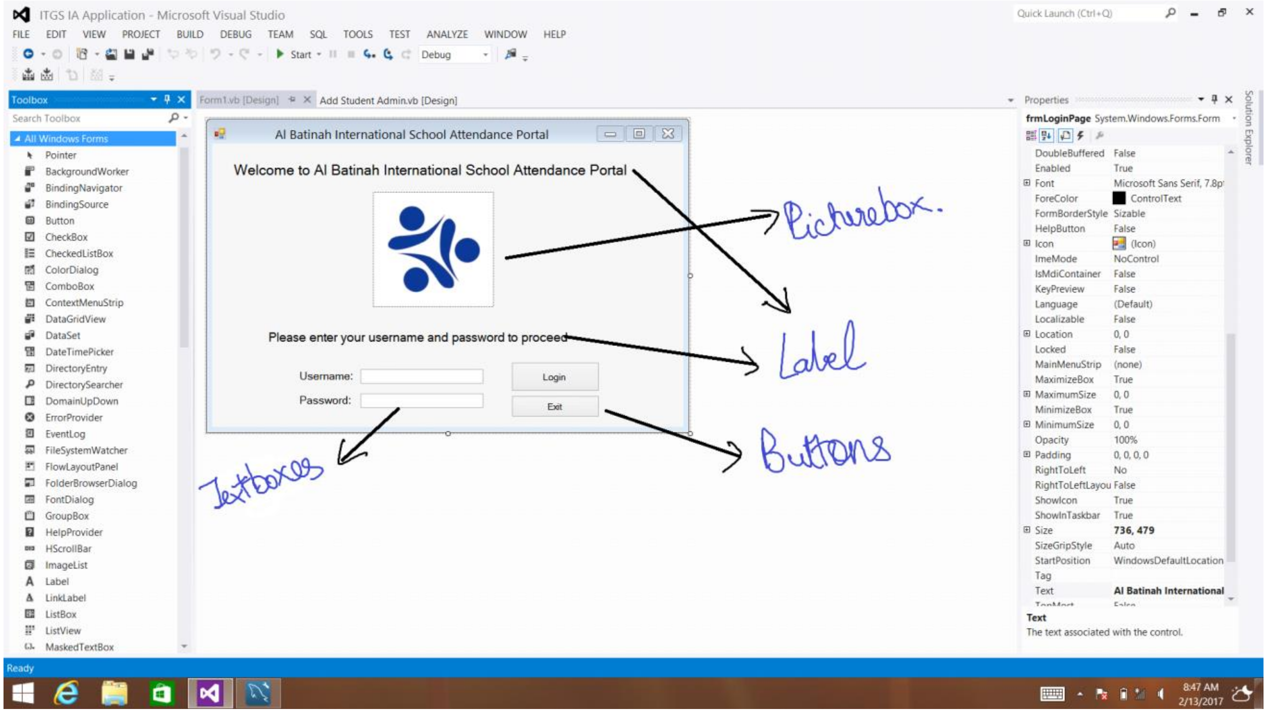
In order to view the records inside a specific DataTable, we need to use a different query. The query is of the structure as follows:

```
select * from abisattendancesystem.bluedolphinsdata;
```

Visual Studio 2012 Program (VB.NET)

Upon creation of the MySQL database, everything is now ready to be integrated into a user friendly application which allows teachers to take attendance and perform other functions.

To start off, a login page was made for the portal where the following designs and coding were implemented.



frmLoginPage System.Windows.Forms.Form

```
DoubleBuffered       False
Enabled              True
Font                 Microsoft Sans Serif, 7.8pt
ForeColor            ControlText
FormBorderStyle      Sizable
HelpButton           False
Icon                 (Icon)
ImeMode              NoControl
IsMdiContainer       False
KeyPreview           False
Language             (Default)
Localizable          False
Location             0, 0
Locked               False
MainMenuStrip        (none)
MaximizeBox          True
MaximumSize          0, 0
MinimizeBox          True
MinimumSize          0, 0
Opacity              100%
Padding              0, 0, 0, 0
RightToLeft          No
RightToLeftLayou     False
ShowIcon             True
ShowInTaskbar        True
Size                 736, 479
SizeGripStyle        Auto
StartPosition        WindowsDefaultLocation
Tag
Text                 Al Batinah International
```

Text
The text associated with the control.

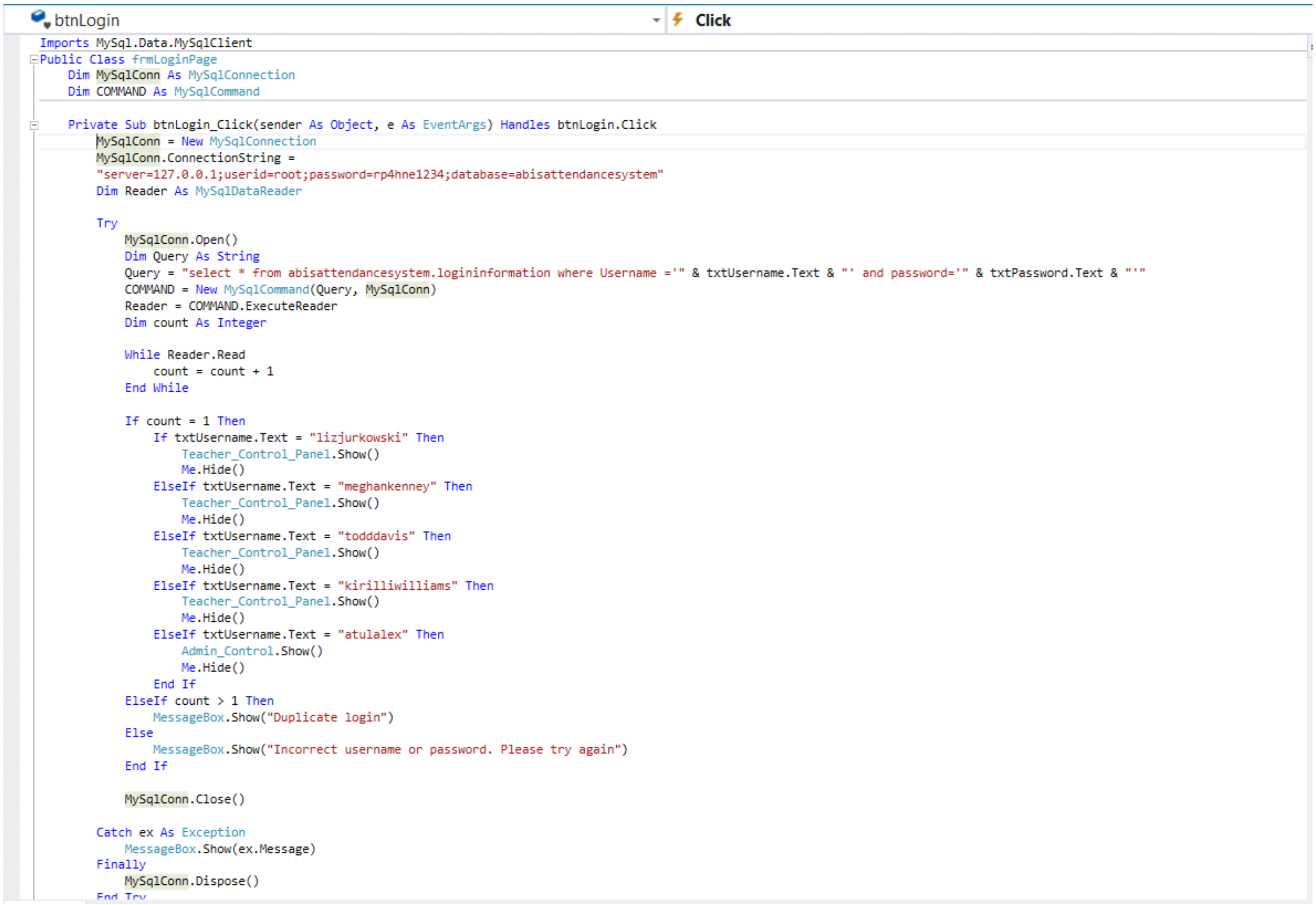
- *Picturebox:* This tool is used to view and set some pictures. To do so, the picture must be in the jpeg, jpg formats only, and must be first copied to the resources folder of the project.

- *Label:* A label is used to hold certain information or text and display it to the user. This can be a prompt, instruction, or a welcome message, as indicated on this form.

- *Buttons:* Buttons are used to perform click events. This means that whenever the user clicks the button, an event should occur.

- *Textboxes:* Textboxes are most frequently used to get the user's input. In this case, the user has been asked to type a username and password.

Now, below is the code viewer window where certain events can be coded to perform specific actions for the Login Page.



```
Imports MySql.Data.MySqlClient
Public Class frmLoginPage
    Dim MySqlConn As MySqlConnection
    Dim COMMAND As MySqlCommand

    Private Sub btnLogin_Click(sender As Object, e As EventArgs) Handles btnLogin.Click
        MySqlConn = New MySqlConnection
        MySqlConn.ConnectionString =
        "server=127.0.0.1;userid=root;password=rp4hne1234;database=abisattendancesystem"
        Dim Reader As MySqlDataReader

        Try
            MySqlConn.Open()
            Dim Query As String
            Query = "select * from abisattendancesystem.logininformation where Username ='" & txtUsername.Text & "' and password='" & txtPassword.Text & "'"
            COMMAND = New MySqlCommand(Query, MySqlConn)
            Reader = COMMAND.ExecuteReader
            Dim count As Integer

            While Reader.Read
                count = count + 1
            End While

            If count = 1 Then
                If txtUsername.Text = "lizjurkowski" Then
                    Teacher_Control_Panel.Show()
                    Me.Hide()
                ElseIf txtUsername.Text = "meghankenney" Then
                    Teacher_Control_Panel.Show()
                    Me.Hide()
                ElseIf txtUsername.Text = "todddavis" Then
                    Teacher_Control_Panel.Show()
                    Me.Hide()
                ElseIf txtUsername.Text = "kirilliwilliams" Then
                    Teacher_Control_Panel.Show()
                    Me.Hide()
                ElseIf txtUsername.Text = "atulalex" Then
                    Admin_Control.Show()
                    Me.Hide()
                End If
            ElseIf count > 1 Then
                MessageBox.Show("Duplicate login")
            Else
                MessageBox.Show("Incorrect username or password. Please try again")
            End If

            MySqlConn.Close()

        Catch ex As Exception
            MessageBox.Show(ex.Message)
        Finally
            MySqlConn.Dispose()
        End Try
```

- Line 1 "Imports MySql.Data.MySqlClient" is used to import a class of MySQL and allow user to gather and perform actions with the data.

- The try block consists of gathering the input from the user and comparing it to the ones present in database.

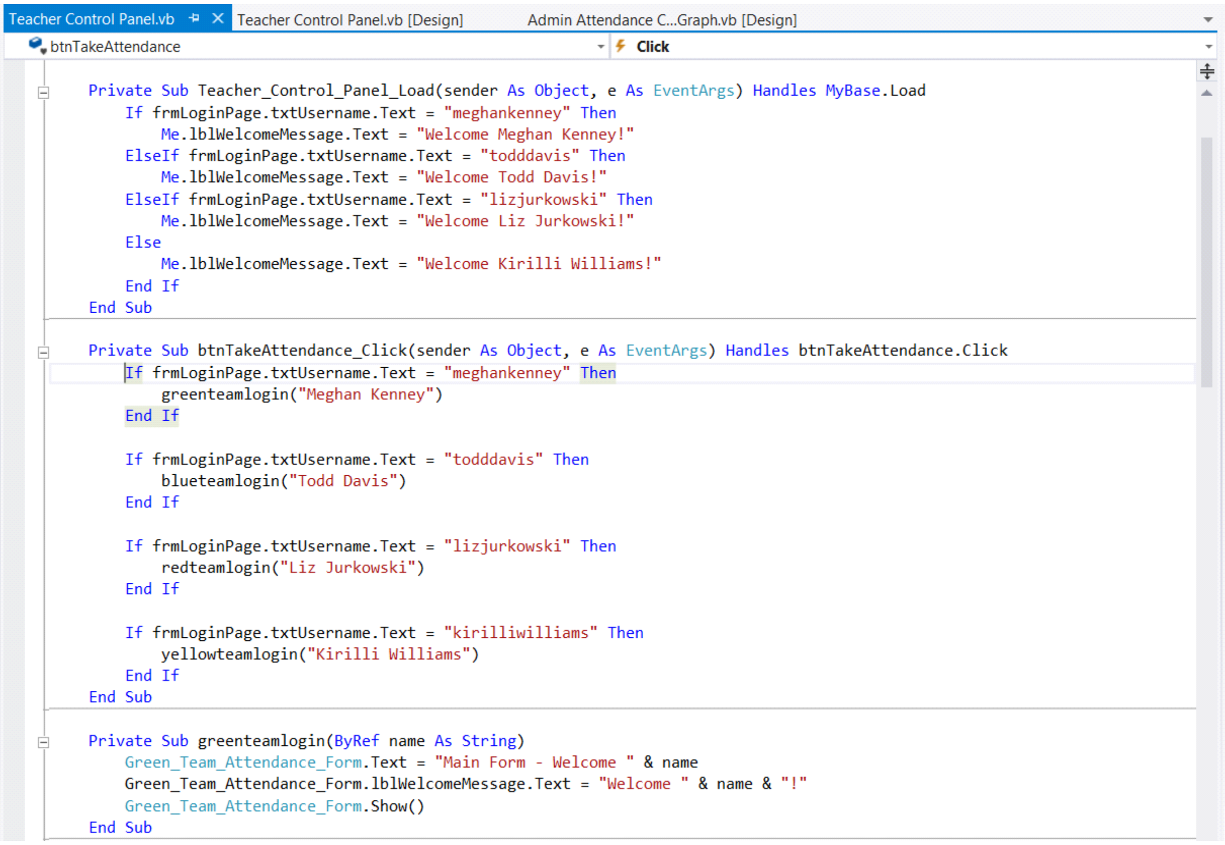
  - In the try block, a query is declared and assigned a string expression. This query specifically makes use of the Where Clause, which filters data according to the condition provided. In this case, the user input is taken and concatenated in the query to see whether there are respective matches in the database.

    - After writing the query, there is a command code, which requires two parameters of the query and the connection. This command code is then taken and passed onto the reader to execute the command.

    - The while loop is used to go through each record in the database and increment the count by 1, if the specific query is performed WHERE the user input matches the data. The while loop will only end once it has gone through each of the records in the database.

- If in any case, an error occurs, then the catch block is executed to display an exception message. Also, by using the Finally statement, the connection is disposed after it has performed the event.

After creating and setting up the login form, we can make another form for the teachers where, if login is successful, then the dashboard for the appropriate teacher is loaded.


Teacher Control Panel.vb
Teacher Control Panel.vb [Design]
Admin Attendance C...Graph.vb [Design]
btnTakeAttendance
Click

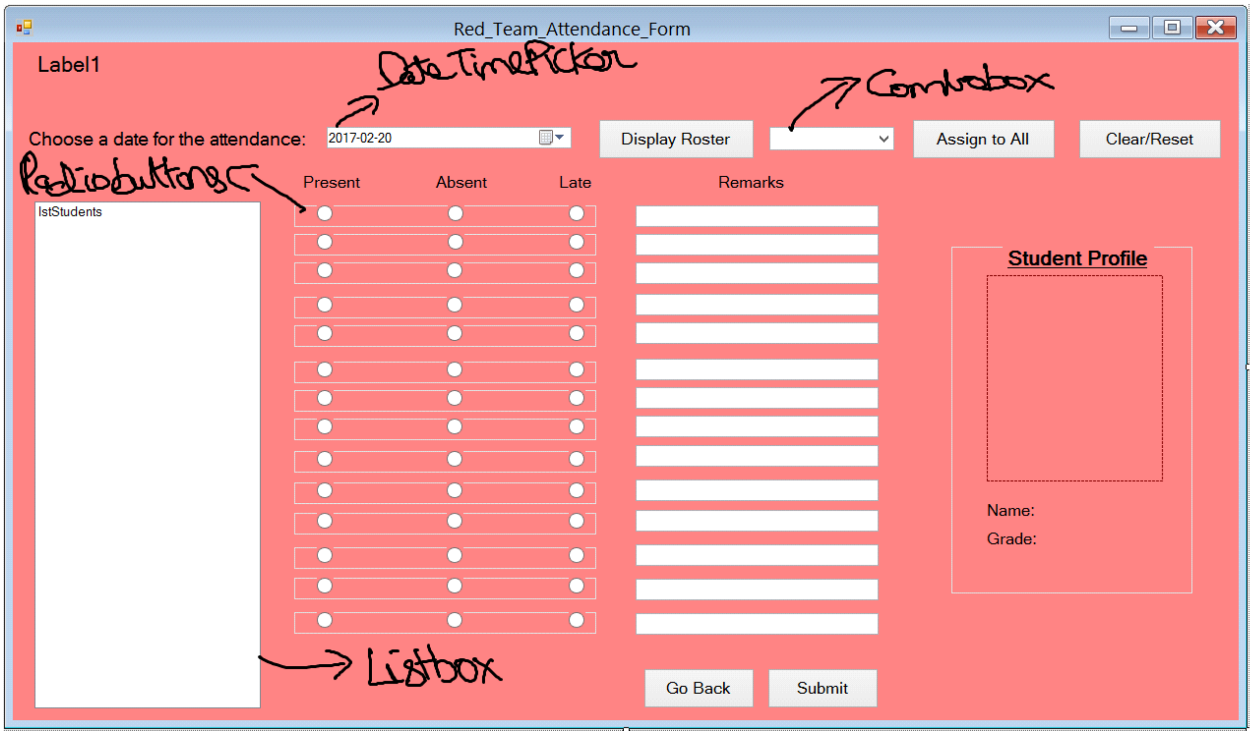
```
Private Sub Teacher_Control_Panel_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    If frmLoginPage.txtUsername.Text = "meghankenney" Then
        Me.lblWelcomeMessage.Text = "Welcome Meghan Kenney!"
    ElseIf frmLoginPage.txtUsername.Text = "todddavis" Then
        Me.lblWelcomeMessage.Text = "Welcome Todd Davis!"
    ElseIf frmLoginPage.txtUsername.Text = "lizjurkowski" Then
        Me.lblWelcomeMessage.Text = "Welcome Liz Jurkowski!"
    Else
        Me.lblWelcomeMessage.Text = "Welcome Kirilli Williams!"
    End If
End Sub

Private Sub btnTakeAttendance_Click(sender As Object, e As EventArgs) Handles btnTakeAttendance.Click
    If frmLoginPage.txtUsername.Text = "meghankenney" Then
        greenteamlogin("Meghan Kenney")
    End If

    If frmLoginPage.txtUsername.Text = "todddavis" Then
        blueteamlogin("Todd Davis")
    End If

    If frmLoginPage.txtUsername.Text = "lizjurkowski" Then
        redteamlogin("Liz Jurkowski")
    End If

    If frmLoginPage.txtUsername.Text = "kirilliwilliams" Then
        yellowteamlogin("Kirilli Williams")
    End If
End Sub

Private Sub greenteamlogin(ByRef name As String)
    Green_Team_Attendance_Form.Text = "Main Form - Welcome " & name
    Green_Team_Attendance_Form.lblWelcomeMessage.Text = "Welcome " & name & "!"
    Green_Team_Attendance_Form.Show()
End Sub
```

- Procedures are used with parameters for the TakeAttendance button click event. Here, different procedures are set for the different teams, and are called whenever the condition is satisfied in the if-then statements.

- Based on the txtUsername input, the relevant procedure is called and the teacher's name is sent to the procedure in form of a string through the ByRef parameter.

- A ByRef parameter is a reference parameter that can both accept the data and return it back to the line of code where it was called.

Below is a screenshot of the Designer View of the Attendance Form and also the code. The following was done for 4 different teams:



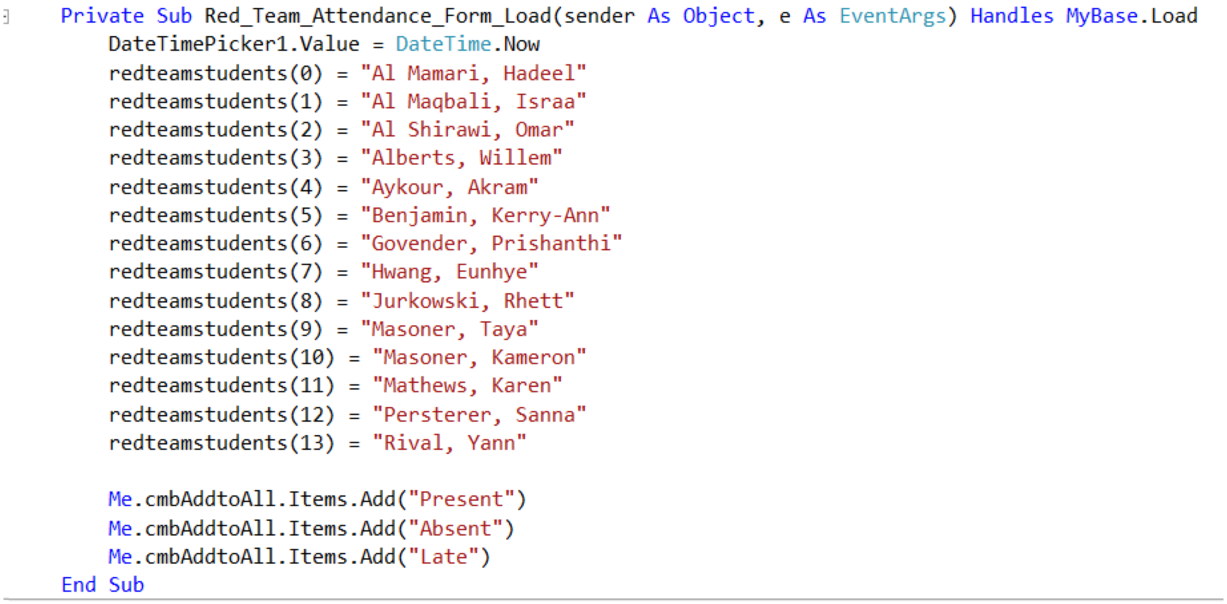
- *DateTimePicker:* This tool is used to give user a calendar of dates where they can choose a specific date for recording the attendance. The format of the date in the Visual Basic program is changed to DD/MM/YY in order to record the appropriate date type format in MySQL Database.

- *Listbox:* In a way, this works like a table where different items can be added and displayed in rows.

- *Combobox:* This is similar to a dropdown list or menu where the user gets different options upon clicking the dropdown button. The items added to this combobox are Present, Absent, and Late.

- *Radiobuttons:* Using this, the attendance array is recorded for each student, where based on the checked radiobutton, the respective attendance is marked in the database.

The two complex techniques addressed in the code below are one dimensional arrays and insert queries.



```
Private Sub Red_Team_Attendance_Form_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    DateTimePicker1.Value = DateTime.Now
    redteamstudents(0) = "Al Mamari, Hadeel"
    redteamstudents(1) = "Al Maqbali, Israa"
    redteamstudents(2) = "Al Shirawi, Omar"
    redteamstudents(3) = "Alberts, Willem"
    redteamstudents(4) = "Aykour, Akram"
    redteamstudents(5) = "Benjamin, Kerry-Ann"
    redteamstudents(6) = "Govender, Prishanthi"
    redteamstudents(7) = "Hwang, Eunhye"
    redteamstudents(8) = "Jurkowski, Rhett"
    redteamstudents(9) = "Masoner, Taya"
    redteamstudents(10) = "Masoner, Kameron"
    redteamstudents(11) = "Mathews, Karen"
    redteamstudents(12) = "Persterer, Sanna"
    redteamstudents(13) = "Rival, Yann"

    Me.cmbAddtoAll.Items.Add("Present")
    Me.cmbAddtoAll.Items.Add("Absent")
    Me.cmbAddtoAll.Items.Add("Late")
End Sub
```

As shown in the above code, each node is assigned a string expression with the students' names. In addition to this, an attendance recorded array is also used to take the teacher's input from the radiobuttons and assign the appropriate attendance to each student as a string variable. This array is used as part of the code below, where insert queries are also used:

```
"insert into abisattendancesystem.redfalconsrecords
(idredfalconsrecords,Date,StudentFirstName,StudentLastName,GradeLevel,AttendanceRecorded,
Comments) values (9, '" & DateTimePicker1.Text & "', 'Rhett', 'Jurkowski', 7 ,'" &
attendancerecorded(8) & "', '" & txtRhettJurkowskiRemarks.Text & "')"
```

- The insert into query is used to take the data from the attendance program and transfer it into the database.

- This is followed by values and the appropriate records that need to be inserted into the database table. Here, the array is used to provide the appropriate attendance for the student.

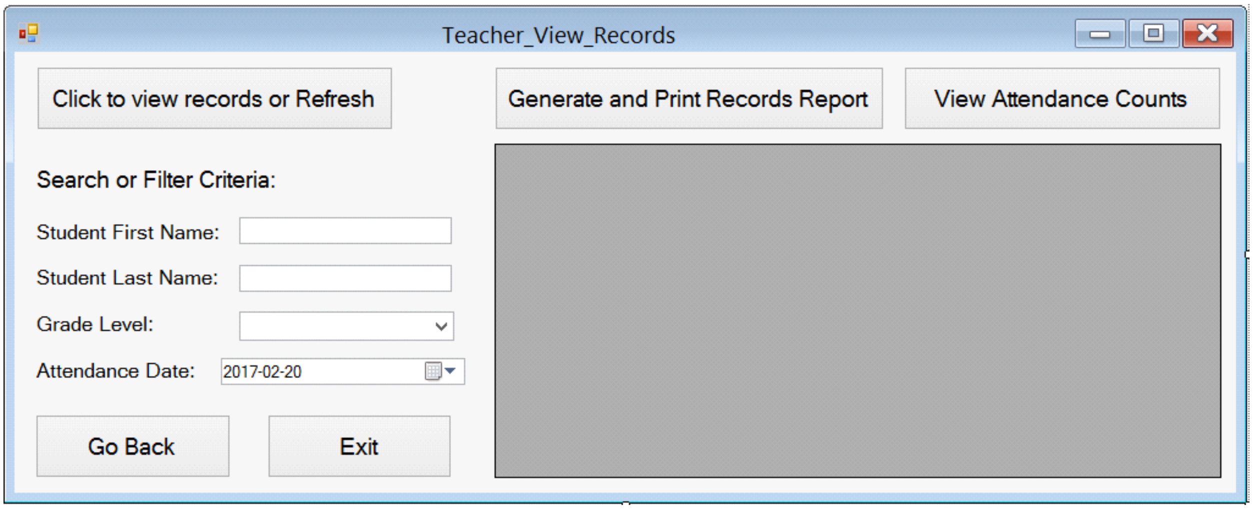
Along with the insert queries, update queries are also used to change the attendance counts for each student.

```
"update abisattendancesystem.redfalconsdata set idredfalconsdata 3,StudentFirstName='" &
StudentFirstName(2) & "',StudentLastName='" & StudentLastName(2) & "',DaysPresent='" &
PresentAttendanceCount(2) & "',DaysAbsent='" & AbsentAttendanceCount(2) & "',DaysLate='" &
LateAttendanceCount(2) & "' where idredfalconsdata=3"
```

- The above code makes use of the WHERE clause to determine the record in the database that needs to be changed. Specifically, the ID number is used over here to update the attendance count for the indicated student with the ID.

- The new data is specified after the word "set" and the appropriate count is updated by 1, based on the teacher's input.

After the data is entered into the database, the teachers have many other options, one of which include viewing the records (in the form of both a table and a chart).

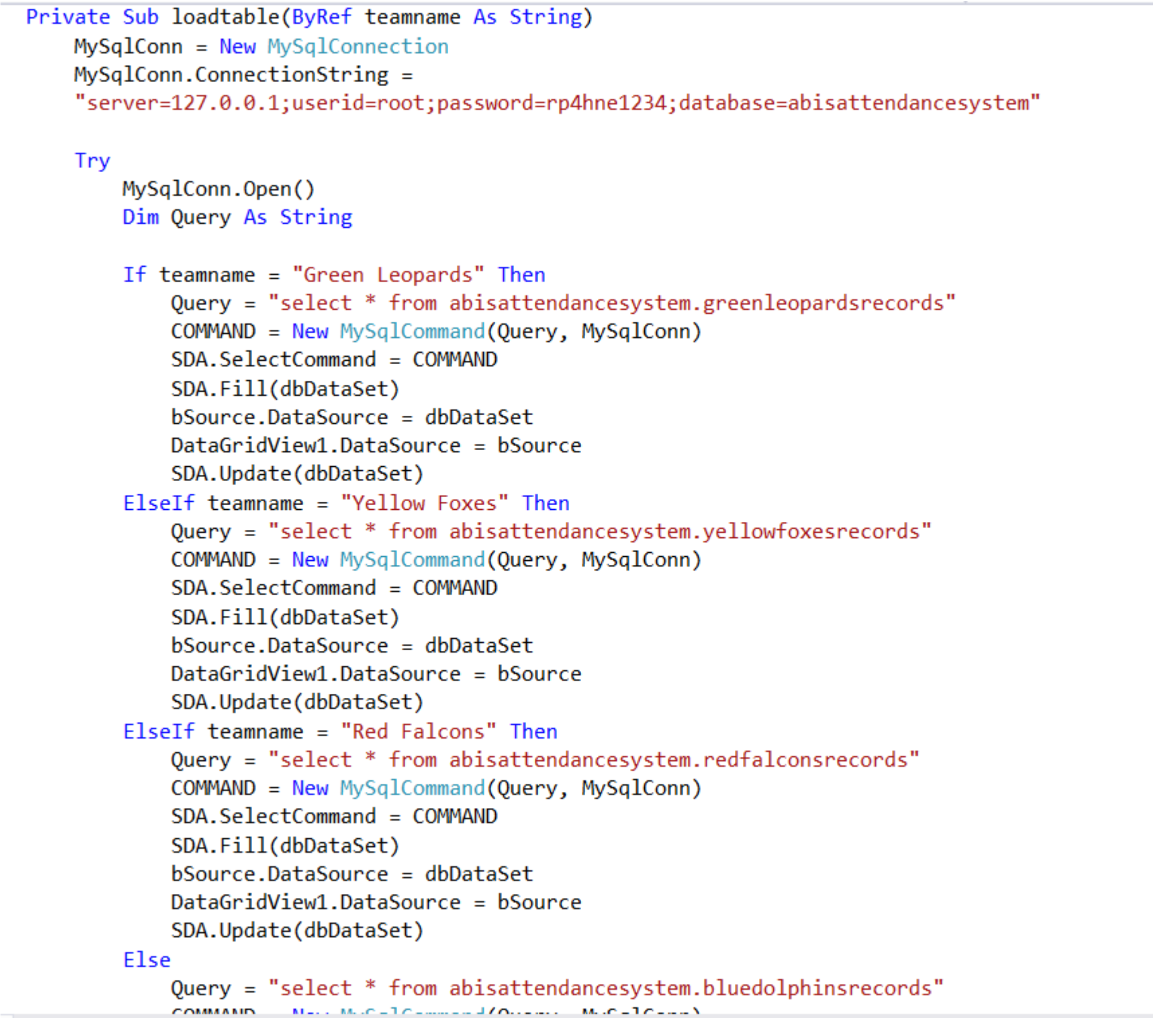

Teacher_View_Records
Click to view records or Refresh
Generate and Print Records Report
View Attendance Counts
Search or Filter Criteria:
Go Back
Exit

Student First Name:

Student Last Name:

Grade Level:

Attendance Date: 2017-02-20

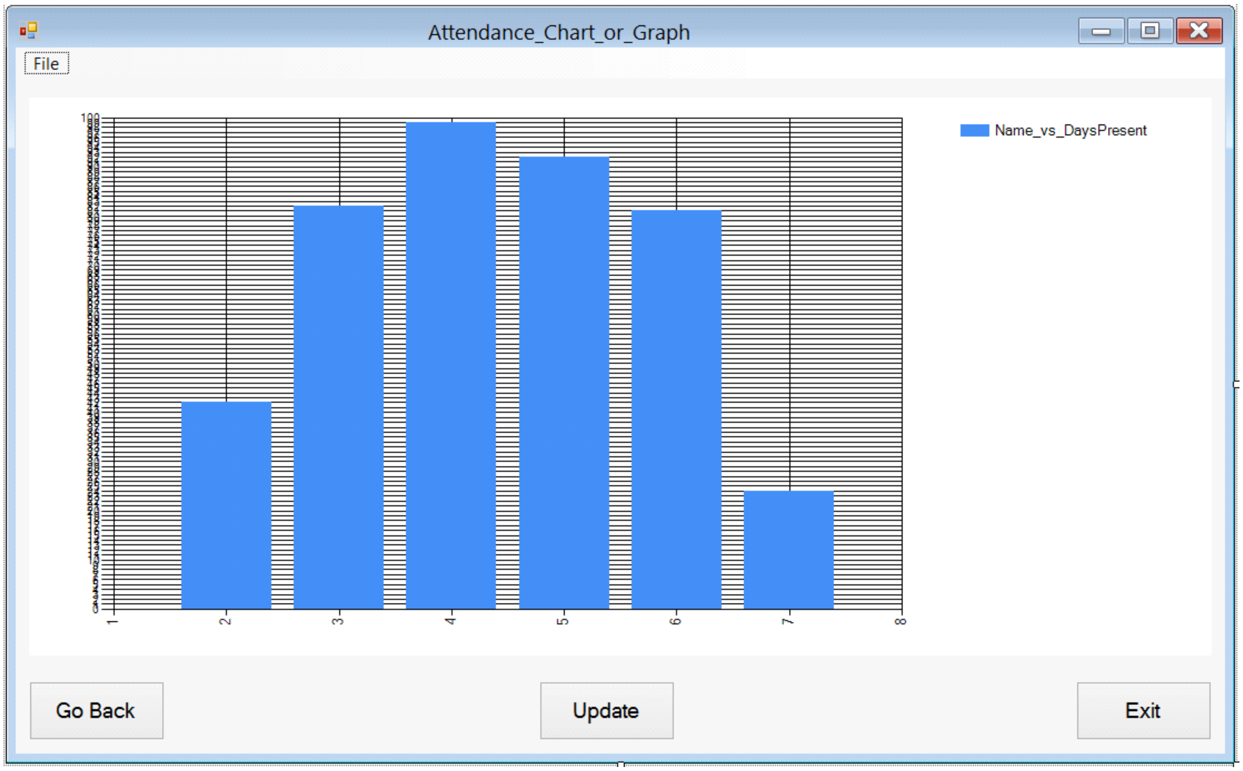


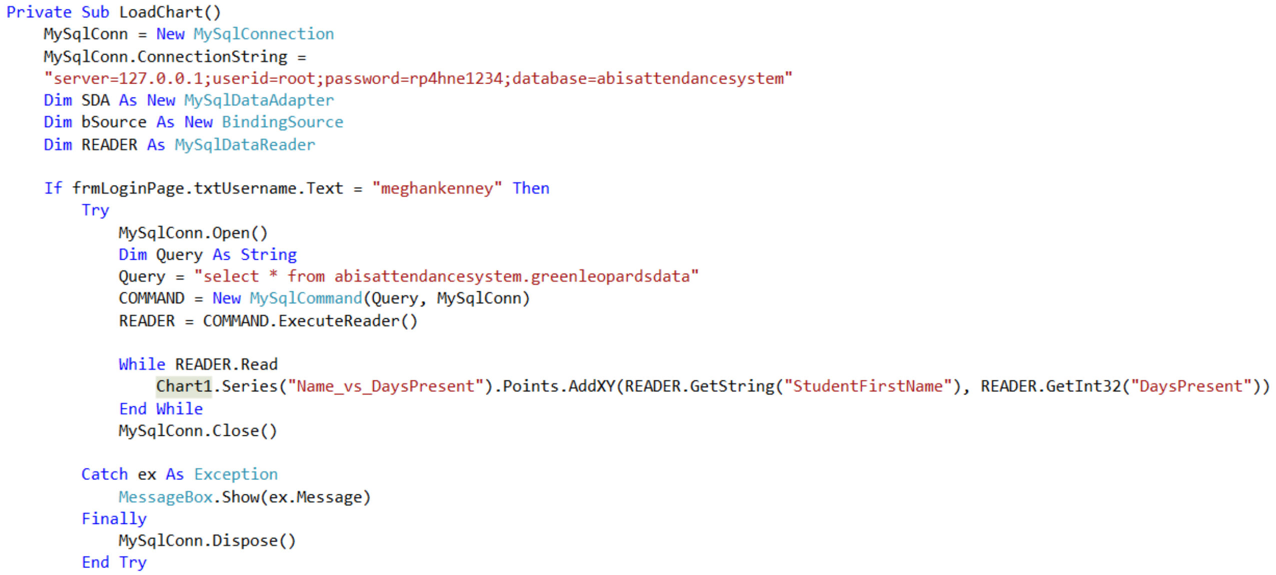
```
Private Sub loadtable(ByRef teamname As String)
    MySqlConn = New MySqlConnection
    MySqlConn.ConnectionString =
    "server=127.0.0.1;userid=root;password=rp4hne1234;database=abisattendancesystem"

    Try
        MySqlConn.Open()
        Dim Query As String

        If teamname = "Green Leopards" Then
            Query = "select * from abisattendancesystem.greenleopardsrecords"
            COMMAND = New MySqlCommand(Query, MySqlConn)
            SDA.SelectCommand = COMMAND
            SDA.Fill(dbDataSet)
            bSource.DataSource = dbDataSet
            DataGridView1.DataSource = bSource
            SDA.Update(dbDataSet)
        ElseIf teamname = "Yellow Foxes" Then
            Query = "select * from abisattendancesystem.yellowfoxesrecords"
            COMMAND = New MySqlCommand(Query, MySqlConn)
            SDA.SelectCommand = COMMAND
            SDA.Fill(dbDataSet)
            bSource.DataSource = dbDataSet
            DataGridView1.DataSource = bSource
            SDA.Update(dbDataSet)
        ElseIf teamname = "Red Falcons" Then
            Query = "select * from abisattendancesystem.redfalconsrecords"
            COMMAND = New MySqlCommand(Query, MySqlConn)
            SDA.SelectCommand = COMMAND
            SDA.Fill(dbDataSet)
            bSource.DataSource = dbDataSet
            DataGridView1.DataSource = bSource
            SDA.Update(dbDataSet)
        Else
            Query = "select * from abisattendancesystem.bluedolphinsrecords"
            COMMAND = New MySqlCommand(Query, MySqlConn)
```

- In order to import MySQL data into the DataGridView, a DataAdapter and DataSet are needed. Therefore, the command is assigned to the DataAdapter, and using the appropriate DataTable, the DataAdapter is filled with the selected query.

- The DataSource of the DataGridView is then filled with the DataSet obtained from the DataAdapter.

Below shows the design and the code view for the Attendance Chart or Graph Form:





```
Private Sub LoadChart()
    MySqlConn = New MySqlConnection
    MySqlConn.ConnectionString =
    "server=127.0.0.1;userid=root;password=rp4hne1234;database=abisattendancesystem"
    Dim SDA As New MySqlDataAdapter
    Dim bSource As New BindingSource
    Dim READER As MySqlDataReader

    If frmLoginPage.txtUsername.Text = "meghankenney" Then
        Try
            MySqlConn.Open()
            Dim Query As String
            Query = "select * from abisattendancesystem.greenleopardsdata"
            COMMAND = New MySqlCommand(Query, MySqlConn)
            READER = COMMAND.ExecuteReader()

            While READER.Read
                Chart1.Series("Name_vs_DaysPresent").Points.AddXY(READER.GetString("StudentFirstName"), READER.GetInt32("DaysPresent"))
            End While
            MySqlConn.Close()

        Catch ex As Exception
            MessageBox.Show(ex.Message)
        Finally
            MySqlConn.Dispose()
        End Try
```

- On the design form above, the Chart tool is used to display the data provided.

- A while loop is used to add points to the chart on the design view. The statement inside the while loop block is used to take the data from the DataTable (using the Reader.GetDataType structure).

- The statement is as follows:

  -
    ```
    Chart1.Series("TITLE").Points.AddXY(READER.GetString("COLUMNXNAME"),
    READER.GetInt32("COLUMNYNAME"))
    ```

Word Count: 1148