

# **Student Management System**

## **A MINOR PROJECT REPORT**

**Submitted in partial fulfillment of the requirement for the award of Degree of  
IMCA**

**Submitted to**



**RAJIV GANDHI PRODYOGIKI VISHWAVIDYALAYA, BHOPAL (M.P.)**

**Submitted by:**

**Mr. Siddharth Jain**

**Enrollment No.: 0827CA21DD54**

**Under the Supervision of**

**Prof. Shruti Vijayvargiya**



**IMCA**

**ACROPOLIS INSTITUTE OF TECHNOLOGY & RESEARCH, INDORE**

**SESSION: 2022-23**

# **Faculty of Computer Applications**

## **AITR, Indore**

### **BONAFIDE CERTIFICATE**

This is to certify that Minor Project entitled “**Student Management System**” being submitted by **Mr. Siddharth Jain (0827CA21DD54)** for partial fulfillment of the requirement for the award of degree of **Integrated Master of Computer Applications** to **Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal (M.P.)** during the academic year 2022-23 is a record of bonafide piece of work, carried out by the student under my supervision and guidance in the **Faculty of Computer Applications, AITR, Indore.**

**(Supervisor)**

Designation, AITR, Indore

**(Prof. Geeta Santhosh)**

Professor & HOD, MCA

**(Internal Examiner)**

**Date:**

**(External Examiner)**

**Date:**

# Table of Contents

S.No.	Content	Page No.
1.	Title Page	1
2.	Certificate	2
3.	Table of Contents	3
4.	Introduction/Objective Of Project	4
5.	Module Description	6
6.	Design/Flow Chart/ER Diagram	8
7.	List of tables	9
8.	Testing/Test Cases	10
9.	Front End Forms	11
10.	Source Code	16
11.	Conclusion	27
12.	Bibliography	28

# Introduction/Objective of Project:

## INTRODUCTION

The Student Management System is a comprehensive software solution designed to streamline the management of student data in educational institutions. This project aims to provide a user-friendly interface for efficient CRUD (Create, Read, Update, Delete) operations on student records, along with a dashboard that highlights important statistics.

The dashboard feature of the Student Management System plays a crucial role in providing valuable insights and statistics related to student demographics and enrollment trends. It offers a visually appealing and interactive interface that showcases key information at a glance.

The dashboard highlights the number of male and female students, providing a gender distribution overview. This helps administrators and educators understand the gender balance within the student population and make informed decisions related to student support programs and resource allocation.

Additionally, the dashboard displays the number of students studying in each course, enabling stakeholders to track the popularity and demand for different programs. This information assists in curriculum planning, resource allocation, and identifying potential areas for program expansion.

Furthermore, the dashboard presents data on the number of students studying in each academic year, ranging from the 1st year to the final year. This provides insights into student retention rates, progression through the academic program, and overall student satisfaction.

## **OBJECTIVE**

The primary objective of this project is to simplify the process of managing student information, reducing manual paperwork, and ensuring data accuracy. By centralizing student records and automating administrative tasks, the system aims to enhance efficiency and productivity in educational institutions.

The objective of this project is to create a robust and user-friendly system that simplifies student management processes, improves data accuracy, and provides valuable insights through a comprehensive dashboard. By automating routine tasks and centralizing student information, the system aims to enhance efficiency, productivity, and decision-making within educational institutions. It seeks to empower administrators, teachers, and students with a reliable and efficient tool for managing student data, enrollment processes, and monitoring academic progress. The intuitive dashboard further enriches the user experience by offering a visual representation of key statistics and trends, facilitating informed decision-making and strategic planning.

In summary, the dashboard feature of the Student Management System serves as a powerful tool for administrators and educators, providing valuable insights into student demographics, course popularity, and academic progression. It enhances decision-making capabilities, promotes transparency, and supports the efficient management of educational institutions.

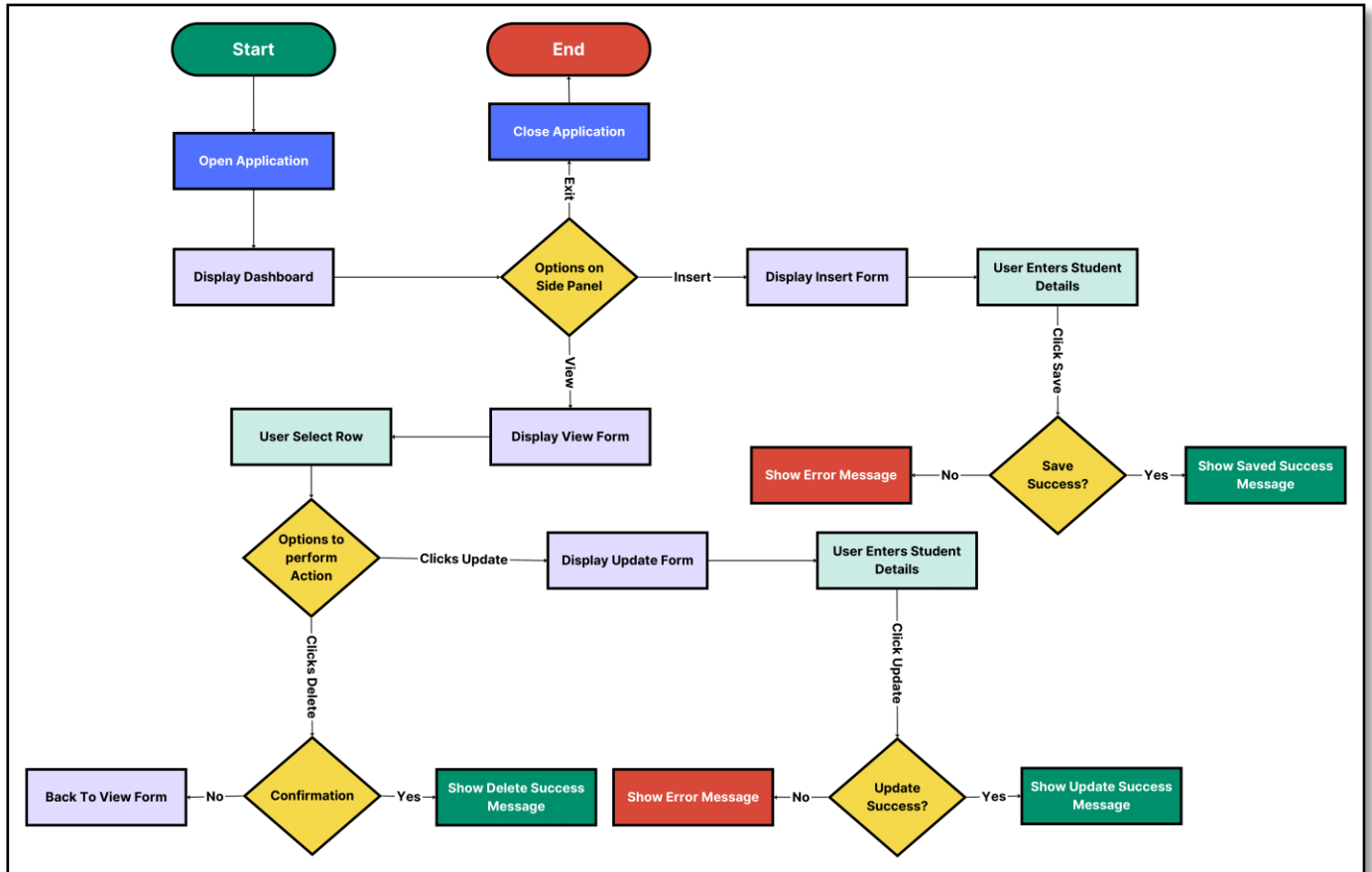
# Module Description:

1. **Insert Module:** The Insert module is responsible for adding new student records into the system. It provides a user-friendly interface where administrators or authorized users can enter the necessary details of a student, including their personal information, contact details, course enrollment, and year of admission. The module ensures data validation and integrity checks to maintain accurate records.
2. **Dashboard Module:** The Dashboard module serves as a central hub that provides a comprehensive overview of student-related data and statistics. It presents important information through a set of visually appealing cards. The dashboard displays key metrics, including the total number of male and female students, the number of students in each course, and the distribution of students across different academic years. These cards offer a quick glance at the essential statistics, enabling users to have an instant understanding of the student population.
3. **View Module:** The View module allows users to access and view the student data stored in the system. It provides a tabular representation of the student records, displaying information such as roll number, name, course, year of admission, and contact details. Users can apply filters, search for specific records, and sort the data based on different criteria to easily locate and retrieve the desired information.
4. **Search Module:** The Search module facilitates easy and quick retrieval of student records based on specific search criteria. Users can search for students by entering their roll number or name in the search field. The module then displays the matching results, allowing users to view the details of the desired student or group of students. This feature is especially useful when administrators or users need to quickly access specific student information without browsing through the entire student database.
5. **Update Module:** The Update module enables authorized users to modify and update the existing student records. It offers a convenient interface where administrators or designated personnel can make changes to student information, such as contact details, course enrollment, or any other relevant data. The module ensures data integrity by validating the inputs and ensuring that only authorized users can perform updates.
6. **Delete Module:** The Delete module allows authorized users to remove student records from the system. It provides a secure mechanism to permanently delete student data that is no longer required or relevant. Users can select specific records or apply filters to identify

the records to be deleted. The module incorporates necessary safeguards and confirmation prompts to prevent accidental deletion of data.

The module-based structure of the Student Management System ensures a modular and scalable design, allowing for easy maintenance, updates, and future enhancements. Each module performs specific functions, contributing to the overall efficiency and effectiveness of managing student-related data within educational institutions.

# Design/Flow Chart/ER Diagram





# List of tables

## **STUDENTS TABLE:**

### **Description:**

The "Students" table includes the essential fields required to store student information in the Student Management System. These fields capture the student's unique identifier, name, father's name, contact details, gender, course details, year of admission, and date of birth. The table serves as the foundation for managing and organizing student data efficiently.

### **Columns:**

1. RollNo: A unique identifier for each student.
2. FName: First name of the student.
3. LName: Last name of the student.
4. FatherName: Name of the student's father.
5. MobNo: Mobile number of the student.
6. AltMobNo: Alternate mobile number of the student.
7. Gender: Gender of the student (e.g., Male, Female).
8. Course: Course in which the student is enrolled.
9. YOA: Year of admission of the student.
10. DOB: Date of birth of the student.

## Testing/Test Cases

Test Case	Test Condition	Test Input	Expected Result	Test Input	Test Result
1	Inserting a New Student	Valid student data	New student is successfully inserted into the database	RollNo: 101, FName: Rahul, LName: Sharma, FatherName: Shyam, MobNo: 1234567890, AltMobNo: 9876543210, Gender: Male, Course: BCA, YOA: 2022, DOB: 01/01/2000	Passed
2	Updating Student Details	Existing student data	Student details are successfully updated in the database	RollNo: 101, FName: Rahul, LName: Sharma, MobNo: 9876543210	Passed
3	Deleting a Student Record	Existing student data	Student record is successfully deleted from the database	RollNo: 101	Passed
4	Searching for a Student	Valid search criteria	Student matching the search criteria is displayed	RollNo: 101	Passed
5	Validation of Mandatory Fields	Missing mandatory field(s)	Appropriate error message(s) are displayed	FName: Rahul, LName: Sharma, Course: (blank)	Passed

# Front End Forms

## 1. DASHBOARD FORM:

### 1.1. Form Description:

The Dashboard Form serves as the main interface of the application, providing an overview of key statistics and data related to student management.

### 1.2. Fields/Components:

- 1.2.1. Title Label: Displays the title of the form as "Dashboard".
- 1.2.2. Total Students Card: Shows the total number of students in the database.
- 1.2.3. Male Students Card: Displays the count of male students in the database.
- 1.2.4. Female Students Card: Displays the count of female students in the database.
- 1.2.5. BCA Students Card: Shows the count of students enrolled in the BCA course.
- 1.2.6. MCA Students Card: Shows the count of students enrolled in the MCA course.
- 1.2.7. IIMCA Students Card: Displays the count of students enrolled in the IIMCA course.

## 2. INSERT FORM:

### 2.1. Form Description:

The Insert Form allows users to add new student records to the database.

### 2.2. Fields/Components:

- 2.2.1. Roll Number (Text Field): For entering the roll number of the student.
- 2.2.2. First Name (Text Field): For entering the first name of the student.
- 2.2.3. Last Name (Text Field): For entering the last name of the student.
- 2.2.4. Father's Name (Text Field): For entering the father's name of the student.
- 2.2.5. Mobile Number (Text Field): For entering the mobile number of the student.
- 2.2.6. Alternate Mobile Number (Text Field): For entering an alternate mobile number of the student.
- 2.2.7. Gender (Dropdown List): For selecting the gender of the student.
- 2.2.8. Course (Dropdown List): For selecting the course of the student.
- 2.2.9. Year of Admission (Automatically Calculated): Based on the current year.
- 2.2.10. Date of Birth (Date Picker): For selecting the date of birth of the student.
- 2.2.11. Save Button: To submit the data and add the new student record to the database.

## 3. View Form:

### 3.1. Form Description:

The View Form displays a tabular representation of all student records stored in the database.

### **3.2. Fields/Components:**

- 3.2.1. Search Field (Text Field): For searching students by roll number or name.
- 3.2.2. Student Table (DataGridView): Showing all student records with columns for roll number, name, gender, course, year of admission, and other details.
- 3.2.3. Update Button: To open the Update Form for modifying the selected student record.
- 3.2.4. Delete Button: To delete the selected student record.

## **4. Update Form:**

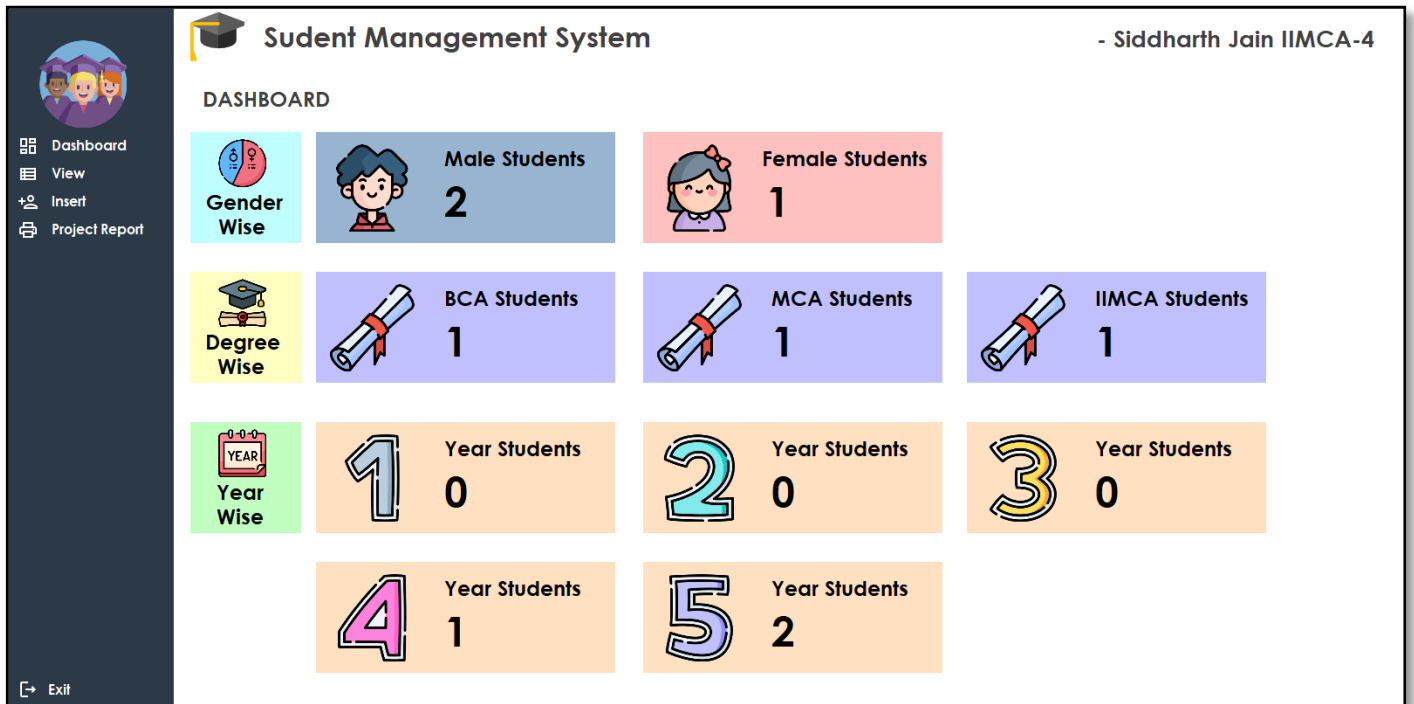
### **4.1. Form Description:**

The Update Form allows users to modify the details of an existing student record.

### **4.2. Fields/Components:**

- 4.2.1. Roll Number (Read-only Text Field): Displaying the roll number of the student (cannot be modified).
- 4.2.2. First Name (Text Field): For editing the first name of the student.
- 4.2.3. Last Name (Text Field): For editing the last name of the student.
- 4.2.4. Father's Name (Text Field): For editing the father's name of the student.
- 4.2.5. Mobile Number (Text Field): For editing the mobile number of the student.
- 4.2.6. Alternate Mobile Number (Text Field): For editing the alternate mobile number of the student.
- 4.2.7. Gender (Dropdown List): For selecting/editing the gender of the student.
- 4.2.8. Course (Dropdown List): For selecting/editing the course of the student.
- 4.2.9. Year of Admission (Read-only Text Field): Displaying the year of admission (cannot be modified).
- 4.2.10. Date of Birth (Date Picker): For selecting/editing the date of birth of the student.
- 4.2.11. Update Button: To save the modified data and update the student record.

# Outputs



Dashboard

**Student Management System** - Siddharth Jain IIMCA-4


**VIEW DATA**

Search


RollNo	FName	LName	FatherName	MobNo	AltMobNo	Gender	Course	YOA	DOB
1	Rahul	Sharma	Panikaj	978546138	978564328	Male	BCA	2019	04-06-2009
2	Ram	Mishra	Ramesh	874567887	788456887	Male	MCA	2020	22-06-2004
3	Sita	Sharma	Mahesh	978465978	897564798	Female	IIMCA	2020	06-07-2004

Update Delete

View



- Dashboard
- View
- Insert
- Project Report



## Sudent Management System


- Siddharth Jain IIMCA-4

VIEW DATA


	RollNo	FName	LName	FatherName	MobNo	AltMobNo	Gender	Course	YOA	DOB
▶	1	Rahul	Sharma	Pankaj	978546138	978564328	Male	BCA	2019	04-06-2009
✱										

Exit

Search



- Dashboard
- View
- Insert
- Project Report



## Sudent Management System

- Siddharth Jain IIMCA-4

INSERT DATA

Roll No

First Name

Last Name

Father Name

Gender

Male

Date of Birth

30 January 2002

Mob No

Alt Mob No

Course

MCA

Session

2019

Exit

Insert

- Dashboard
- View
- Insert
- Project Report

## Sudent Management System

- Siddharth Jain IIMCA-4

### EDIT DATA

Roll No

First Name

Last Name

Father Name

Gender

Male

Date of Birth

04 June 2009

Mob No

Alt Mob No

Course

BCA

Session

2019

Update

Cancel

Student Management System

Record updated successfully

OK

Exit

Update

- Dashboard
- View
- Insert
- Project Report

## Sudent Management System

- Siddharth Jain IIMCA-4

### VIEW DATA

Search

	RollNo	FName	LName	FatherName	MobNo	AltMobNo	Gender	Course	YOA	DOB
	1	Rahul	Sharma	Panikaj	978546138	978564328	Male	BCA	2019	04-06-2009
	2	Ram	Mishra	Ramesh	874567887	788456887	Male	MCA	2020	22-06-2004
	3	Sita	Sharma	Mahesh	978465979	897564799	Female	IIMCA	2020	06-07-2004

Update

Delete

Student Management System

Record deleted successfully

OK

Exit

Delete

# Source Code

## 1. DASHBOARD FORM:

```
Imports System.Data.OleDb
```

```
Public Class Form1
```

```
    Dim stReader As OleDb.OleDbDataReader
```

```
    Dim stcommand As OleDb.OleDbCommand
```

```
    Dim stdataadapter As New OleDb.OleDbDataAdapter
```

```
    Dim stDS As DataSet
```

```
    Dim rowno As Integer
```

```
    Dim reccount As Integer
```

```
    Private insertForm As Form
```

```
    Private editForm As Form
```

```
    Private viewForm As Form
```

```
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
```

```
        Module1.connectDB()
```

```
        rowno = 0
```

```
        LoadData()
```

```
    End Sub
```

```
    Public Sub LoadData()
```

```
        ' Query to get the count of male students
```

```
        Dim maleQuery As String = "SELECT COUNT(*) FROM students WHERE  
gender = 'Male'"
```

```
        stcommand = New OleDb.OleDbCommand(maleQuery, dbcon)
```

```
        Dim maleCount As Integer = CInt(stcommand.ExecuteScalar())
```

```
        ' Query to get the count of female students
```

```
        Dim femaleQuery As String = "SELECT COUNT(*) FROM students WHERE  
gender = 'Female'"
```

```
        stcommand = New OleDb.OleDbCommand(femaleQuery, dbcon)
```

```
        Dim femaleCount As Integer = CInt(stcommand.ExecuteScalar())
```

```
        ' Update the TextBoxes with the counts
```

```
        txtMaleStudents.Text = maleCount.ToString()
```

```
        txtFemaleStudents.Text = femaleCount.ToString()
```

```
        ' Query to get the count of students in BCA course
```



```

    Dim bcaQuery As String = "SELECT COUNT(*) FROM students WHERE
course = 'BCA'"
    stcommand = New OleDb.OleDbCommand(bcaQuery, dbcon)
    Dim bcaCount As Integer = CInt(stcommand.ExecuteScalar())

    ' Query to get the count of students in MCA course
    Dim mcaQuery As String = "SELECT COUNT(*) FROM students WHERE
course = 'MCA'"
    stcommand = New OleDb.OleDbCommand(mcaQuery, dbcon)
    Dim mcaCount As Integer = CInt(stcommand.ExecuteScalar())

    ' Query to get the count of students in IIMCA course
    Dim iimcaQuery As String = "SELECT COUNT(*) FROM students WHERE
course = 'IIMCA'"
    stcommand = New OleDb.OleDbCommand(iimcaQuery, dbcon)
    Dim iimcaCount As Integer = CInt(stcommand.ExecuteScalar())

    ' Update the TextBoxes with the course counts
    txtBCAStudents.Text = bcaCount.ToString()
    txtMCAStudents.Text = mcaCount.ToString()
    txtIIMCAStudents.Text = iimcaCount.ToString()

    ' Calculate the number of students in each year of their respective
courses
    Dim currentYear As Integer = 2023

    ' Query to get the count of students studying in the 1st year
    Dim firstYearQuery As String = "SELECT COUNT(*) FROM students WHERE
yoa = '" & currentYear.ToString() & "'"
    stcommand = New OleDb.OleDbCommand(firstYearQuery, dbcon)
    Dim firstYearCount As Integer = CInt(stcommand.ExecuteScalar())

    ' Query to get the count of students studying in the 2nd year
    Dim secondYearQuery As String = "SELECT COUNT(*) FROM students
WHERE yoa = '" & (currentYear - 1).ToString() & "'"
    stcommand = New OleDb.OleDbCommand(secondYearQuery, dbcon)
    Dim secondYearCount As Integer = CInt(stcommand.ExecuteScalar())

    ' Query to get the count of students studying in the 3rd year
    Dim thirdYearQuery As String = "SELECT COUNT(*) FROM students WHERE
yoa = '" & (currentYear - 2).ToString() & "'"
    stcommand = New OleDb.OleDbCommand(thirdYearQuery, dbcon)
    Dim thirdYearCount As Integer = CInt(stcommand.ExecuteScalar())

```

```

' Query to get the count of students studying in the 4th year
Dim fourthYearQuery As String = "SELECT COUNT(*) FROM students
WHERE yoa = '" & (currentYear - 3).ToString() & "'"
stcommand = New OleDb.OleDbCommand(fourthYearQuery, dbcon)
Dim fourthYearCount As Integer = CInt(stcommand.ExecuteScalar())

' Query to get the count of students studying in the 5th year
Dim fifthYearQuery As String = "SELECT COUNT(*) FROM students WHERE
yoa = '" & (currentYear - 4).ToString() & "'"
stcommand = New OleDb.OleDbCommand(fifthYearQuery, dbcon)
Dim fifthYearCount As Integer = CInt(stcommand.ExecuteScalar())

' Update the TextBoxes with the counts
txt1stYear.Text = firstYearCount.ToString()
txt2ndYear.Text = secondYearCount.ToString()
txt3rdYear.Text = thirdYearCount.ToString()
txt4thYear.Text = fourthYearCount.ToString()
txt5thYear.Text = fifthYearCount.ToString()
End Sub

Private Sub Button8_Click(sender As Object, e As EventArgs) Handles
Button8.Click
    Me.Close()
    Application.Exit()
End Sub

Private Sub btnNew_Click(sender As Object, e As EventArgs) Handles
btnNew.Click
    If insertForm Is Nothing OrElse insertForm.IsDisposed Then
        insertForm = New InsertForm()
    End If
    insertForm.TopLevel = False
    Panel5.Controls.Add(insertForm)
    insertForm.BringToFront()
    insertForm.Show()
End Sub

Private Sub btnHome_Click(sender As Object, e As EventArgs) Handles
btnHome.Click
    If insertForm IsNot Nothing AndAlso Not insertForm.IsDisposed Then
        insertForm.Hide()
    End If
End Sub

```

```

End If

If editForm IsNot Nothing AndAlso Not editForm.IsDisposed Then
    editForm.Hide()
End If

If viewForm IsNot Nothing AndAlso Not viewForm.IsDisposed Then
    viewForm.Hide()
End If

LoadData()
' Show the main panel
Panel5.Visible = True
End Sub

Private Sub btnView_Click(sender As Object, e As EventArgs) Handles
btnView.Click
    If viewForm Is Nothing OrElse viewForm.IsDisposed Then
        viewForm = New ViewForm()
    End If
    viewForm.TopLevel = False
    Panel5.Controls.Add(viewForm)
    viewForm.BringToFront()
    viewForm.Show()
    If Panel5.Controls.Contains(viewForm) Then
        DirectCast(viewForm, ViewForm).RefreshData()
    End If
End Sub

Private Sub btnReport_Click(sender As Object, e As EventArgs) Handles
btnReport.Click
    Dim url As String = "https://github.com/siddharth9300/Student-
Management-System-
VB.Net/blob/master/Siddharth%20Jain%20Project%20Report.pdf" ' Replace with
your desired web link
    System.Diagnostics.Process.Start(url)
End Sub
End Class

```

## 2. INSERT FORM:

```
Imports System.Data.OleDb
```

```
Public Class InsertForm
```

```
    Private Sub btnSave_Click(sender As Object, e As EventArgs) Handles  
btnSave.Click
```

```
        Try
```

```
            Dim saveSQL As String
```

```
            Module1.connectDB()
```

```
            saveSQL = "INSERT INTO students (rollno, fname, lname,  
fathername, mobno, altmobno, gender, course, yoa, dob) VALUES (@RollNo,  
@FName, @LName, @FatherName, @MobNo, @AltMobNo, @Gender, @Course, @yoa,  
@DOB) "
```

```
            Dim stcommand As New OleDb.OleDbCommand(saveSQL, dbcon)
```

```
            stcommand.Parameters.AddWithValue("@RollNo", txtRollNo.Text)
```

```
            stcommand.Parameters.AddWithValue("@FName", txtFirstName.Text)
```

```
            stcommand.Parameters.AddWithValue("@LName", txtLastName.Text)
```

```
            stcommand.Parameters.AddWithValue("@FatherName",
```

```
txtFatherName.Text)
```

```
            stcommand.Parameters.AddWithValue("@MobNo", txtMobNo.Text)
```

```
            stcommand.Parameters.AddWithValue("@AltMobNo",
```

```
txtAltMobNo.Text)
```

```
            stcommand.Parameters.AddWithValue("@Gender",
```

```
cmbGender.SelectedItem.ToString())
```

```
            stcommand.Parameters.AddWithValue("@Course",
```

```
cmbCourse.SelectedItem.ToString())
```

```
            stcommand.Parameters.AddWithValue("@yoa",
```

```
cmbyoa.SelectedItem.ToString())
```

```
            stcommand.Parameters.AddWithValue("@DOB", dtpDOB.Text)
```

```
            stcommand.ExecuteNonQuery()
```

```
            MsgBox("Record inserted successfully", vbInformation)
```

```
            Form1.LoadData()
```

```
            ViewForm.LoadData()
```

```
            ClearFields()
```

```
        Catch ex As Exception
```

```
            MsgBox("Error: " & ex.Message, vbExclamation)
```

```
        End Try
```

```
    End Sub
```

```
    Private Sub ClearFields()
```

```
        txtRollNo.Text = ""
```

```

        txtFirstName.Text = ""
        txtLastName.Text = ""
        txtFatherName.Text = ""
        txtMobNo.Text = ""
        txtAltMobNo.Text = ""
        cmbGender.Text = ""
        cmbCourse.Text = ""
        cmbbyoa.Text = ""
        dtpDOB.Value = Date.Now
        ' Clear or reset any other fields related to the photo upload if
applicable
    End Sub

    Private Sub btnCancel_Click(sender As Object, e As EventArgs) Handles
btnCancel.Click
        ClearFields()
        Me.Close()
    End Sub

End Class

```

### 3. VIEW FORM:

```

Imports System.Data.OleDb
Imports System.Windows.Forms.VisualStyles.VisualStyleElement.Status

Public Class ViewForm

    Private editForm As Form

    Dim stdataadapter As New OleDb.OleDbDataAdapter

    Private Sub ViewForm_Load(sender As Object, e As EventArgs) Handles
 MyBase.Load
        Module1.connectDB()
        'LoadData()
        RefreshData()
        DataGridView1.AutoGenerateColumns = True
        DataGridView1.AutoSizeColumnsMode =
DataGridViewAutoSizeColumnsMode.Fill

        Dim headerStyle As DataGridViewCellStyle = New
DataGridViewCellStyle()

```

```

        headerStyle.BackColor = Color.FromArgb(255, 192, 192, 255) ' Set
the background color of column headers
        headerStyle.ForeColor = SystemColors.WindowText ' Set the text
color of column headers
        headerStyle.SelectionBackColor = SystemColors.Highlight ' Set the
background color of selected column headers
        headerStyle.SelectionForeColor = SystemColors.HighlightText ' Set
the text color of selected column headers
        headerStyle.Font = New Font("Century Gothic", 14.25,
FontStyle.Bold) ' Set the font of column headers
        headerStyle.WrapMode = DataGridViewTriState.True ' Enable text
wrapping in column headers
        headerStyle.Alignment = DataGridViewContentAlignment.MiddleLeft '
Set the alignment of column headers
        DataGridView1.ColumnHeadersDefaultCellStyle = headerStyle

End Sub

Public Sub RefreshData()
    LoadData()
End Sub

Public Sub LoadData()
    stdataadapter = New OleDb.OleDbDataAdapter("SELECT * FROM
students", dbcon)
    Dim stDS As New DataSet
    stdataadapter.Fill(stDS, "students")
    DataGridView1.DataSource = stDS.Tables("students")
End Sub

Public Sub LoadSData(searchQuery As String)
    Dim query As String = "SELECT * FROM students"
    If Not String.IsNullOrEmpty(searchQuery) Then
        query += " WHERE rollno = '" + searchQuery + "' OR fname LIKE
'" + searchQuery + "%'"
    End If
    stdataadapter = New OleDb.OleDbDataAdapter(query, dbcon)
    Dim stDS As New DataSet
    stdataadapter.Fill(stDS, "students")
    DataGridView1.DataSource = stDS.Tables("students")
End Sub

```

```

Private Sub btnUpdate_Click(sender As Object, e As EventArgs) Handles
btnUpdate.Click
    If DataGridView1.SelectedRows.Count > 0 Then
        Dim rollNo As String =
DataGridView1.SelectedRows(0).Cells("rollno").Value.ToString()
        If editForm Is Nothing OrElse editForm.IsDisposed Then
            editForm = New EditForm(rollNo)
        Else
            DirectCast(editForm, EditForm).UpdateRollNo(rollNo)
        End If
        editForm.TopLevel = False
        Form1.Panel15.Controls.Add(editForm)
        editForm.BringToFront()
        editForm.Show()
        AddHandler editForm.FormClosed, AddressOf EditFormClosed '
Attach event handler for EditForm closing
    Else
        MsgBox("Please select a row to edit", vbExclamation)
    End If
End Sub

Private Sub btnDelete_Click(sender As Object, e As EventArgs) Handles
btnDelete.Click
    If DataGridView1.SelectedRows.Count > 0 Then
        Dim selectedRow As DataGridViewRow =
DataGridView1.SelectedRows(0)
        Dim rollNo As String =
selectedRow.Cells("rollno").Value.ToString()
        If MessageBox.Show("Are you sure you want to delete this
record?", "Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Question)
= DialogResult.Yes Then
            Module1.connectDB()
            Dim delquery As String = "DELETE FROM students WHERE
rollno=@rollno"
            Dim stcommand As New OleDb.OleDbCommand(delquery, dbcon)
            stcommand.Parameters.AddWithValue("@rollno", rollNo)
            stcommand.ExecuteNonQuery()
            dbcon.Close()
            MsgBox("Record deleted successfully", vbInformation)

            LoadData() ' Reload the data after deleting the record
        End If
    Else

```

```

        MsgBox("Please select a row to delete", vbExclamation)
    End If
End Sub

Private Sub btnSearch_Click(sender As Object, e As EventArgs) Handles
btnSearch.Click
    Dim searchQuery As String = txtSearch.Text.Trim()
    ' Call the LoadData method with the search query
    LoadSData(searchQuery)
End Sub

Private Sub EditFormClosed(sender As Object, e As FormClosedEventArgs)
    LoadData() ' Reload the data after closing the EditForm
End Sub

End Class

```

#### 4. EDIT FORM:

```

Imports System.Data.OleDb
Imports System.Windows.Forms.VisualStyles.VisualStyleElement.Status

Public Class EditForm
    Private rollNo As String

    Public Sub New(rollNo As String)
        InitializeComponent()
        Me.rollNo = rollNo
        LoadData()
    End Sub

    Private Sub LoadData()
        Dim selectSQL As String = "SELECT * FROM students WHERE rollno =
@rollno"

        Module1.connectDB()
        Dim stcommand As New OleDb.OleDbCommand(selectSQL, dbcon)
        stcommand.Parameters.AddWithValue("@rollno", rollNo)
        Dim stReader As OleDb.OleDbDataReader = stcommand.ExecuteReader()
        If stReader.Read() Then
            txtRollNo.Text = stReader("rollno").ToString()
            txtFirstName.Text = stReader("fname").ToString()
            txtLastName.Text = stReader("lname").ToString()
            txtFatherName.Text = stReader("fathername").ToString()
        End If
    End Sub
End Class

```



```

        txtMobNo.Text = stReader("mobno").ToString()
        txtAltMobNo.Text = stReader("altmobno").ToString()
        cmbGender.SelectedItem = stReader("gender").ToString()
        cmbCourse.SelectedItem = stReader("course").ToString()
        cmbyoa.SelectedItem = stReader("yoa").ToString()
        dtpDOB.Value = DateTime.Parse(stReader("dob").ToString())
    End If
    stReader.Close()
End Sub

Public Sub UpdateRollNo(rollNo As String)
    Me.rollNo = rollNo
    LoadData()
End Sub

Private Sub btnCancel_Click(sender As Object, e As EventArgs) Handles
btnCancel.Click
    Me.Close()
End Sub

Private Sub btnSave_Click(sender As Object, e As EventArgs) Handles
btnSave.Click
    Try
        Dim updateSQL As String = "UPDATE students SET fname = @fname,
lname = @lname, fathername = @fathername, mobno = @mobno, altmobno =
@altmobno, gender = @gender, course = @course, yoa = @yoa, dob = @dob WHERE
rollno = @rollno"

        Module1.connectDB()
        Dim stcommand As New OleDb.OleDbCommand(updateSQL, dbcon)
        stcommand.Parameters.AddWithValue("@fname", txtFirstName.Text)
        stcommand.Parameters.AddWithValue("@lname", txtLastName.Text)
        stcommand.Parameters.AddWithValue("@fathername",
txtFatherName.Text)
        stcommand.Parameters.AddWithValue("@mobno", txtMobNo.Text)
        stcommand.Parameters.AddWithValue("@altmobno",
txtAltMobNo.Text)
        stcommand.Parameters.AddWithValue("@gender",
cmbGender.SelectedItem.ToString())
        stcommand.Parameters.AddWithValue("@course",
cmbCourse.SelectedItem.ToString())
        stcommand.Parameters.AddWithValue("@yoa",
cmbyoa.SelectedItem.ToString())
        stcommand.Parameters.AddWithValue("@dob", dtpDOB.Text)
    
```

```
        stcommand.Parameters.AddWithValue("@rollno", rollNo)
        stcommand.ExecuteNonQuery()
        MsgBox("Record updated successfully", vbInformation)
        Form1.LoadData()
        Me.Close()
    Catch ex As Exception
        MsgBox("Error: " & ex.Message, vbExclamation)
    End Try
End Sub

End Class
```

# Conclusion

The Student Management System project has successfully achieved its objectives of efficient student data management and streamlined processes. Through the implementation of various modules and functionalities, the system has provided a user-friendly interface for managing student records, performing CRUD operations, and generating insightful reports.

During the development and testing phases, we encountered and addressed several challenges, ensuring the system's functionality and reliability. The feedback received from users during the evaluation phase has been invaluable in making improvements and enhancements to the system, resulting in a more user-centric and efficient solution.

The Student Management System has proven to be an effective tool for educational institutions, enabling them to maintain accurate student information, monitor student enrollment and progress, and generate meaningful reports. The system's dashboard provides a comprehensive overview of key statistics, such as the number of male and female students, the distribution of students across courses, and the count of students in each academic year.

Overall, the implementation of the Student Management System has yielded positive results, simplifying administrative tasks, improving data accuracy, and enhancing decision-making processes. The system has demonstrated its potential for further expansion and customization, allowing for additional features and functionalities to be integrated in the future.

In conclusion, the Student Management System has emerged as a valuable asset for educational institutions, contributing to efficient student data management, improved workflow, and enhanced user experience. With the continued support and feedback from users, the system can be further enhanced to meet evolving needs and serve as a cornerstone for effective student administration.

# Bibliography

During the development of the Student Management System project, the following resources were referenced and utilized:

## 1. WEBSITES

- 1.1. <https://chat.openai.com/>
- 1.2. <https://stackoverflow.com/>
- 1.3. <https://www.youtube.com/>
- 1.4. <https://www.canva.com/>
- 1.5. <https://feathericons.com/>
- 1.6. <https://fonts.google.com/icons>
- 1.7. <https://www.freepik.com/>
- 1.8. <https://fontawesome.com/>

## 2. APPLICATIONS

- 2.1. Visual Studio 2022