SYNOPSIS REPORT

Title of the Project: Student Management System

Submitted by:

Name: Surenjit Kumar

Roll no.: 65

Section: A

Department: SOET

Course: B.Tech CSE (AI / ML)

University: D.Y. Patil University Ambi, Pune

Submitted To: Mr. Manaan Rasool Sir

Sr No.	Table of Contents
1	Introduction
2	Methodology
2.1	Tools Used
2.2	Functions Used
2.3	Classes Used
2.4	Methods Implemented
2.5	System Flowchart
2.6	Architecture Diagram
3	Future Scope
4	Conclusion

1.**Introduction**

My Project is based on the Student Management System and this is created by python and csv. This project stored student's data digitally and it is easier to add, search, view, update and delete data of students easily without any delay or any issues. And it saves data of students in a csv file.The system allows to add new students along with their subject marks then it automatically calculates average marks and gives the grade of students based on their average marks. Also view all students details at one place in csv file and search a specific student using their roll number, and update student data when filled with wrong data, and delete a student when needs.By using Python concepts like classes, objects, file handling, exception handling, and conditional statements like if-elif-else.

2. **Methodology**

2.1 **Tools Used:**

In this project, i used many tools in my project:

- Python 3.13.9
- Visual Studio Code: For Edit and writing the code.
- CSV module: For save student's details.
- File Handling: For write, read, append in the code.
- OPP: For creating classes objects.
- Exception File Handling: try and except.
- Loop
- Conditional Statements.
- Input: Taking input from the user.

2.2 <u>Functions Used:</u>

I used many Common Functions Such as:

- Constructor Function: __init__() it stores name, roll number, class, section, marks of all seven subjects and calculates average marks and gives grade basis on average marks.
- AddStudent(): This function adds new students.
- ViewStudent(): This function prints the complete list of all students.
- SearchStudent(): This function searches a specific student using their roll number.
- DeleteStudent(): This function deletes student data based on their roll number.
- The While Loop: This loop shows the options until when the user chooses.
- Grade(): This function calculates the student grade based on their average marks.
- Input Handling.
- Data Processing.
- File Operations.
- Calculation.
- Valid Function

2.3 <u>Classes Used:</u>

I used mainly used two classes for this project:

- Student Class.
- Choices Class.

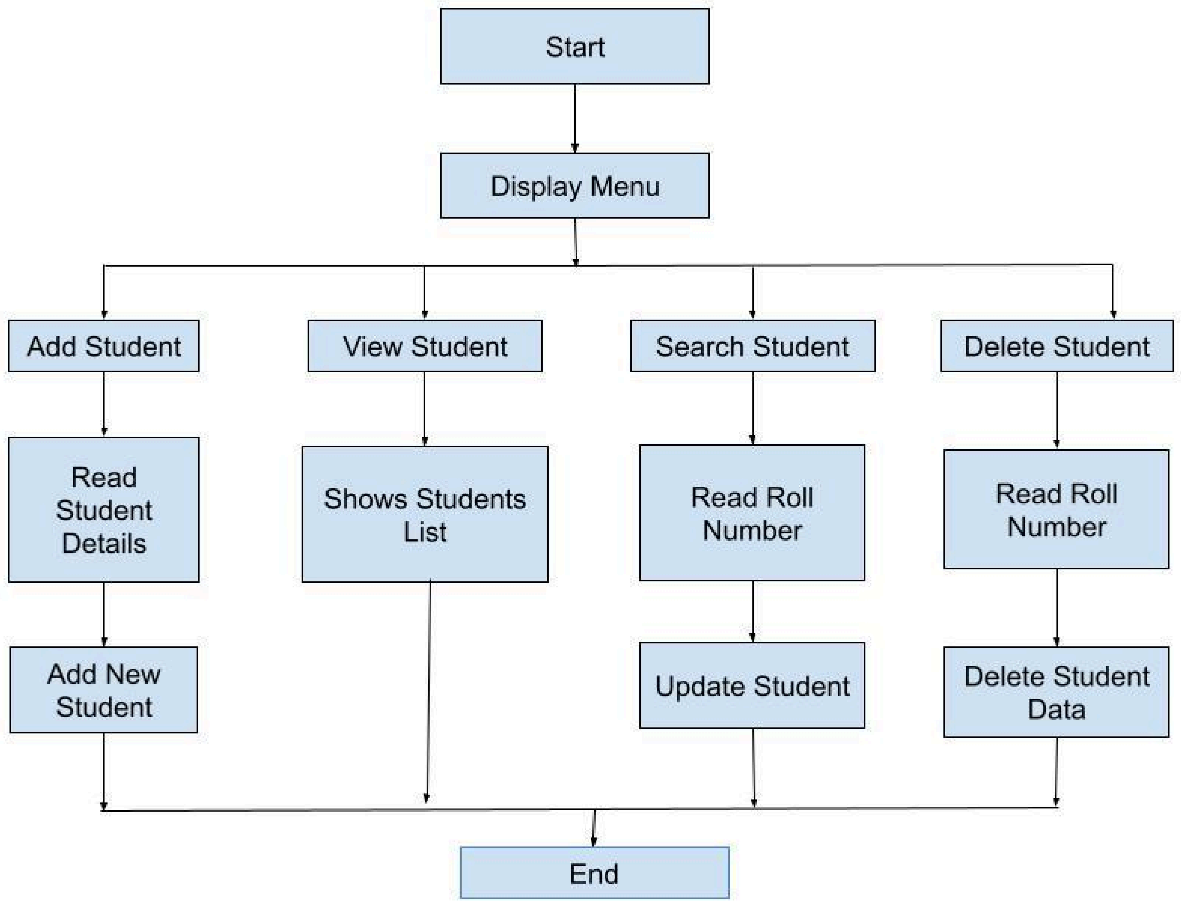
2.4 <u>Methods Implemented:</u>

I used many types of methods:

- Constructor method.
- Input method.
- Read, write CSV file method.
- Grade method.
- File Handling method.
- Error Handling Method.
- Validation Method.

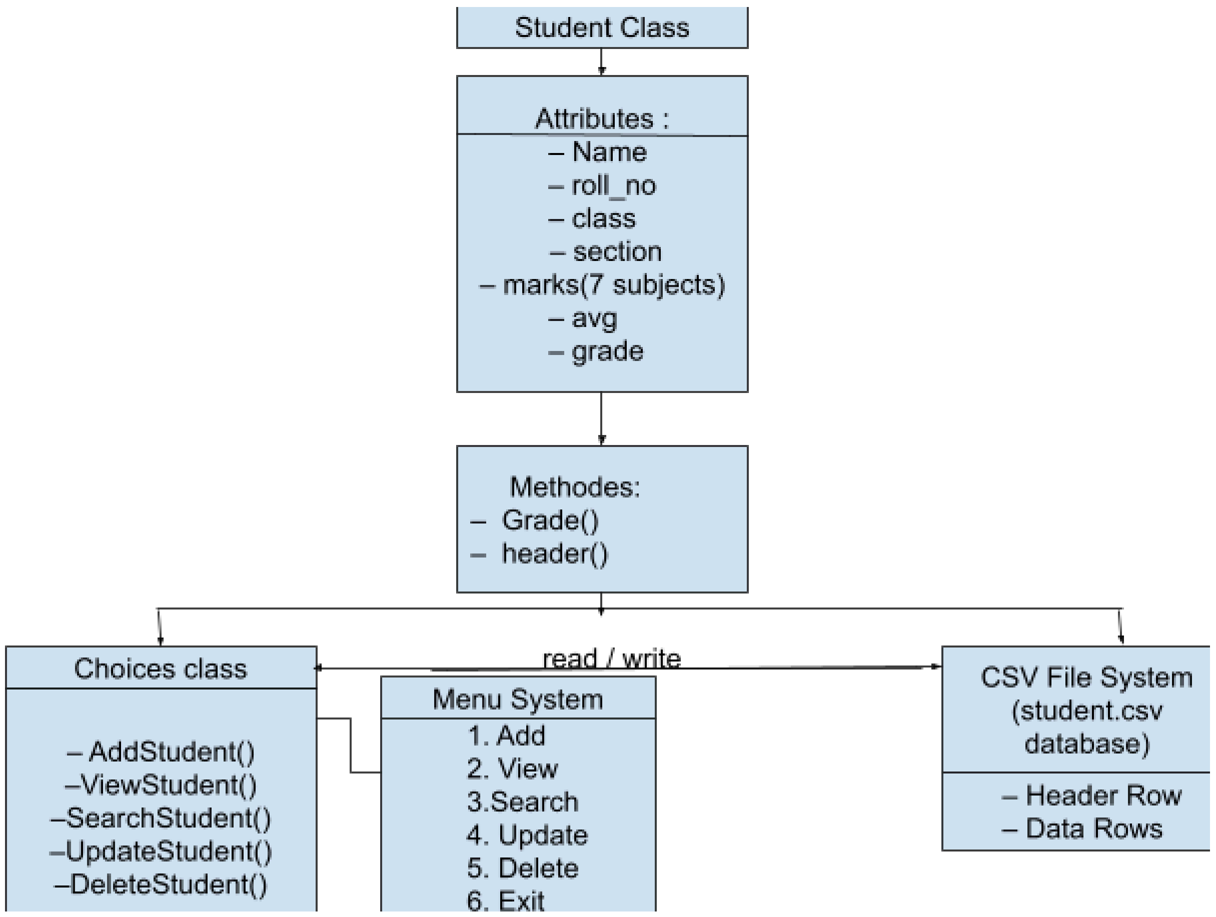
2.5 <u>System Flowchart</u>

The flow chart of my project is:

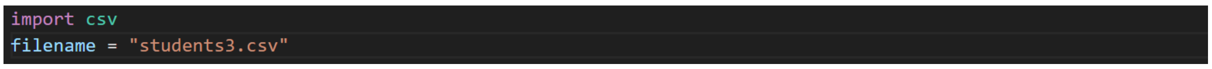


2.6 **Architecture Diagram**

The Architecture Diagram of my project:

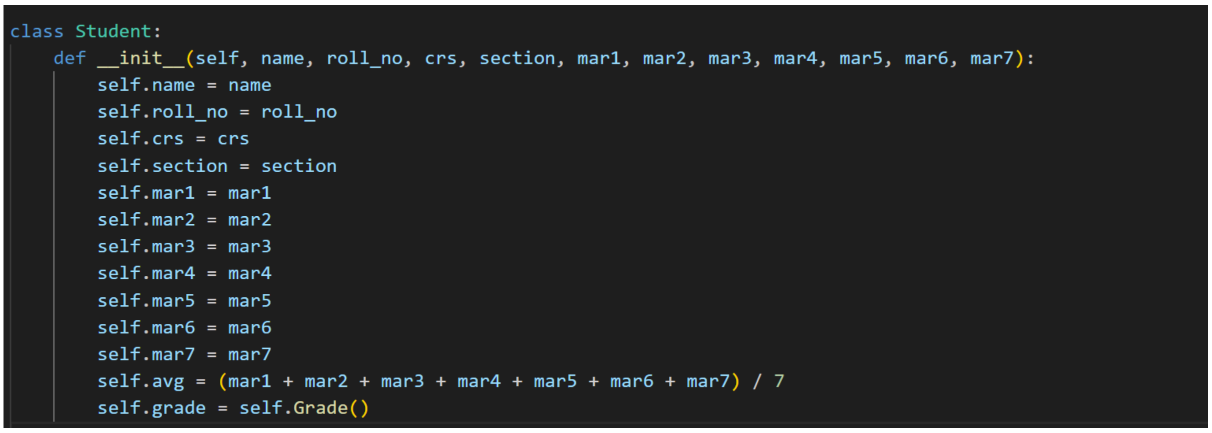


- Firstly, I'm importing CSV Files. csv file save all data of student:



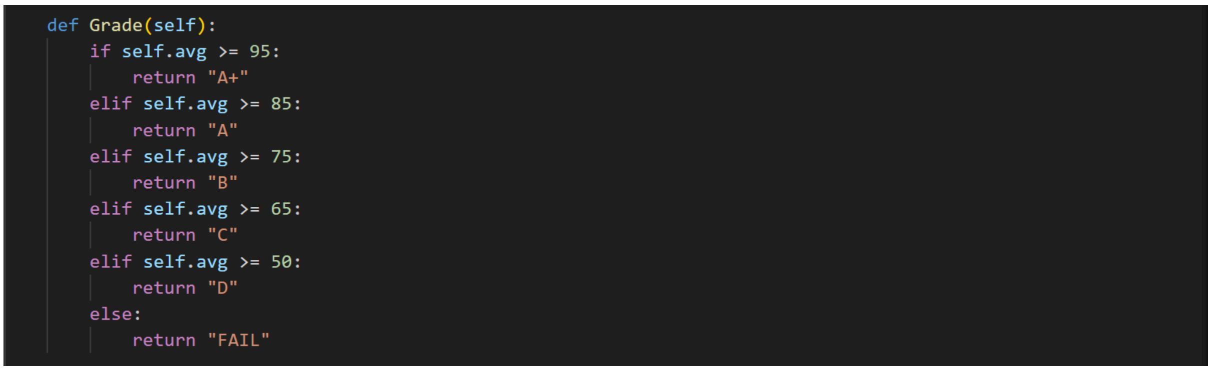
```
import csv
filename = "students3.csv"
```

- Define Student class and this is the main class and others functions or methods(__init__) write inside this class:



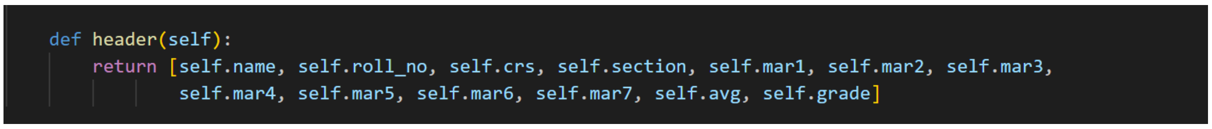
```python
class Student:
    def __init__(self, name, roll_no, crs, section, mar1, mar2, mar3, mar4, mar5, mar6, mar7):
        self.name = name
        self.roll_no = roll_no
        self.crs = crs
        self.section = section
        self.mar1 = mar1
        self.mar2 = mar2
        self.mar3 = mar3
        self.mar4 = mar4
        self.mar5 = mar5
        self.mar6 = mar6
        self.mar7 = mar7
        self.avg = (mar1 + mar2 + mar3 + mar4 + mar5 + mar6 + mar7) / 7
        self.grade = self.Grade()
```

- Define Grade Function: This function writes the student’s grade by using conditional statements and the student's grade given based on their average marks.



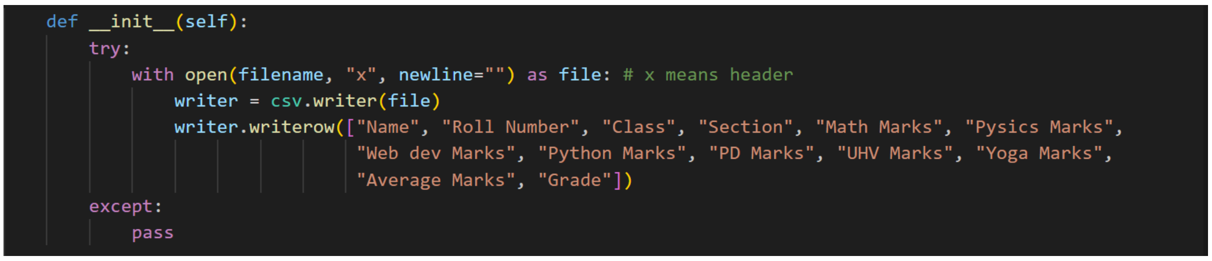
```python
    def Grade(self):
        if self.avg >= 95:
            return "A+"
        elif self.avg >= 85:
            return "A"
        elif self.avg >= 75:
            return "B"
        elif self.avg >= 65:
            return "C"
        elif self.avg >= 50:
            return "D"
        else:
            return "FAIL"
```

- Define header Function: The header function writes the data of students in the header of csv file.



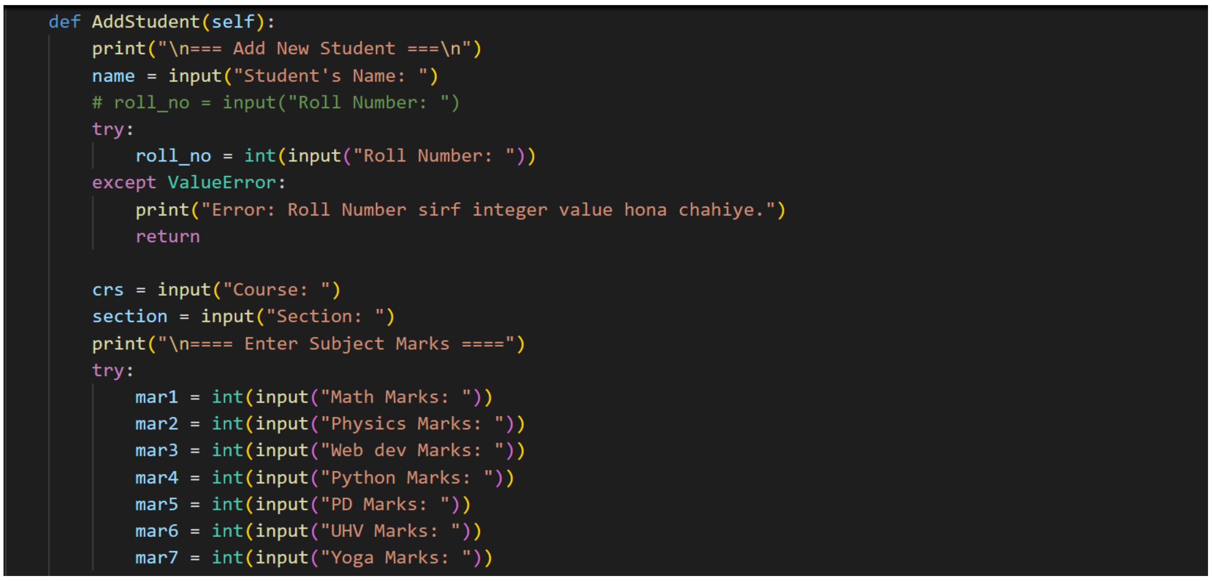
```python
    def header(self):
        return [self.name, self.roll_no, self.crs, self.section, self.mar1, self.mar2, self.mar3,
                self.mar4, self.mar5, self.mar6, self.mar7, self.avg, self.grade]
```

- Constructive method: In constructive method i add, view, search, update, delete and exiting the student;s data and i’m writing header name of student’s data in csv file with help of file handling and try except. Because of the understanding and for better visualization of the student’s data.

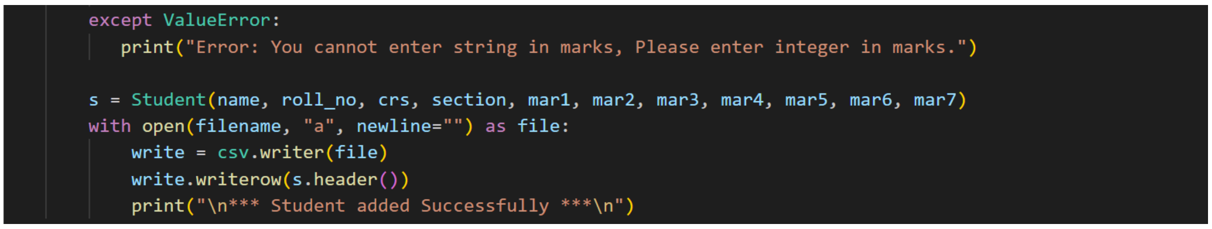


```python
    def __init__(self):
        try:
            with open(filename, "x", newline="") as file: # x means header
                writer = csv.writer(file)
                writer.writerow(["Name", "Roll Number", "Class", "Section", "Math Marks", "Pysics Marks",
                                 "Web dev Marks", "Python Marks", "PD Marks", "UHV Marks", "Yoga Marks",
                                 "Average Marks", "Grade"])
        except:
            pass
```

- AddStudent function: AddStudent function is defined for adding new students and their data with the help of file handling and data save in csv file.

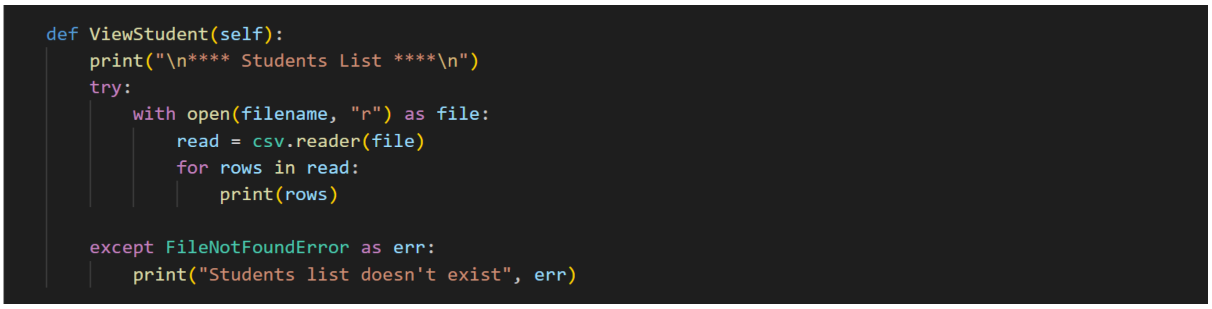


```python
def AddStudent(self):
    print("\n=== Add New Student ===\n")
    name = input("Student's Name: ")
    # roll_no = input("Roll Number: ")
    try:
        roll_no = int(input("Roll Number: "))
    except ValueError:
        print("Error: Roll Number sirf integer value hona chahiye.")
        return

    crs = input("Course: ")
    section = input("Section: ")
    print("\n==== Enter Subject Marks ====")
    try:
        mar1 = int(input("Math Marks: "))
        mar2 = int(input("Physics Marks: "))
        mar3 = int(input("Web dev Marks: "))
        mar4 = int(input("Python Marks: "))
        mar5 = int(input("PD Marks: "))
        mar6 = int(input("UHV Marks: "))
        mar7 = int(input("Yoga Marks: "))
```



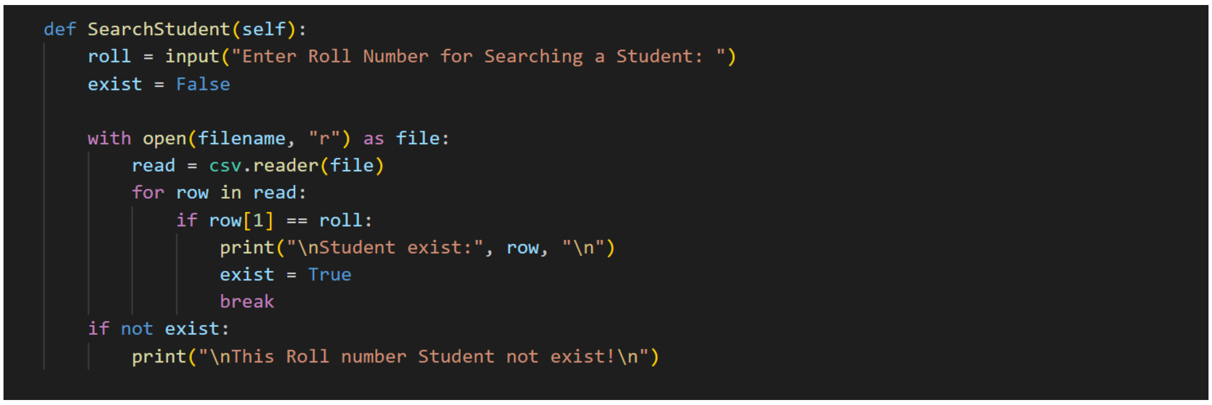
```python
    except ValueError:
        print("Error: You cannot enter string in marks, Please enter integer in marks.")

    s = Student(name, roll_no, crs, section, mar1, mar2, mar3, mar4, mar5, mar6, mar7)
    with open(filename, "a", newline="") as file:
        write = csv.writer(file)
        write.writerow(s.header())
        print("\n*** Student added Successfully ***\n")
```

- ViewStudent function: The viewstudent function is for displaying all student lists in one place with help of csv file reading.



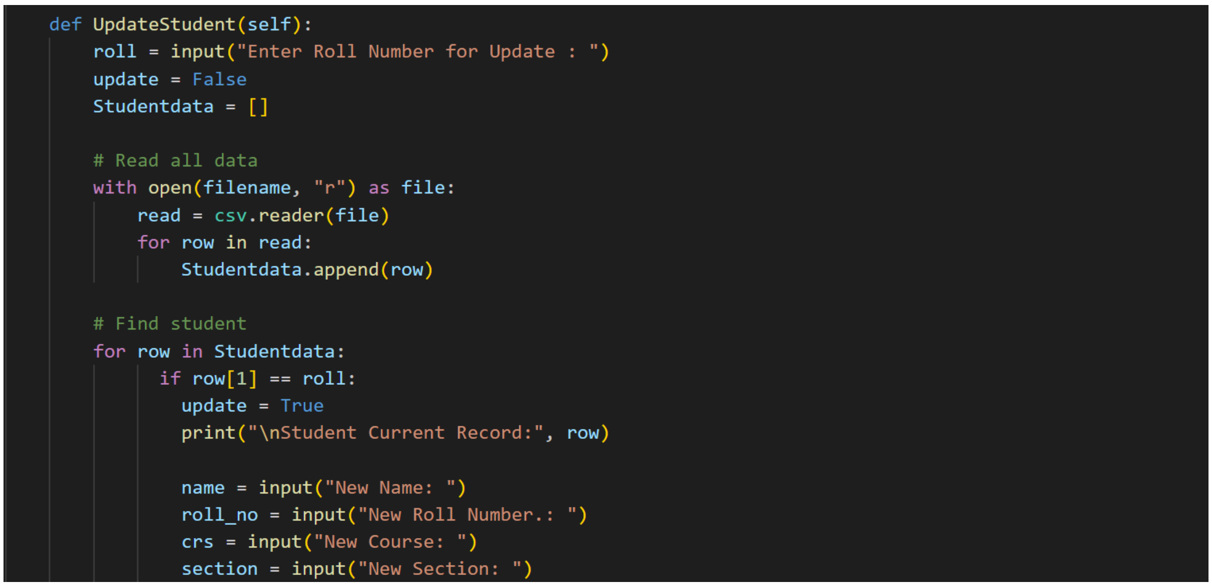
```python
def ViewStudent(self):
    print("\n**** Students List ****\n")
    try:
        with open(filename, "r") as file:
            read = csv.reader(file)
            for rows in read:
                print(rows)

    except FileNotFoundError as err:
        print("Students list doesn't exist", err)
```

- SearchStudent Function: This function is used for searching a specific student by their roll number and displaying the student with help of file handling.

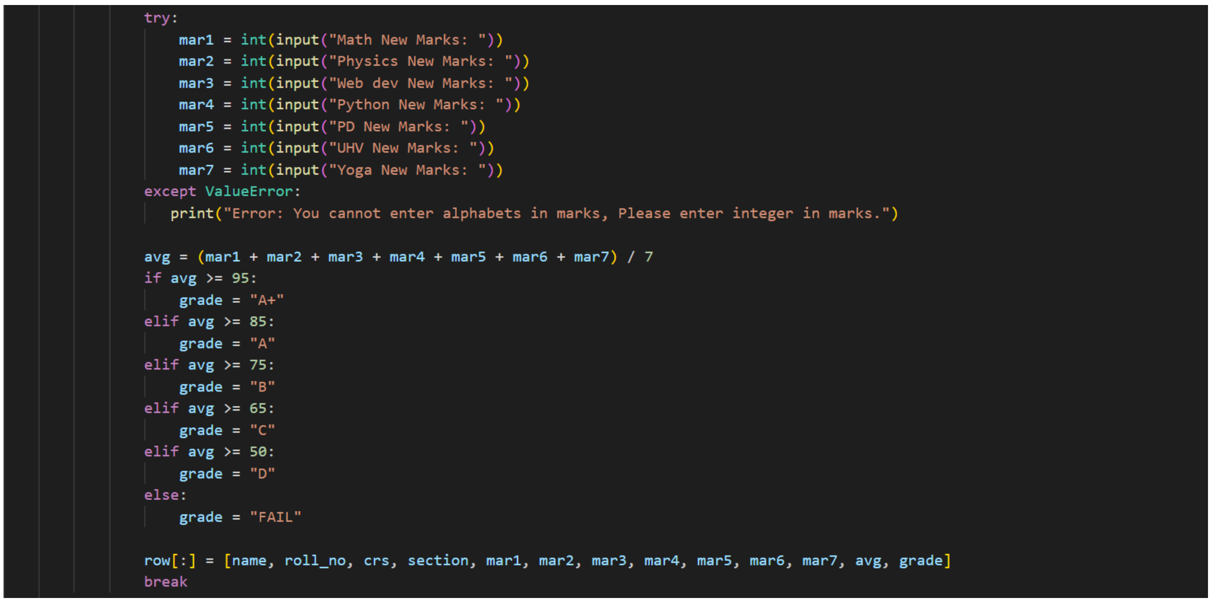


```python
def SearchStudent(self):
    roll = input("Enter Roll Number for Searching a Student: ")
    exist = False

    with open(filename, "r") as file:
        read = csv.reader(file)
        for row in read:
            if row[1] == roll:
                print("\nStudent exist:", row, "\n")
                exist = True
                break
    if not exist:
        print("\nThis Roll number Student not exist!\n")
```

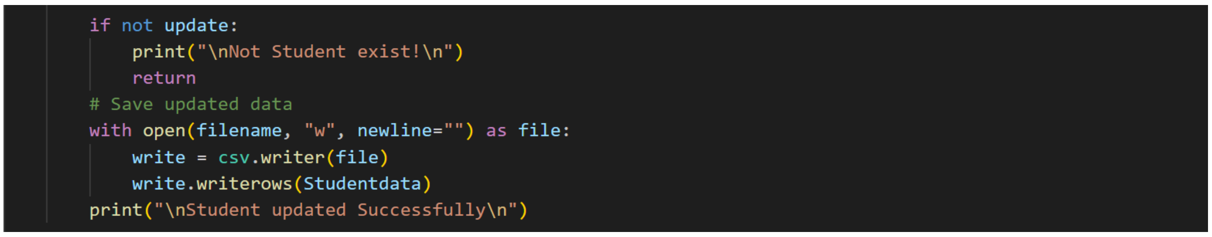
- UpdateStudent Function: Update function is used for updating the student data very easily and without any delaying or issue and if any specific student’s data is wrong then we use update function for improving the wrong data.



```python
def UpdateStudent(self):
    roll = input("Enter Roll Number for Update : ")
    update = False
    Studentdata = []

    # Read all data
    with open(filename, "r") as file:
        read = csv.reader(file)
        for row in read:
            Studentdata.append(row)

    # Find student
    for row in Studentdata:
        if row[1] == roll:
            update = True
            print("\nStudent Current Record:", row)

            name = input("New Name: ")
            roll_no = input("New Roll Number.: ")
            crs = input("New Course: ")
            section = input("New Section: ")
```

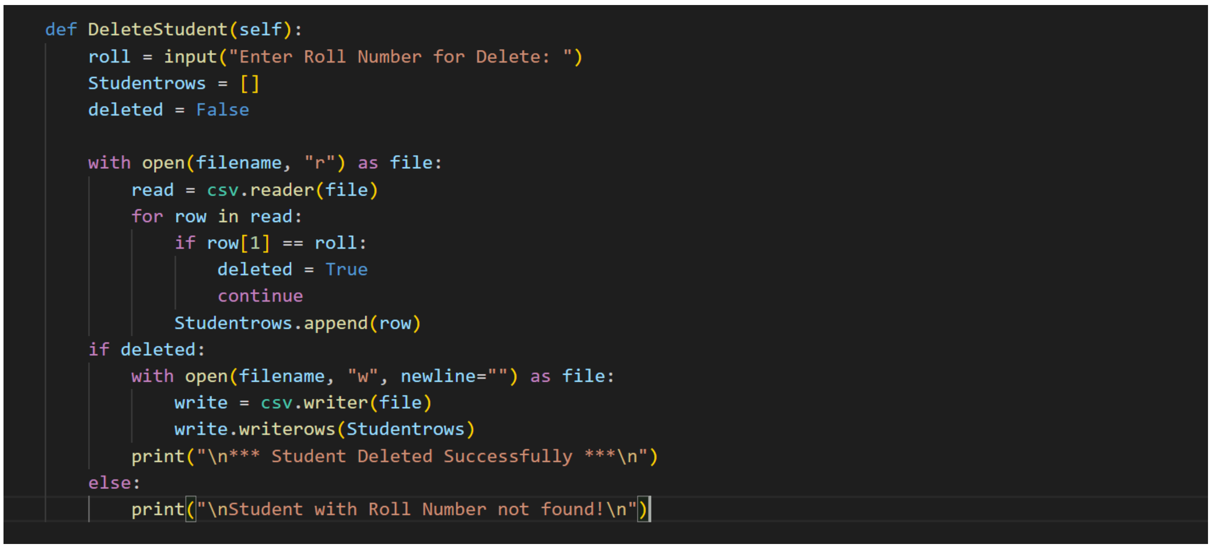


```
                try:
                    mar1 = int(input("Math New Marks: "))
                    mar2 = int(input("Physics New Marks: "))
                    mar3 = int(input("Web dev New Marks: "))
                    mar4 = int(input("Python New Marks: "))
                    mar5 = int(input("PD New Marks: "))
                    mar6 = int(input("UHV New Marks: "))
                    mar7 = int(input("Yoga New Marks: "))
                except ValueError:
                    print("Error: You cannot enter alphabets in marks, Please enter integer in marks.")

                avg = (mar1 + mar2 + mar3 + mar4 + mar5 + mar6 + mar7) / 7
                if avg >= 95:
                    grade = "A+"
                elif avg >= 85:
                    grade = "A"
                elif avg >= 75:
                    grade = "B"
                elif avg >= 65:
                    grade = "C"
                elif avg >= 50:
                    grade = "D"
                else:
                    grade = "FAIL"

                row[:] = [name, roll_no, crs, section, mar1, mar2, mar3, mar4, mar5, mar6, mar7, avg, grade]
                break
```



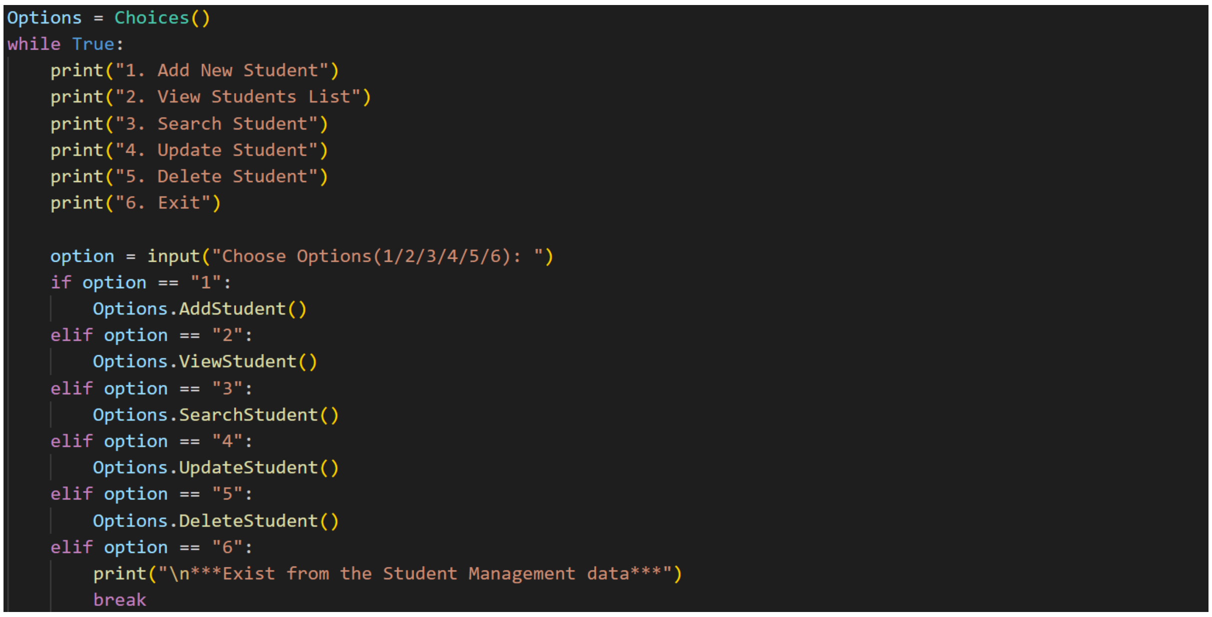
```
        if not update:
            print("\nNot Student exist!\n")
            return
        # Save updated data
        with open(filename, "w", newline="") as file:
            write = csv.writer(file)
            write.writerows(Studentdata)
        print("\nStudent updated Successfully\n")
```

- DeleteStudent Function: This function is used for deleting the student from the system with the help of file handling and conditional statements.

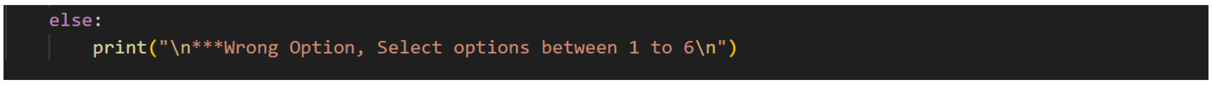


```
    def DeleteStudent(self):
        roll = input("Enter Roll Number for Delete: ")
        Studentrows = []
        deleted = False

        with open(filename, "r") as file:
            read = csv.reader(file)
            for row in read:
                if row[1] == roll:
                    deleted = True
                    continue
                Studentrows.append(row)
        if deleted:
            with open(filename, "w", newline="") as file:
                write = csv.writer(file)
                write.writerows(Studentrows)
            print("\n*** Student Deleted Successfully ***\n")
        else:
            print("\nStudent with Roll Number not found!\n")
```

- Conditional statements on choosing options: This is used for choosing operations based on numbers that are 1, 2, 3, 4, 5, 6 by help of if-elif-else statements.



```python
Options = Choices()
while True:
    print("1. Add New Student")
    print("2. View Students List")
    print("3. Search Student")
    print("4. Update Student")
    print("5. Delete Student")
    print("6. Exit")

    option = input("Choose Options(1/2/3/4/5/6): ")
    if option == "1":
        Options.AddStudent()
    elif option == "2":
        Options.ViewStudent()
    elif option == "3":
        Options.SearchStudent()
    elif option == "4":
        Options.UpdateStudent()
    elif option == "5":
        Options.DeleteStudent()
    elif option == "6":
        print("\n***Exist from the Student Management data***")
        break
```



```python
    else:
        print("\n***Wrong Option, Select options between 1 to 6\n")
```

3. Future Scope

In future i will add many thing such as:

- Add GUI Support.
- Integrate database.
- Student login System
- Data analytic of students.
- Add mobile number of students and their parents for sending performance of students via email, message.
- Add Web extension.

4. Conclusion

The Conclusion of my Project is how data can store, delete, Updated, searches in an easier manner. By using classes, objects, constructors, file handling, csv for writing, reading and save students data, exception handling for handle the error. This system is also calculate the average marks of students and gives a grade basis on the average marks. This project is simple & it helps to stored students data in a clean and automated way, making it easy to operate.