## 2. Database Descriptions:-

The database used in the project is CSV files.

A CSV (Comma Separated Value) file is a type of plain text file that uses specific structure to arrange tabular data. Because its a plain text file, it can contain only actual text data—in other words, printable ASCII or Unicode characters. The structure of a CSV file is given away by its name. Normally, CSV files use a comma to separate each specific data value.

**2.1 Database Samples:**

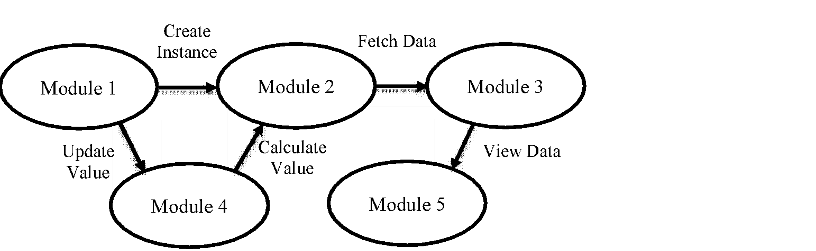
CSV sample of 1st module:--

### Students.csv

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Serial Number | Student ID | Name | Roll | Stream:- |  |
| 1 | IEM-01 | Debdatta Basak | 1 | IT |  |
| 2 | IEM-02 | Som Sen | 2 | CSE |  |
| 3 | IEM-03 | Raj Bhandari | 3 | EE |  |
| 4 | IEM-04 | Sridatri Banerjee | 4 | ME |  |
| 5 | IEM-05 | Soham Roy | 5 | CSE |  |
| 6 | IEM-06 | Sounak Bose | 6 | CSE |  |
| 7 | IEM-07 | Saswata Guha | 7 | EE |  |
| 8 | IEM-08 | Sanket Sinha | 8 | ME |  |
| 9 | IEM-09 | Subhayu Roy | 9 | IT |  |
| 10 | IEM-10 | Arya Sengupta | 10 | EE |  |
| 11 | IEM-11 | Ari Chowdhury | 11 | ME |  |

## 3. Data Flow and E-R Diagrams:-

Demonstrates the dependency of all the python modules written using a data flow diagram.



## 4. Programs:-

Python program of 1st module:--

### Students Examination Portal.py

|  |
| --- |
| import csv student\_fields = ['Student ID', 'Name', 'Class Roll Number', 'Batch Name'] student\_database = 'students.csv'  def display\_menu():  print("---------------------------") print("Student Examination Portal.") print("---------------------------") print("1.Add New Student") print("2.View Students") print("3.Update Student") print("4.Delete Student") print("5.Calculate Grade") print("6.Quit")    def add\_student():  print("Add Student Information:") print("------------------------") global student\_fields global student\_database student\_data = [] for field in student\_fields: value = input("Enter " + field + ": ") student\_data.append(value) with open(student\_database, "a", encoding="utf-8") as f:  writer = csv.writer(f) writer.writerows([student\_data]) print("Data saved successfully!") input("Press enter to continue.")  def view\_students(): |

|  |
| --- |
| global student\_fields global student\_database print("--- Student Records: ---") print("------------------------") with open(student\_database, "r", encoding="utf-8") as f:  reader = csv.reader(f) for x in student\_fields: print(x, end='\t|') print("\n-------------------------------------------------------------  -----") for row in reader: for item in row:  print(item, end="\t|") print("\n") input("Press enter to continue.")    def update\_student():  global student\_fields global student\_database print("--- Update Student: ---") print("-----------------------") roll = input("Enter Student ID to update: ") index\_student = None updated\_data = [] with open(student\_database, "r", encoding="utf-8") as f:  reader = csv.reader(f) counter = 0 for row in reader: if len(row) > 0: if roll == row[0]:  index\_student = counter print("Student found at index ", index\_student) student\_data = [] for field in student\_fields: value = input("Enter " + field + ": ") student\_data.append(value) updated\_data.append(student\_data) else:  updated\_data.append(row) counter += 1 if index\_student is not None: with open(student\_database, "w", encoding="utf-8") as f:  writer = csv.writer(f) writer.writerows(updated\_data) print("Student ID", roll, "updated successfully!") else:  print("Student ID not found in our database!") |

|  |
| --- |
| input("Press enter to continue.")    def delete\_student():  global student\_fields global student\_database print("--- Delete Student: ---") print("-----------------------") roll = input("Enter Student ID to delete: ") student\_found = False updated\_data = [] with open(student\_database, "r", encoding="utf-8") as f:  reader = csv.reader(f) counter = 0 for row in reader: if len(row) > 0: if roll != row[0]: updated\_data.append(row) counter += 1 else:  student\_found = True if student\_found is True: with open(student\_database, "w", encoding="utf-8") as f:  writer = csv.writer(f) writer.writerows(updated\_data) print("Student ID", roll, "deleted successfully!") else:  print("Student ID not found in our database!") input("Press enter to continue.")    def search\_student():  global student\_fields global student\_database print("--- Search Student: ---") print("-----------------------") roll = input("Enter Student ID to search: ") with open(student\_database, "r", encoding="utf-8") as f:  reader = csv.reader(f) for row in reader: if len(row) > 0: if roll == row[0]:  print("Student found with the following details...") print("Student ID: ", row[0]) print("Name: ", row[1]) print("Class Roll Number: ", row[2]) print("Batch Name: ", row[3]) grade() break |

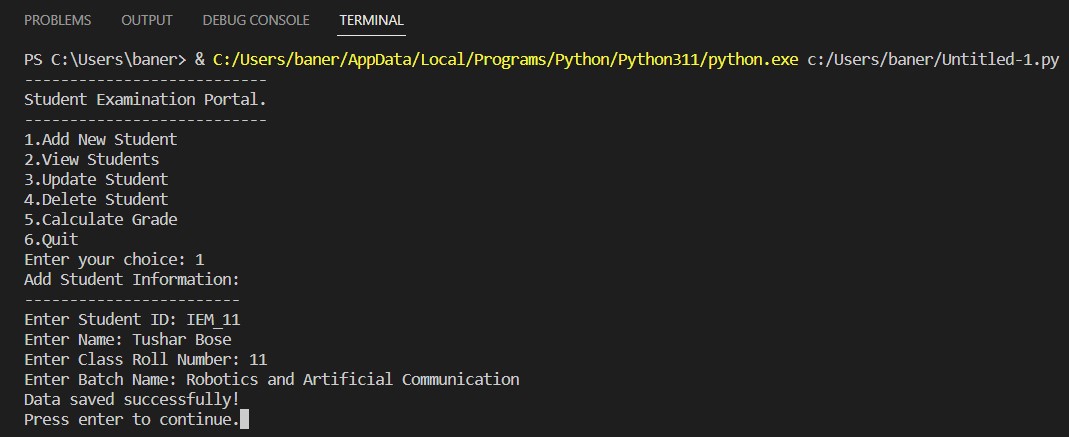
|  |
| --- |
| else:  print("Student ID not found in our database!") input("Press enter to continue.")    def grade():  print("Enter marks out of 100:") m1 = int(input("Enter marks in 1st subject: ")) m2 = int(input("Enter marks in 2nd subject: ")) m3 = int(input("Enter marks in 3rd subject: ")) m4 = int(input("Enter marks in 4th subject: ")) m5 = int(input("Enter marks in 5th subject: ")) tmarks = m1+m2+m3+m4+m5 per = (tmarks)//5 if per >= 90:  print("Total marks = ", tmarks, "\nPercentage = ", per, "\nGrade= A\nStatus: Passed!") elif per >= 80 and per < 90:  print("Total marks = ", tmarks, "\nPercentage = ", per, "\nGrade= B\nStatus: Passed!") elif per >= 70 and per < 80:  print("Total marks = ", tmarks, "\nPercentage = ", per, "\nGrade= C\nStatus: Passed!") elif per >= 60 and per < 70:  print("Total marks = ", tmarks, "\nPercentage = ", per, "\nGrade= D\nStatus: Passed!") elif per >= 50 and per < 60:  print("Total marks = ", tmarks, "\nPercentage = ", per, "\nGrade= E\nStatus: Passed!") else:  print("Total marks = ", tmarks, "\nPercentage = ", per, "\nGrade= F\nStatus: Failed!")    while True:  display\_menu() choice = input("Enter your choice: ") if choice == '1': add\_student() elif choice == '2': view\_students() elif choice == '3': update\_student() elif choice == '4': delete\_student() elif choice == '5': search\_student() else:  break |

print("-------------------------------") print("Thank you for using our system.") print("-------------------------------")

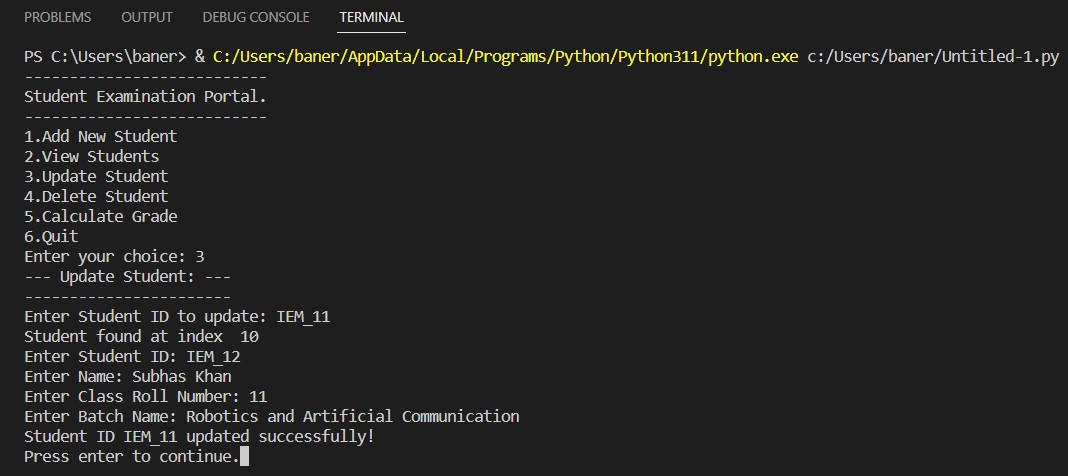
## 5. Outputs:-

Sample outputs(screenshot) to demonstrate the functionalities in programs.

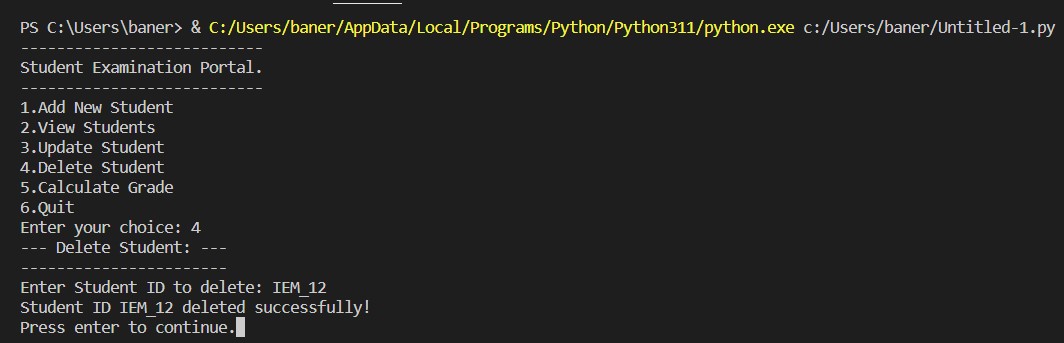
1. Creating a student using Student ID, Name, Class Roll Number and Batch Name.



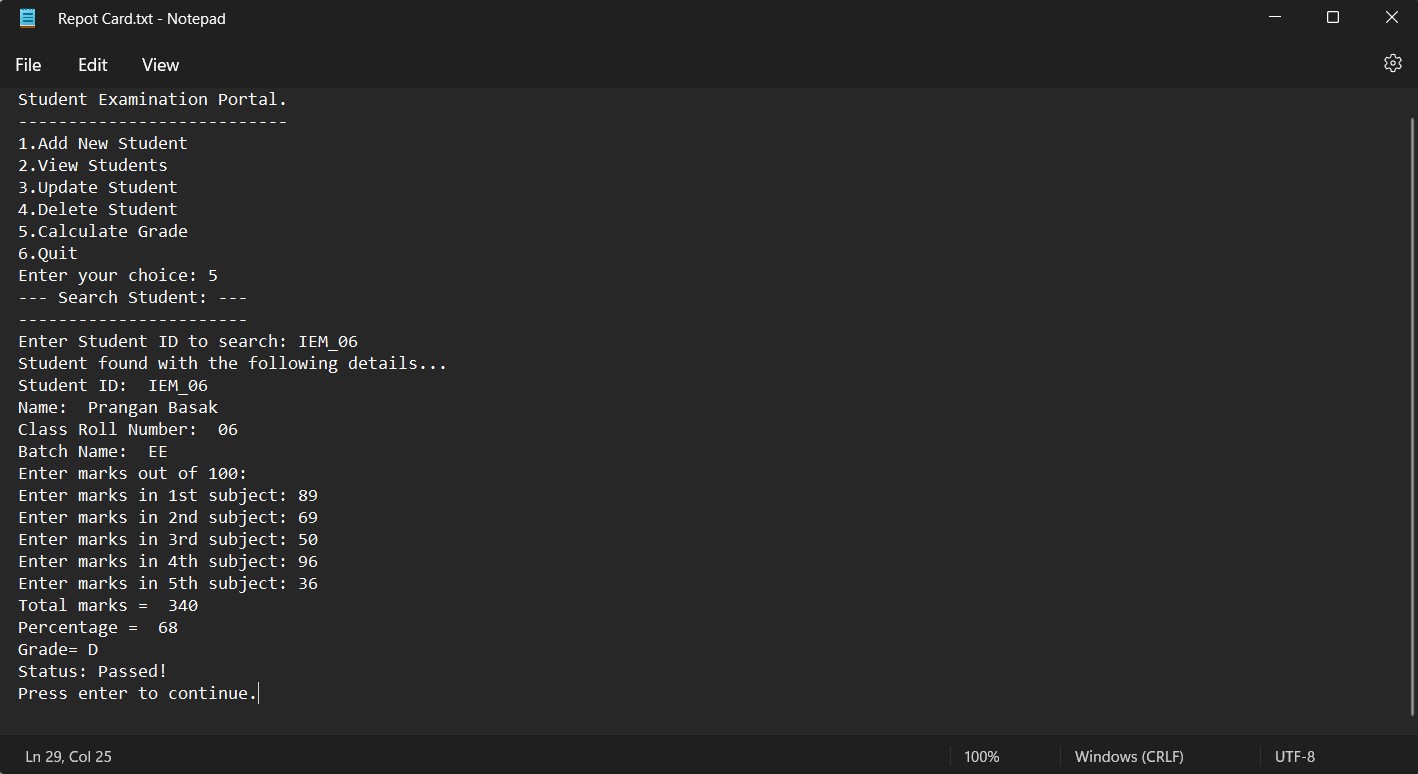
1. Updating student details.



1. Removing a student from the database.



1. Generating a report card (text file) of student showing percentage, grade in each subject and whether he passed or failed with all the marks uploaded.



**Thank You**