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:--

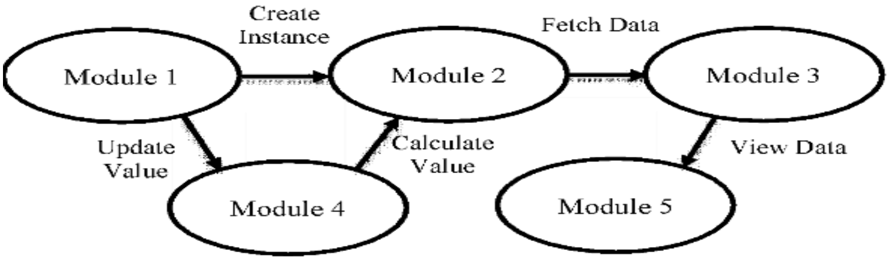
Students.csv

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:-*

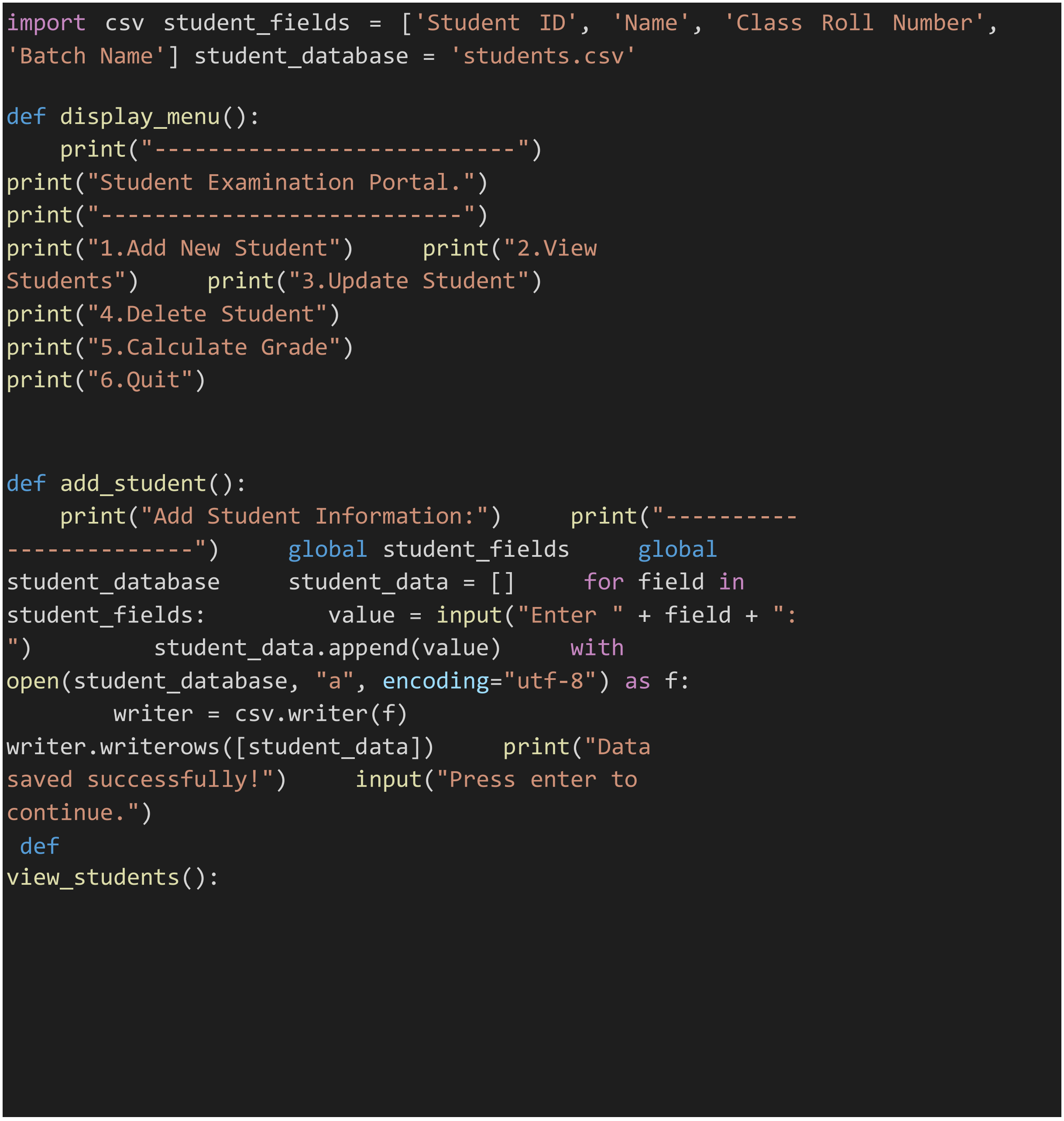
Demonstrates the dependency of all the python modules written using a data flow diagram.



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
```

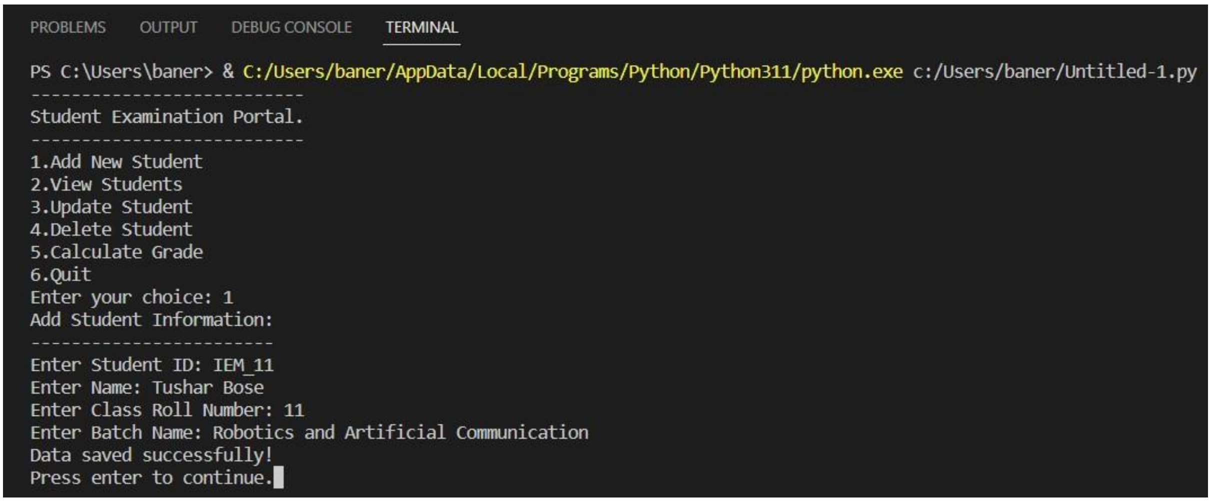
```
        else:
            print("Student ID not found in our database!")
input("Press enter to continue.")


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:-

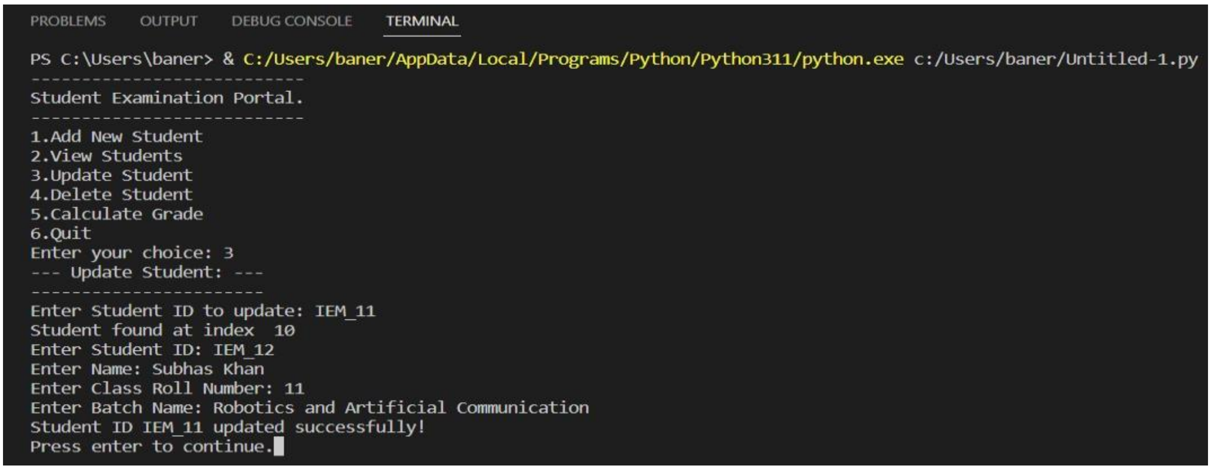
Sample outputs(screenshot) to demonstrate the functionalities in programs.

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



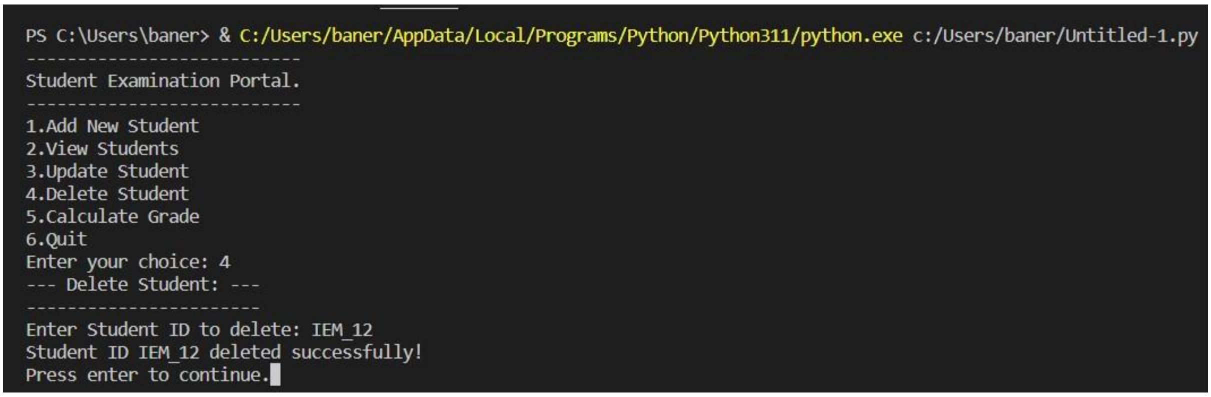
```
PS C:\Users\baner> & C:/Users/baner/AppData/Local/Programs/Python/Python311/python.exe c:/Users/baner/Untitled-1.py
---------------------------
Student Examination Portal.
---------------------------
1.Add New Student
2.View Students
3.Update Student
4.Delete Student
5.Calculate Grade
6.Quit
Enter your choice: 1
Add Student Information:
------------------------
Enter Student ID: IEM_11
Enter Name: Tushar Bose
Enter Class Roll Number: 11
Enter Batch Name: Robotics and Artificial Communication
Data saved successfully!
Press enter to continue.
```

2. Updating student details.



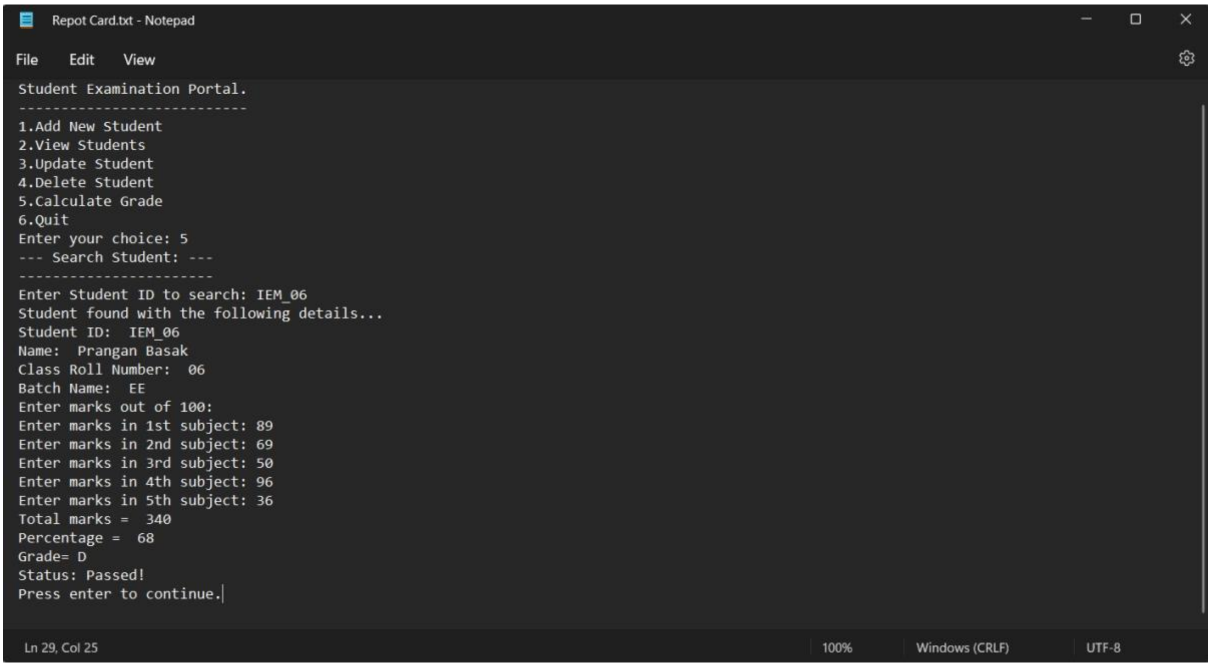
```
PS C:\Users\baner> & C:/Users/baner/AppData/Local/Programs/Python/Python311/python.exe c:/Users/baner/Untitled-1.py
---------------------------
Student Examination Portal.
---------------------------
1.Add New Student
2.View Students
3.Update Student
4.Delete Student
5.Calculate Grade
6.Quit
Enter your choice: 3
--- Update Student: ---
-----------------------
Enter Student ID to update: IEM_11
Student found at index  10
Enter Student ID: IEM_12
Enter Name: Subhas Khan
Enter Class Roll Number: 11
Enter Batch Name: Robotics and Artificial Communication
Student ID IEM_11 updated successfully!
Press enter to continue.
```

3. Removing a student from the database.



```
PS C:\Users\baner> & C:/Users/baner/AppData/Local/Programs/Python/Python311/python.exe c:/Users/baner/Untitled-1.py
---------------------------
Student Examination Portal.
---------------------------
1.Add New Student
2.View Students
3.Update Student
4.Delete Student
5.Calculate Grade
6.Quit
Enter your choice: 4
--- Delete Student: ---
-----------------------
Enter Student ID to delete: IEM_12
Student ID IEM_12 deleted successfully!
Press enter to continue.
```

4. 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.



```
Student Examination Portal.
---------------------------
1.Add New Student
2.View Students
3.Update Student
4.Delete Student
5.Calculate Grade
6.Quit
Enter your choice: 5
--- Search Student: ---
-----------------------
Enter Student ID to search: IEM_06
Student found with the following details...
Student ID:  IEM_06
Name:  Prangan Basak
Class Roll Number:  06
Batch Name:  EE
Enter marks out of 100:
Enter marks in 1st subject: 89
Enter marks in 2nd subject: 69
Enter marks in 3rd subject: 50
Enter marks in 4th subject: 96
Enter marks in 5th subject: 36
Total marks =  340
Percentage =  68
Grade= D
Status: Passed!
Press enter to continue.
```

Thank You