STUDENT EXAMINATION PORTAL

Submitted by

Name of the Students: AHELI MAJUMDAR
Enrolment Number: 12022002003225
Section: I
Class Roll Number: 73
Stream: ECE
Subject: Programming for Problem Solving with Python
Subject Code: IVC101
Department: Basic Science and Humanities

Under the supervision of
Mrs Sumana Sinha

Academic Year: 2022-26

PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE FIRST SEMESTER

DEPARTMENT OF BASIC SCIENCE AND HUMANITITES
INSTITUTE OF ENGINEERING AND MANAGEMENT, KOLKATA

CERTIFICATE OF RECOMMENDATION

We hereby recommend that the project prepared under our supervision by **AHELI MAJUMDAR** entitled **STUDENT EXAMINATION PORTAL** be accepted in partial fulfillment of the requirements for the degree of partial fulfillment of the first semester.

_____	_____
Head of the Department Basic Sciences and Humanities IEM, Kolkata	Project Supervisor

1 Introduction

A CSV file is a type of text file using specific structure to arrange tabular data. SQL stands for Structural Query Language which lets us access and manipulate databases. We have to prepare Student Examination Portal using python through MYSQL and CSV implement.

1.1 Objective

The objective of this project is to make better understanding of CSV file in python programming as well as to manage all information like students' batch, courses, profiles,marks, etc. The purpose of the project is to build an application program to reduce manual work for managing the required information students.

1.2 Organization of the Project

This project consists of three sections
i) Taking data input from the user
ii)Storing the data into different databases
iii) Accessing information from databases and giving the desired output.

2 Database Descriptions

We intend to create four databases:
1)STUDENT: Stores details of a student
2)COURSE: Stores details of all courses
3)BATCH: Stores details of all courses
4)DEPARTMENT: Stores details of all courses

2.1 Database Samples

3 Data Flow and E-R DIAGRAMS

4 Programs

Provide the python programs of the various modules.

```
import os
import csv
import subprocess
import time
import sys
try:
   import matplotlib.pyplot as plt
except:
   subprocess.run(['pip', 'install', 'matplotlib'])
   import matplotlib.pyplot as plt

path='C:\Users\aheli\OneDrive\Desktop\iem python proj'
print('-'*50)


#All the Functions used Throughout the code
def loading_screen():
   for i in range(10):
      sys.stdout.write("\rLoading" + "." * i)
      sys.stdout.flush()
      time.sleep(0.5)
   sys.stdout.write("\rLoading complete!")

def createfile(name,lst):
   with open(f'{path}/{name}','a',newline='')as f:
      script= csv.writer(f)
      script.writerow(lst)
      print(f"{name} file has been UPDATED")

def percent(num):
   if stream.lower()=='cse' or stream.lower()=='cseai' or
stream.lower()=='cseaiml' or stream.lower()=='cseiotcsbs':
      num=(num*100)//600
   elif stream.lower()=='it' or stream.lower()=='ece' or stream.lower()=='me':
      num=(num*100)//500
   return num
```

```
def grade(num):
    if num>=90:
        return("Outstanding Performance... You have passed the exam with grade
A.")
    elif num<90 and num>=80:
        return("Excellent Performance... You have passed the exam with grade
B.")
    elif num<80 and num>=70:
        return("Good Performance... You have passed the exam with grade C.")
    elif num<70 and num>=60:
        return("Your performance is average... Work hard... You have passed the
exam with grade D.")
    elif num<60 and num>=50:
        return("Your performance is below average... There is massive scope of
improvement... You have barely passed the exam with grade E.")
    else:
        return("Extremely poor performance... You have Failed the Exam and got
F.")

def count(lst):
    num=0
    for i in lst:
        if str(type(i))=="<class 'int'>":
            num+=1
        else:
            pass
    return num


def add(lst):
    plus=0
    for i in lst:
        try:
            plus+=i
        except:
            pass
    return plus
```

```
def duplicate(file,attr,pos=0):
    with open(f'{path}/{file}','r') as f:
        reader = csv.reader(f)
        dup_lst=[]
        for i in reader:
            dup_lst+=[i[pos]]
    if attr in dup_lst:
        return True
    else:
        return False

def choice(stream):
    if stream.lower()=='cse' or stream.lower()=='cseai' or
stream.lower()=='cseaiml' or stream.lower()=='cseiotcsbs':
        return ("C001:C002:C003:C004:C005:C006")
    elif stream.lower()=='it' or stream.lower()=='ece' or stream.lower()=='me':
        return ("C002:C003:C004:C005:C006")

def get_batch():
    with open(f'C:/PythonProgrammingProject_main-folder/Batch.csv','r') as f:
        reader=csv.reader(f)
        rows=[row for row in reader]
        column=[]
        for i in range(len(rows)):
            if i==0:
                pass
            else:
                column+=[rows[i][0]]
    return column

def remove(string):
    with
open(f'C:/PythonProgrammingProject_main-folder/Student.csv','r+',newline="
) as f:
        script=csv.reader(f)
        rows=[row for row in script]
        for i in rows:
            if i[0]==string:
                rows[rows.index(i)]=["","","",""]
            else:
```

```
            pass
        f.seek(0)
        f.truncate()
        writer=csv.writer(f)
        writer.writerows(rows)



def course_graph():

color_lst=['#C70039','#9BB1F2','#FFC300','#FF5733','#DAAFB1','#86B7C8']
    fig, ax = plt.subplots()
    legend_properties = {'weight':'heavy'}
    ax.set_facecolor("Black")
    ax.tick_params(axis="both", colors="white")
    fig.set_facecolor("Black")
    ax.set_xlabel('Grades--------->', color="white")
    ax.set_ylabel('No. of Students--------->', color="white")
    ax.spines["bottom"].set_color("white")
    ax.spines["left"].set_color("white")
    ax.xaxis.label.set_weight("heavy")
    ax.yaxis.label.set_weight("heavy")
    count=0
    with open(f'{path}/Course.csv','r')as f:
        script= csv.reader(f)
        rows=[row for row in script]
        req=[]
        for i in range(len(rows)):
            if i==0:
                pass
            else:
                req+=[rows[i][2]]
        lst=[['Python',(req[0].split('-'))[0:-1]],
             ['Math',(req[1].split('-'))[0:-1]],
             ['Physics',(req[2].split('-'))[0:-1]],
             ['Chemistry',(req[3].split('-'))[0:-1]],
             ['Biology',(req[4].split('-'))[0:-1]],
             ['English',(req[5].split('-'))[0:-1]]]

        for i in range(len(lst)):
```

```
            for j in range(len(lst[i][1])):
                try:
                    lst[i][1][j]=grade(int((lst[i][1][j].split(':'))[-1]))[-2]
                except:
                    lst[i][1][j]=''

        for k in range(6):
            a=lst[k][1].count('A')
            b=lst[k][1].count('B')
            c=lst[k][1].count('C')
            d=lst[k][1].count('D')
            e=lst[k][1].count('E')
            f=lst[k][1].count('F')
            lst[k][1]={'A':a,'B':b,'C':c,'D':d,'E':e,'F':f}

        for j in lst:
            x=list(j[1].keys())
            y=list(j[1].values())
            ax.plot(x, y,marker=",",color=color_lst[count],label=j[0],linewidth=3)
            leg=plt.legend(fontsize=10,loc="upper right",
facecolor="Black",edgecolor="Black",prop=legend_properties)
            count+=1

        for text in leg.get_texts():
            text.set_color('White')

        plt.show()

def batch_graph(arg):
    with open(f'{path}/Batch.csv','r') as f:
        reader=csv.reader(f)
        req=''
        rows=[row for row in reader]
        for i in range(len(rows)):
            if arg==rows[i][0]:
                req=rows[i][4]
                break
    req_lst=req.split(':')
    with open(f'{path}/Course.csv','r') as f:
        reader=csv.reader(f)
```

```
        rows=[row for row in reader]
        column=[]
        for i in range(len(rows)):
            if i==0:
                pass
            else:
                column+=[rows[i][2]]
        new_column=[]
        for j in range(len(column)):
            new_column+=(column[j].split('-'))[0:-1]
    new_req_lst=[]
    temp=[]
    for i in req_lst:
        for j in range(len(new_column)):
            if i in new_column[j]:
                temp+=[(new_column[j].split(':'))[-1]]
        new_req_lst+=[[[i]]+[temp]]
        temp=[]
    lst=[]
    temp=0
    grade_lst=[]
    for i in range(len(new_req_lst)):
        for j in range(6):
            try:
                temp+=int(new_req_lst[i][1][j])
            except:
                pass
        lst+=[new_req_lst[i][0]+[temp]]
        temp=0
    for i in range(len(lst)):
        if lst[i][0][:3]=='CSE':
            grade_lst+=[grade((lst[i][1]*100)//600)[-2]]
            lst[i][1]=grade((lst[i][1]*100)//600)[-2]
        else:
            grade_lst+=[grade((lst[i][1]*100)//500)[-2]]
            lst[i][1]=grade((lst[i][1]*100)//500)[-2]

grade_no_lst={'A':grade_lst.count('A'),'B':grade_lst.count('B'),'C':grade_lst.co
unt('C'),'D':grade_lst.count('D'),'E':grade_lst.count('E'),'F':grade_lst.count('F')}
```

```
    labels = list(grade_no_lst.keys())
    sizes = list(grade_no_lst.values())

color_lst=['#C70039','#9BB1F2','#FFC300','#FF5733','#DAAFB1','#86B7C8']
    explode = (0.01,0.1,0.02,0.05,0.03,0.1)
    new_labels=[]
    for i in range(len(labels)):
        new_labels+=[f'{labels[i]} : {str(sizes[i])}']

    fig,ax = plt.subplots()
    ax.set_facecolor("Black")
    fig.set_facecolor("Black")
    plt.rcParams['font.weight'] = 'heavy'
    #plt.rcParams['font.size'] = '1'

    patches, texts=ax.pie(sizes, labels=new_labels,
colors=color_lst,explode=explode,shadow=True,startangle=
-90,textprops={'fontsize': 0})

    centre_circle = plt.Circle((0,0),0.60,fc='black')
    fig = plt.gcf()
    fig.gca().add_artist(centre_circle)

    legend_properties = {'weight':'heavy'}

    leg=plt.legend(fontsize=10,loc="center",
facecolor="Black",edgecolor="Black",prop=legend_properties)
    for text in leg.get_texts():
        text.set_color('white')

    plt.title('Overall Grades vs No. of Students',color='White',weight='heavy')
    plt.axis('equal')
    plt.show()

def department_graph():
    need={}
    with open(f'{path}/Batch.csv','r') as f:
        reader=csv.reader(f)
        batch=[batch[0] for batch in reader]
        batch=batch[1:]
```

```
for arg in batch:
    avg=0
    with open(f'{path}/Batch.csv','r') as f:
        reader=csv.reader(f)
        req=''
        rows=[row for row in reader]
        for i in range(len(rows)):
            if arg==rows[i][0]:
                req=rows[i][4]
                break
    req_lst=req.split(':')
    with open(f'{path}/Course.csv','r') as f:
        reader=csv.reader(f)
        rows=[row for row in reader]
        column=[]
        for i in range(len(rows)):
            if i==0:
                pass
            else:
                column+=[rows[i][2]]
        new_column=[]
        for j in range(len(column)):
            new_column+=(column[j].split('-'))[0:-1]
    new_req_lst=[]
    temp=[]
    for i in req_lst:
        for j in range(len(new_column)):
            if i in new_column[j]:
                temp+=[(new_column[j].split(':'))[-1]]
        new_req_lst+=[[[i]]+[temp]]
        temp=[]
    lst=[]
    temp=0
    grade_lst=[]
    for i in range(len(new_req_lst)):
        for j in range(6):
            try:
                temp+=int(new_req_lst[i][1][j])
            except:
                pass
```

```
            lst+=[new_req_lst[i][0]+[temp]]
            temp=0
        for i in range(len(lst)):
            if lst[i][0][:3]=='CSE':
                lst[i][1]=(lst[i][1]*100)/600
            else:
                lst[i][1]=(lst[i][1]*100)/500
        for i in range(len(lst)):
            avg+=lst[i][1]
        avg=int(avg//len(lst))
        need[arg]=avg

    xdata = list(need.keys())
    ydata = list(need.values())

color_lst=['#C70039','#9BB1F2','#FFC300','#FF5733','#DAAFB1','#86B7C8']
    fig,ax = plt.subplots()
    ax.set_facecolor("Black")
    fig.set_facecolor("Black")
    ax.set_xlabel("X axis", color="white")
    ax.set_ylabel("Y axis", color="white")
    ax.spines["bottom"].set_color("white")
    ax.spines["left"].set_color("white")
    ax.spines['bottom'].set_linewidth(2)
    ax.spines['left'].set_linewidth(2)
    ax.xaxis.label.set_weight("heavy")
    ax.yaxis.label.set_weight("heavy")
    ax.tick_params(axis='x', labelcolor='white',
labelsize=10,color='white',width=2)
    ax.tick_params(axis='y', labelcolor='white',
labelsize=10,color='white',width=2)

    plt.barh(xdata,ydata,color=color_lst,height=0.3,align='center')

    plt.title('Histogram of Average of Students vs
Batch',color='white',pad=17,fontweight='bold')
    plt.xlabel('Average----------------->')
    plt.ylabel('Batch----------------->', labelpad=15)
    plt.show()
```

```
#Creation of Folder and all the Modules recquired...
try:
    os.makedirs(f'{path}/ReportCards')
    message=True
except:
    message=False

while message:
    createfile('Batch.csv',['Batch ID','Batch Name','Department Name','List of
Courses','List of Students'])
    createfile('Course.csv',['Course ID','Course Name','Marks Obtained'])
    with open(f'{path}/Course.csv','a',newline='')as f:
        script= csv.writer(f)
        script.writerow(['C001','Python Programming'])
        script.writerow(['C002','Math'])
        script.writerow(['C003','Physics'])
        script.writerow(['C004','Chemistry'])
        script.writerow(['C005','Biology'])
        script.writerow(['C006','English'])
    createfile('Department.csv',['Department ID','Department Name','List of
Batches'])
    with open(f'{path}/Department.csv','a',newline='')as f:
        script= csv.writer(f)
        script.writerow(['CSE','Computer Sience and Engineering'])
        script.writerow(['CSEAI','Computer Sience and Engineering and
Artificial Intelligence'])
        script.writerow(['CSEAIML','Computer Sience and Engineering and
Artificial Intelligence and Machine Learning'])
        script.writerow(['CSEIOTCSBS','Computer Sience and Engineering and
Internet of Things and Business Studies'])
        script.writerow(['IT','Information Technology'])
        script.writerow(['ECE','Electrical and Communications Engineering'])
        script.writerow(['ME','Mechanical Engineering'])
    createfile('Student.csv',['Student ID','Name','Class Roll Number','Batch
ID'])
    createfile('Examination.csv',['Course Name','Student ID','Marks'])
    break
```

```
print('\n','Computer Sience and Engineering : CSE','\n',
      'Computer Sience and Engineering and Artificial Intelligence : CSEAI','\n',
      'Computer Sience and Engineering and Artificial Intelligence and Machine
Learning : CSEAIML','\n',
      'Computer Sience and Engineering and Internet of Things and Business
Studies : CSEIOTCSBS','\n',
      'Information Technology : IT','\n',
      'Electrical and Communications Engineering : ECE','\n',
      'Mechanical Engineering : ME','\n')
print("Please write all the stream name in short form as mentioned above and
in capital letters only!!!")
print()


student_no=int(input("Enter the no. of students whose data you want to input :
"))
print()
print('-'*50)
for i in range(student_no):
    name=input("Enter Student's Name : ")
    batch=input("Which batch they are in (e.g. 2022-26) : ")
    stream=input("Which Stream are you in (e.g. CSE) : ")
    roll=input("What is your Class Roll Number : ")

    batch_id=stream+batch[2:4]
    student_id=batch_id+roll
    batch_name=stream+batch

    if duplicate('Student.csv',student_id,0):
        print("the student is already present in the directory")
        print(f"You can find your report card here :
{path}/ReportCards/{student_id}_{name}.txt")
    else:
        print()
        print("The subjects are
[Python,Math,Physics,Chemistry,Biology,English]")
        print('please enter the subjects marks in the above mentioned order in a
list type and if you dont have a particular subject write there "null" (e.g.
[100,100,"null",75,69,85])')
        print('Each Subject is ot of 100 marks')
```

```
        print()
        marks_lst=eval(input("Enter the Marks list : "))
        total_marks=add(marks_lst)
        print()

        with
open(f"{path}/ReportCards/{student_id}_{".join(name.split())}.txt",'w') as f:

            f.writelines([f'Name of the student : {name} \n',
                          f'Class Roll of the student : {roll} \n',
                          f'Stream of the student : {stream} \n',
                          f'Your Student ID is : {student_id}\n',
                          '\n',
                          f'Marks obtained in Math is : {marks_lst[1]} \n',
                          f'Marks obtained in Python is : {marks_lst[0]} \n',
                          f'Marks obtained in Physics is : {marks_lst[2]} \n',
                          f'Marks obtained in Chemistry is : {marks_lst[3]} \n',
                          f'Marks obtained in Biology is : {marks_lst[4]} \n',
                          f'Marks obtained in English is : {marks_lst[5]} \n'])

            f.write('\n')
            f.write(f'You have got {total_marks} in total with
{percent(total_marks)}%\n')
            f.write(grade(total_marks/count(marks_lst)))
        createfile('Student.csv',[student_id,name,roll,batch_id])
        print(f"You can find your report card here :
{path}/ReportCards/{student_id}_{".join(name.split())}.txt")
        openpath=f"{path}/ReportCards/{student_id}_{".join(name.split())}.txt"
        subprocess.run(['start',openpath], shell=True)

        ask=input("Do you want to remove this name from database now is the
time (Y/N) : ")

        if ask.lower()=='n':
            if duplicate('Batch.csv',batch_id,0):
                with open(f'{path}/Batch.csv','r+',newline='') as f:
                    script=csv.reader(f)
                    rows=[row for row in script]
                    for i in rows:
                        if batch_id==i[0]:
```

```
                    rows[rows.index(i)][4]+=f':{student_id}'
                f.seek(0)
                f.truncate()
                writer=csv.writer(f)
                writer.writerows(rows)

            print("Batch.csv has been updated")
        else:

createfile('Batch.csv',[batch_id,batch_name,stream,choice(stream),student_id]
)

        with open(f'{path}/Course.csv','r+',newline='') as f:
            script=csv.reader(f)
            rows=[row for row in script]
            for i in range(len(rows)):
                if i==0:
                    pass
                else:
                    try:
                        rows[i][2]+=f'{student_id}:{marks_lst[i-1]}-'
                    except:
                        rows[i].append(f'{student_id}:{marks_lst[i-1]}-')
            f.seek(0)
            f.truncate()
            writer=csv.writer(f)
            writer.writerows(rows)
    else:
        remove(student_id)
        subprocess.call("TASKKILL /F /IM notepad.exe", shell=True)
        os.remove(openpath)
        print('Your details have been successfully removed from the directory')
print('-'*50)
print()

try:
    with open(f'{path}/Department.csv','r+',newline='') as f:
        script=csv.reader(f)
        rows=[row for row in script]
        lst=get_batch()
```

```
        for i in lst:
            for j in rows:
                if i[0:-2]==j[0]:
                    try:
                        if i in j[2]:
                            pass
                        else:
                            rows[rows.index(j)][2]+=f'{i}:'
                    except:
                        rows[rows.index(j)].append(f'{i}:')
                    break
        f.seek(0)
        f.truncate()
        writer=csv.writer(f)
        writer.writerows(rows)

except:
    print("Nothing to add in Department.csv")



#Creation of the Graphs...
print()
print("Give the details Below to see the Batchwise percent Graph")
batch=input("Which batch they are in (e.g. 2022-26) : ")
stream=input("Which Stream are they in (e.g. CSE) : ")
print('Please Close the Figure window after viewing to continue')
batch_id=stream+batch[2:4]

with open(f'{path}/Batch.csv','r') as f:
    reader=csv.reader(f)
    batch=[batch[0] for batch in reader]
    batch=batch[1:]

while True:
    if batch_id in batch:
        batch_graph(batch_id)
        break
    else:
        print(f'details with {batch_id} this Batch ID is not in the directory')
```

```
        ask=input("Do you want to continue (y/n) : ")
        if ask.lower()=='y':
            batch=input("Which batch they are in (e.g. 2022-26) : ")
            stream=input("Which Stream are they in (e.g. CSE) : ")
            batch_id=stream+batch[2:4]
            continue
        else:
            print('OK')
            break
print()
print('The overall Course graph will come now')
print('Please Close the Figure window after viewing to continue')
loading_screen()
course_graph()
print()
print()
print("The overall Department wise average graph will come now")
print('Please Close the Figure window after viewing to continue')
loading_screen()
department_graph()
print()
print()

last=input("Press Enter to exit")
subprocess.call("TASKKILL /F /IM notepad.exe", shell=True)
```

5 Outputs

Describe sample outputs to demonstrate the functionalities in programs. You may use screenshots.