**CERTIFICATE**

This is to certify that Aayushi Ramesh Patel of Class 12A has prepared the investigatory project in Computer Science entitled Application Shortlisting Software.

The report is the result of her efforts and endeavours. The report is found worthy of acceptance as the final project report for the subject Computer Science of Class XII. She has prepared this report under my guidance.

**Internal Examiner:** **Signature:**

**Name: Mrs. Bharati Rameshbhai Patel**

**External Examiner: Signature:**

**Head of Senior: Signature:**

**Name: Mrs. Priya Sadasivam**

**Principal** **Signature:**

**Name: Mr. Hussein Modak**

**ACKNOWLEDGEMENT**

I would like to express my deep sense of thanks and gratitude to the Subject guide Mrs. Bharati Rameshbhai Patel, HOD of Computer Science for guiding me immensely through the course of my project. The teacher had always evinced keen interest in my work. The constructive advice and constant motivation provided, have been responsible for successful completion of this project.

My sincere thanks also go to go to the Principal, Mr. Hussein Modak and Mrs Priya Sadasivam, Head of School for extending every possible support.

I also thank my parents and my classmates for their timely support and guidance. I sincerely extend my thanks to all those who had helped directly or indirectly towards the completion of this project in fulfilment of my course.

Name of the Student: Signature:

Aayushi Ramesh Patel

**CONTENT**

1. Introduction 5

* Python 5
* SQL(Structured Query Language) 6
* Tkinter 6
* Numpy 7
* Matplotlib 7

1. Overview 8
2. System Requirements 8
3. Source Code 9

* Application Form 9
* SQL Database 10
* SQL Queries done through Python 12

1. Output 15
2. Shortcomings 18
3. Bibliography 18

**INTRODUCTION**

**Python**



Python is a general-purpose programming language. It was created in 1991 by Guido van Rossum. Hence, you can use the programming language for developing both desktop and web applications.

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics.

Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, Easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance.

Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms and can be freely distributed.

Python provides increased productivity. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault.

A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on.

**SQL(Structured Query Language)**

SQL is a language that enables you to work with a database. Using SQL, you can insert records, update records, and delete records. You can also create new database objects such as databases and tables. And you can drop (delete) them.

More advanced features include creating stored procedures (self contained scripts), views (pre-made queries), and setting permissions on database objects (such as tables, stored procedures, and views).

Although SQL is an ANSI (American National Standards Institute) standard, there are many different versions of SQL. Different database vendors have their own variations of the language.

Having said this, to be in compliance with the ANSI standard, they need to at least support the major commands such as DELETE, INSERT, UPDATE, WHERE etc. Also, you will find that many vendors have their own extensions to the language — features that are only supported in their database system.

**Tkinter**

Tkinter is Python's de-facto standard GUI (Graphical User Interface) package used to create simple GUI apps. It is the most commonly used module for GUI apps in the Python. The Tkinter module (“Tk interface”) is the standard Python interface to the Tk GUI toolkit from [Scriptics](http://www.scriptics.com/) (formerly developed by Sun Labs).

Both Tk and Tkinter are available on most Unix platforms, as well as on Windows and Macintosh systems. Starting with the 8.0 release, Tk offers native look and feel on all platforms.

Tkinter consists of a number of modules. The Tk interface is provided by a binary extension module named **\_**tkinter. This module contains the low-level interface to Tk, and should never be used directly by application programmers. It is usually a shared library (or DLL) but might in some cases be statically linked with the Python interpreter.

**NumPy**

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

* a powerful N-dimensional array object
* sophisticated functions
* useful linear algebra

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data.

**Matplotlib**

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib tries to make easy things easy and hard things possible.

You can generate plots, histograms, power spectra, bar charts, scatterplots, etc., with just a few lines of code.

For simple plotting the pyplot module provides an interface where you have full control of line styles, font properties, axes properties, etc, via an object oriented interface.

**OVERVIEW**

This Application Shortlisting Software is made to ease the strenuous process of resume selection. The utilization of this software is quite commonplace in the job market and also in universities.

A combination of programs has been used enable a connection between Python and SQL. The selection procedure begins with the input of a student’s details in a form created by using the Tkinter module provided by python. The student details include their first and last names, Date of birth, Department of choice, Subject marks and grades, SAT subject scores and School Rank.

All of the data from the forms is saved on an SQL Database using the ‘mysqlconnector’ module. The information is further retrieved in python by running multiple queries. These queries are used to shortlist the best candidates. The retrieved information is displayed in a creative manner using graphs and buttons, thereby making the project interactive.

**SYSTEM REQUIREMENTS**

**For development**

OS: Windows 7 SP1+, 8, 10, 64-bit versions only; macOS 10.11+

4GBRAM or more

**For running the application software**

* MySQL Workbench
* Python 3
* Mysqlconnector module installed through pip installation
* Matplotlib module installed through pip installation
* Numpy module installed through pip installation
* Tkinter module

**APPLICATION FORM**

from tkinter import \*

from tkinter import filedialog

root = Tk()

root.geometry('1000x1500')

root.title("Registration Form")

import os

import subprocess as sp

import mysql.connector

mydb=mysql.connector.connect(host='localhost',user='root',passwd='Akis.123',database="Application")

mycursor=mydb.cursor()

#To specify what kind of variable is going to be used

fn=StringVar()

ln=StringVar()

dob=StringVar()

engi=StringVar()

st=IntVar()

ge= IntVar()

phy=IntVar()

chem=IntVar()

math=IntVar()

sn=StringVar()

cr=IntVar()

eg=StringVar()

ep=DoubleVar()

mg=StringVar()

mp=DoubleVar()

og=StringVar()

op=DoubleVar()

cg=StringVar()

cp=DoubleVar()

pg=StringVar()

pp=DoubleVar()

#function to exit the form

def exitt():

exit()

#to store values inputed in form into variables and print them

def printt():

first=fn.get()

sec=ln.get()

var1=var.get()

date1=d.get()

month1=m.get()

year1=y.get()

dept=engi.get()

stc=st.get()

ma=math.get()

c=chem.get()

p=phy.get()

sstc=int(ma)+int(c)+int(p)

gen=ge.get()

school=sn.get()

rank=cr.get()

ep1=ep.get()

mp1=mp.get()

op1=op.get()

cp1=cp.get()

pp1=pp.get()

overall=(ep1+mp1+op1+cp1+pp1)/5

if gen==2:

gen="female"

elif gen==1:

gen="male"

details=[first+' '+sec, date1+"/"+month1+"/"+year1,var1,school,dept,int(stc),int(sstc),gen,overall,rank]

details=tuple(details)

path="C:\\Users\\500490\\Desktop\\cs project"

cn1=os.path.join(path,'Detailsfile.txt' )

dfile=open(cn1,"w")

dfile.write(str(details))

print (details)

mycursor.execute('insert resumemain values (%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)',details)

mydb.commit()

mycursor.execute("select \* from Resumemain")

for i in mycursor:

print (i)

#important widgets and uses

'''

The Entry widget is used to display a single-line text field for accepting values from a user.

'''

'''

The Label widget is used to provide a single-line caption for other widgets. It can also contain images.

'''

'''

The Menu widget is used to provide various commands to a user. These commands are contained inside Menubutton.

'''

'''

The Radiobutton widget is used to display a number of options as radio buttons. The user can select only one option at a time.

'''

'''

The place() Method − This geometry manager organizes widgets by placing them in a specific position in the parent widget.

'''

label\_0=Label(root,text="Registration Form", relief="solid",width=20)

label\_0.place(x=175,y=10)

#for First Name of person

label\_1=Label(root,text="FirstName :",width=20,font=("bold",10))

label\_1.place(x=80,y=40)

entry\_1=Entry(root,textvar=fn)

entry\_1.place(x=240,y=42)

#for last name of person

label\_2=Label(root,text="LastName :",width=20,font=("bold",10))

label\_2.place(x=80,y=80)

entry\_2=Entry(root,textvar=ln)

entry\_2.place(x=240,y=82)

#For date of birth of person

dob=Label(root, text="DOB :",width=20,font=("bold", 10))

dob.place(x=65,y=120)

#drop down menu for Date of birth

l=[]

for i in range(1,32):

l.append(str(i))

day=l;

d=StringVar()

dropd=OptionMenu(root,d, \*day)

dropd.config(width=4)

d.set("Date")

dropd.place(x=240,y=125)

#drop down menu for Month of birth

month = ['January','February','March','April','May','June','July','August','September','October','November','December'];

m=StringVar()

dropm=OptionMenu(root,m, \*month)

dropm.config(width=10)

m.set("Month")

dropm.place(x=300,y=125)

#drop down menu for Year of birth

year=['1999','2000','2001','2002','2003','2004','2005','2006','2007'];

y=StringVar()

dropy=OptionMenu(root,y, \*year)

dropy.config(width=5)

y.set("Year")

dropy.place(x=400,y=125)

#for Nationality

label\_4=Label(root,text="Nationality :",width=20,font=("bold",10))

label\_4.place(x=75,y=170)

var=StringVar()

l=['Indian','British','American','Canadian','Qatari']

droplist=OptionMenu(root,var,\*l)

var.set('Select Nationality')

droplist.config(width=15)

droplist.place(x=230,y=170)

#Program of Application

label\_5=Label(root,text="I am applying for :",width=20,font=("bold",10))

label\_5.place(x=75,y=220)

l1=["Chemical Engineering"," Civil Engineering","Mechanical Engineering","IT","Computer Science","Electronics and Communication","Mathematics"]

droplist=OptionMenu(root,engi,\*l1)

engi.set('Select Department')

droplist.config(width=20)

droplist.place(x=230,y=225)

#SAT scores

label\_6=Label(root,text="SAT Scores :",width=20,font=("bold",10))

label\_6.place(x=75,y=270)

entry\_6=Entry(root,textvar=st)

entry\_6.place(x=240,y=275)

#SAT subject scores which will be later added together

label\_7=Label(root,text="SAT Subject Scores :",width=20,font=("bold",10))

label\_7.place(x=75,y=320)

label\_7a=Label(root,text="Physics:",width=10,font=("bold",10))

label\_7a.place(x=150,y=350)

label\_7b=Label(root,text="Chemistry:",width=10,font=("bold",10))

label\_7b.place(x=150,y=380)

label\_7c=Label(root,text="Maths:",width=10,font=("bold",10))

label\_7c.place(x=150,y=410)

entry\_7c=Entry(root,textvar=math)

entry\_7c.place(x=270,y=415)

entry\_7b=Entry(root,textvar=chem)

entry\_7b.place(x=270,y=385)

entry\_7a=Entry(root,textvar=phy)

entry\_7a.place(x=270,y=355)

#Gender of applicant

label\_8= Label(root, text="Gender",width=20,font=("bold", 10))

label\_8.place(x=75,y=450)

Radiobutton(root, text="Male",padx = 5, variable=ge, value=1).place(x=240,y=450)

Radiobutton(root, text="Female",padx = 20, variable=ge, value=2).place(x=300,y=450)

#School Name

label\_9=Label(root,text="School Name :",width=20,font=("bold",10))

label\_9.place(x=600,y=40)

entry\_9=Entry(root,textvar=sn)

entry\_9.place(x=750,y=42)

#Class Rank

label\_10=Label(root,text="Class Rank:",width=20,font=("bold",10))

label\_10.place(x=594,y=80)

entry\_10=Entry(root,textvar=cr)

entry\_10.place(x=750,y=82)

#grades and percentage

label\_11=Label(root,text="GRADES AND PERCENTAGE:",width=40,font=("bold",10))

label\_11.place(x=600,y=120)

#English

label\_11a=Label(root,text="English:",width=40,font=("bold",10))

label\_11a.place(x=600,y=160)

label\_11aa=Label(root,text="Grade:",width=20,font=("bold",10))

label\_11aa.place(x=600,y=190)

label\_11ab=Label(root,text="%:",width=20,font=("bold",10))

label\_11ab.place(x=600,y=220)

entry\_11aa=Entry(root,textvar=eg)

entry\_11aa.place(x=730,y=190)

entry\_11ab=Entry(root,textvar=ep)

entry\_11ab.place(x=730,y=220)

#Maths

label\_11b=Label(root,text="Maths:",width=40,font=("bold",10))

label\_11b.place(x=600,y=260)

label\_11ba=Label(root,text="Grade:",width=20,font=("bold",10))

label\_11ba.place(x=600,y=290)

label\_11bb=Label(root,text="%:",width=20,font=("bold",10))

label\_11bb.place(x=600,y=320)

entry\_11ba=Entry(root,textvar=mg)

entry\_11ba.place(x=730,y=290)

entry\_11bb=Entry(root,textvar=mp)

entry\_11bb.place(x=730,y=320)

#Optionals

label\_11c=Label(root,text="Optional Subject:",width=40,font=("bold",10))

label\_11c.place(x=600,y=360)

label\_11ca=Label(root,text="Grade:",width=20,font=("bold",10))

label\_11ca.place(x=600,y=390)

label\_11cb=Label(root,text="%:",width=20,font=("bold",10))

label\_11cb.place(x=600,y=420)

entry\_11ca=Entry(root,textvar=og)

entry\_11ca.place(x=730,y=390)

entry\_11cb=Entry(root,textvar=op)

entry\_11cb.place(x=730,y=420)

#Chemistry

label\_11d=Label(root,text="Chemistry:",width=40,font=("bold",10))

label\_11d.place(x=600,y=460)

label\_11da=Label(root,text="Grade:",width=20,font=("bold",10))

label\_11da.place(x=600,y=490)

label\_11db=Label(root,text="%:",width=20,font=("bold",10))

label\_11db.place(x=600,y=520)

entry\_11da=Entry(root,textvar=cg)

entry\_11da.place(x=730,y=490)

entry\_11db=Entry(root,textvar=cp)

entry\_11db.place(x=730,y=520)

#Physics

label\_11e=Label(root,text="Physics:",width=40,font=("bold",10))

label\_11e.place(x=600,y=560)

label\_11ea=Label(root,text="Grade:",width=20,font=("bold",10))

label\_11ea.place(x=600,y=590)

label\_11eb=Label(root,text="%:",width=20,font=("bold",10))

label\_11eb.place(x=760,y=620)

entry\_11ea=Entry(root,textvar=pg)

entry\_11ea.place(x=730,y=590)

entry\_11eb=Entry(root,textvar=pp)

entry\_11eb.place(x=730,y=620)

#To upload files from any folder and save the file name as the name of the person

def upload():

root.filename = filedialog.asksaveasfilename(initialdir = "/",title = "Select file",filetypes = (("txt files","\*.txt"),("all files","\*.\*")))

print (root.filename)

f=open(root.filename,'r')

frd=f.read()

first=fn.get()

#to save the filename as the name of the applicant

a=str(first)+'.txt'

#path=r"C:\\Users\\500490\\Desktop\\cs project"

#cn=os.path.join(path,r'{first}.txt')

file=open(a,"w")

#file=open(f'{first}.txt','w')

file.write(frd)

file.close()

'''

The Button widget is used to display buttons in your application.

'''

but\_u=Button(root,text="Upload File on Extracurriculars(.txt)",width=35,bg="brown",fg="White",command=upload).place(x=200,y=520)

but\_signup=Button(root,text="Submit",width=20,bg="brown",fg="White",command=printt).place(x=150,y=600)

but\_quit=Button(root,text="Quit",width=20,bg='brown',fg='white',command=exitt).place(x=350,y=600)

root.mainloop()

**SQL DATABASE**

create database Application;

use Application;

show databases;

create table Resumemain(Fullname char(40),

DOB char(40),

Nationality char(40),

School char(40),

Dept char(60),

SAT int,

SATSubject int,

Gender char(10),

Overall int,

ClassRank int);

desc resumemain;

select \* from resumemain;

drop database Application;

**RUNNING QUERIES**

from tkinter import Tk, Label, Button

from matplotlib import pyplot as plt;plt.rcdefaults()

import mysql.connector

import numpy as np

#HOW WE CONNECT PYTHON TO SQL

'''

There are mainly seven steps in order to create a database connectivity.

STEPS

1.Start Python

2.Import the packages required for database programming

3.Open a connection to database

4.Create a cursor instance

5.Execute a query

6.Extract data from result set

7.Clean up the environment

'''

mydb = mysql.connector.connect(

host="localhost",

user="root",

passwd="Akis.123",

database="Application"

)

mycursor = mydb.cursor()

#A Class is like an object constructor, or a "blueprint" for creating objects

#self

#The word 'self' is used to represent the instance of a class.

#By using the "self" keyword we access the attributes

#and methods of the class in python.

#\_\_init\_\_ method

#"\_\_init\_\_" is a reseved method in python classes.

#It is called as a constructor in object oriented terminology.

#This method is called when an object is created from a class

#and it allows the class to initialize the attributes of the class.

class App:

def \_\_init\_\_(self, master):

self.master = master

master.title("Queries")

self.label = Label(master, text="Select any one")

self.label.pack()

self.sat\_button = Button(master, text="SAT toppers", command=self.sat)

self.sat\_button.pack()

self.sats\_button = Button(master, text="SAT Subject toppers", command=self.sats)

self.sats\_button.pack()

self.overall\_button = Button(master, text="High Overall scores", command=self.overall)

self.overall\_button.pack()

self.nat\_button = Button(master, text="Nationality", command=self.nat)

self.nat\_button.pack()

self.extra\_button = Button(master, text="Extracurricular", command=self.extra)

self.extra\_button.pack()

self.age\_button = Button(master, text="Age", command=self.age)

self.age\_button.pack()

'''

1. data=cursor.fetchall() returns all the records

retrieved as per query in tuple form

2. data=cursor.fetchone()returns one record

retrieved from the resultset as per query as a tuple or a list.

•First time it returns the first record.

Next time it will fetch the next record and so on.

•The method returns one record as a tuple.

•If there are no more records then it returns None

'''

def sat(self):

mycursor.execute("SELECT Fullname, SAT FROM resumemain order by SAT desc")

l=[]

for i in range(5):

myresult1 = mycursor.fetchone()

a=list(myresult1)

l.append(a)

print (myresult1)

print (l)

l2=[]

l3=[]

for i in range(0,len(l)):

l2.append(l[i][0])

y=l2

for i in range(0,len(l)):

l3.append(l[i][1])

z=l3

plt.plot(y,z,label="SAT SCORE TOPPERS", color='red')

plt.xlabel("NAMES")

plt.ylabel("SAT SCORES")

plt.legend()

plt.show()

def sats(self):

mycursor.execute("SELECT Fullname, SATSubject FROM resumemain order by SATSubject desc")

l=[]

for i in range(5):

myresult1 = mycursor.fetchone()

a=list(myresult1)

l.append(a)

print (myresult1)

print (l)

l2=[]

l3=[]

for i in range(0,len(l)):

l2.append(l[i][0])

y=l2

for i in range(0,len(l)):

l3.append(l[i][1])

z=l3

plt.plot(y,z,label="SAT SUBJECT SCORE TOPPERS", color='green')

plt.xlabel("NAMES")

plt.ylabel("SAT SUBJECT SCORES")

plt.legend()

plt.show()

def nat(self):

mycursor.execute("SELECT Fullname, Nationality FROM resumemain order by classrank asc")

n=input("Which nationality do you prefer?")

l=[]

myresult1 = mycursor.fetchall()

for i in myresult1:

x=list(i)

for j in x:

if j==n:

l.append(i)

print (l)

'''

NumPy arange() is one of the array creation routines

based on numerical ranges. It creates an instance of ndarray

with evenly spaced values and returns the reference to it.

You can define the interval of the values contained

in an array, space between them, and their type with

four parameters of arange():

numpy.arange([start, ]stop, [step, ], dtype=None)

'''

def overall(self):

mycursor.execute("SELECT Fullname, Overall FROM resumemain order by Overall desc")

l=[]

for i in range(5):

myresult1 = mycursor.fetchone()

a=list(myresult1)

l.append(a)

print (myresult1)

print (l)

l2=[]

l3=[]

for i in range(len(l)):

l2.append(l[i][0])

y=l2

for i in range(len(l)):

l3.append(l[i][1])

z=l3

plt.bar(y,z,align='center',alpha=0.5,color="")

plt.title("OVERALL TOPPERS")

plt.xlabel("NAMES")

plt.ylabel("OVERALL SCORES")

plt.yticks(np.arange(0,100,2))

plt.show()

def extra(self):

mycursor.execute("select count(Fullname) from resumemain")

myre=mycursor.fetchone()

print (myre)

q=int(myre[0])

print (q)

mycursor.execute("select Fullname from resumemain")

l2=[]

c=input("Enter what criteria you want to select on:")

for i in range(q):

myresult=mycursor.fetchone()

x=str(myresult[0])

a=x.split()

l2.append(a[0])

print (l2)

L=[]

'''

file.readlines( )

readlines() returns the complete file as a list of strings

each separated by \n

'''

for i in range(len(l2)):

q=str(l2[i])+'.txt'

print (q)

f=open(q , "r")

s1=f.readlines()

for z in range(len(s1)):

s=s1[z].split(" ")

for k in range(len(s)):

a=s[k].strip()

if a.upper()==c.upper():

L.append(l2[i])

print (c,':',L)

for i in L:

q=str(i)+'.txt'

f=open(q , "r")

f1=f.read()

print (f1)

def age(self):

mycursor.execute("SELECT Fullname, DOB FROM resumemain order by DOB")

l=[]

for i in range(5):

myresult1 = mycursor.fetchone()

a=list(myresult1)

l.append(a)

print (myresult1)

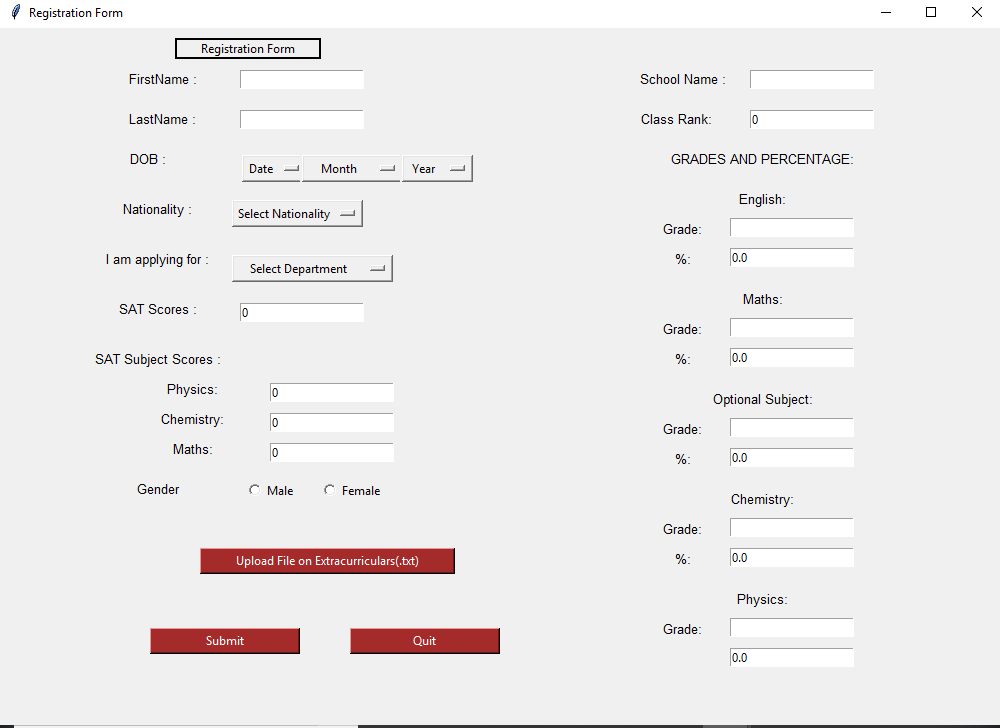
print (l)

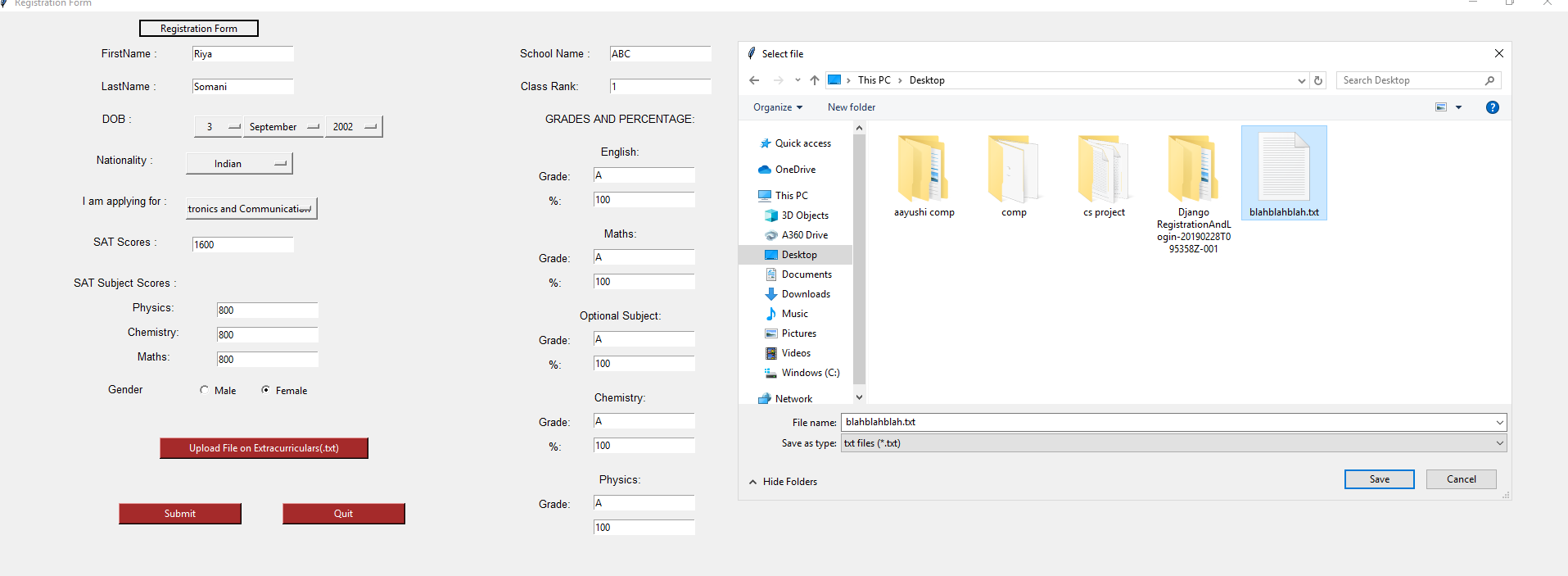
root = Tk()

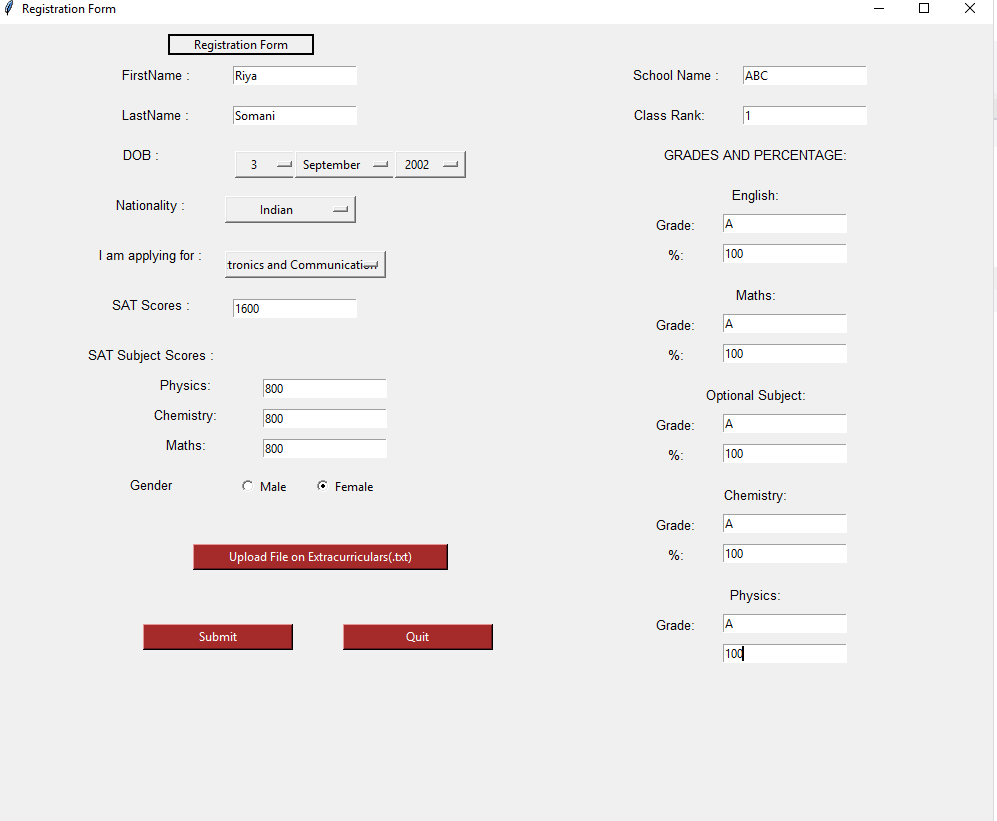
App(root)

**OUTPUTS**

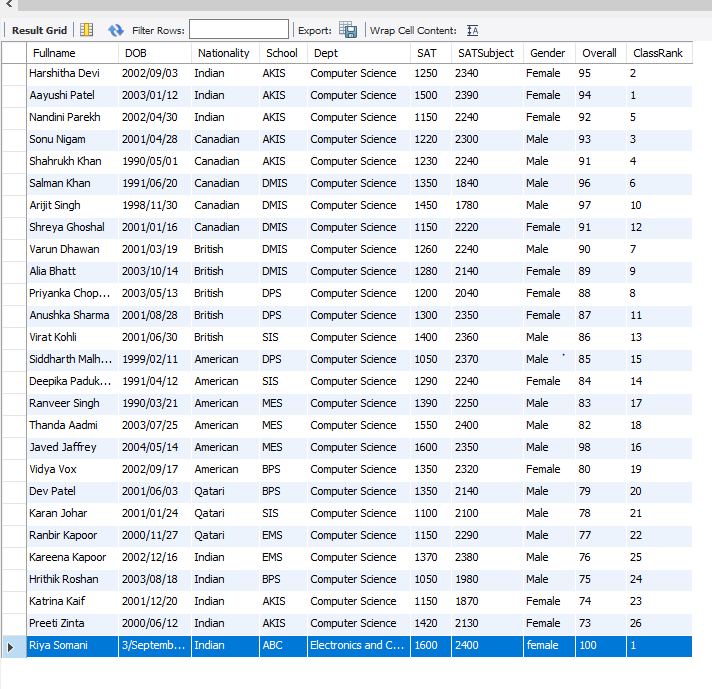
**Application form**



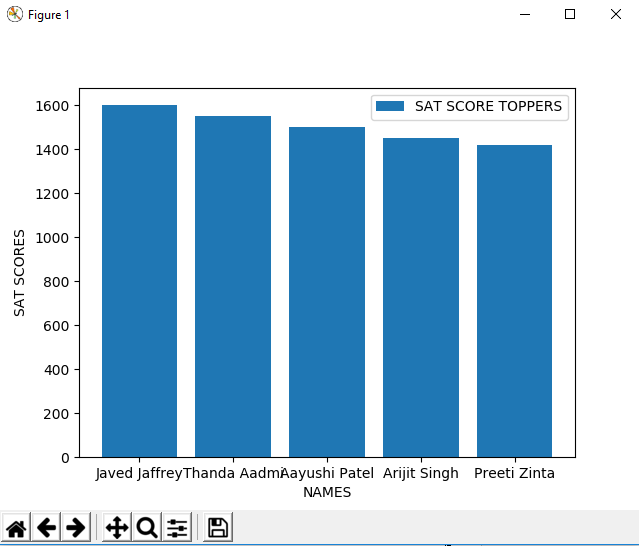




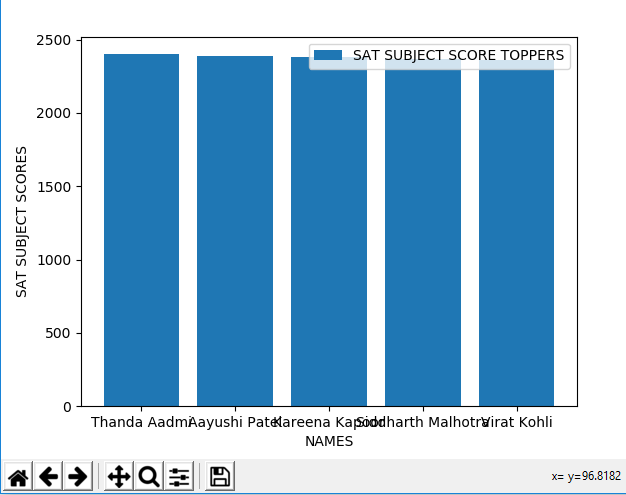
**SQL**



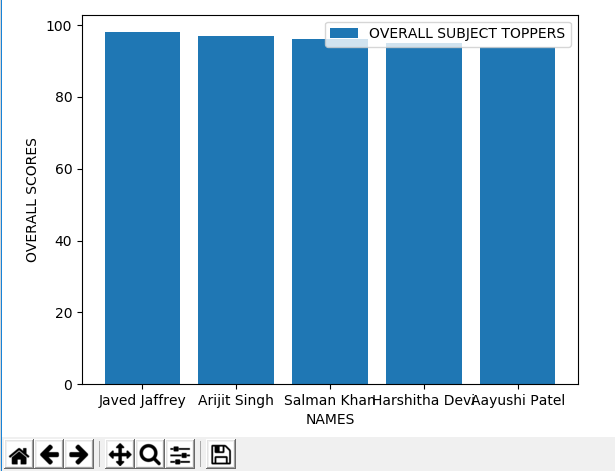
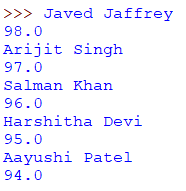
**Query 1**



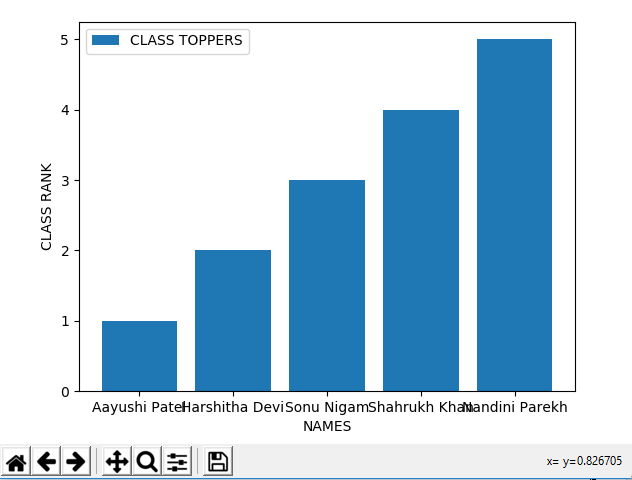
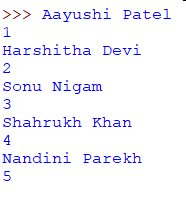
**Query 2**



**Query 3**



**Query 4**



**Query 5**



**SHORTCOMINGS**

The project could have been further extended by

* Adding more search queries in the form of buttons
* Adding more graphs
* Doing a better keyword search for choosing those with the best extra-curricular activities.
* Making our program work in other platforms too.(cross-platform)
* Connecting it to Django

**BIBLIOGRAPHY**

**Web Links**

* https://matplotlib.org
* https://numpy.org/
* https://www.datacamp.com/community/tutorials/gui-tkinter-python
* https://effbot.org/tkinterbook/tkinter-whats-tkinter.htm
* https://www.python.org/doc/essays/blurb/
* https://www.guru99.com/introduction-to-database-sql.html
* https://www.w3schools.com/
* https://stackoverflow.com/

**Books**

* Computer Science with Python Class XII-Sumita Arora
* Computer Science with Python Class XI-Sumita Arora
* NCERT Computer Science Grade 11
* NCERT Computer Science Grade 12