

Development of a Student Profile System

Submitted To Submitted by

**Deepak Kumar Karna** **James Group**

Department of Cyber Security and Digital Forensics

The British College, Kathmandu

(Under the co-operation of University of West England Bristol)

**Declaration**

This report is the result of a case study done in Student Profile System using (Data Science). The case study is the requirement of the completion of the second semester of Bachelor in Cyber Security and Digital Forensics. The work presented in this project report is original and has not been submitted elsewhere for any other academic or non-academic purpose.Any external sources of information used in this project have been properly cited and

**Word count: 1979**

**Date of Submission: 6th April 2024**

**Group members**

***1)David Rai***

***2)James Limbu***

***3) Aju Ra***i

**Abstract**

This project aims to develop a Student Profile System for the Fundamentals of Data Science Level 3 course at The British College. The system is designed to manage student personal information, academic records, and extracurricular activities efficiently. It incorporates principles of object-oriented programming, modular design, file handling, and error handling to create a user-friendly and robust application. The project report outlines the solution design, implementation details, submission requirements, and a reflection on the development process. Through this project, we demonstrate our understanding of fundamental data science concepts and software development techniques.

**Acknowledgement**

We would like to express our sincere gratitude to our respected tutor, Deepak Kumar Karna, for providing us with the opportunity to work on this project and for guiding us throughout the development process. His valuable feedback and support were instrumental in the successful completion of this project. We would also like to thank the British College for providing us with the necessary resources and facilities to undertake this project. Additionally, we extend our appreciation to our classmates for their constructive criticism and encouragement.

Finally, we acknowledge the contributions of all individuals whose work has influenced our project, whether directly or indirectly. Their insights and innovations have enriched our understanding and inspired our approach to software development.

**Table of Contents**  **Pg. n.o**

* Abstract 3
* Acknowledgement 4
* Student Profile System code (6-25)
* Project Objective 26
* Information of project (27-32)

**Student Profile System Code**

import csv

import tkinter as tk

from tkinter import messagebox

# Credentials for login

credentials = {'username': 'admin', 'password': 'admin123'}

# Fields for student data

student\_fields = ['roll', 'name', 'age', 'email', 'phone']

# File paths for databases

student\_database = 'user.txt'

grade\_database = 'grades.txt'

eca\_database = 'eca.txt'

# Function to create login window

def login\_window():

# Creating Tkinter window for login

login\_window = tk.Tk()

login\_window.title("Login")

# Function to handle login attempt

def handle\_login():

username = username\_entry.get()

password = password\_entry.get()

if username == credentials['username'] and password == credentials['password']:

messagebox.showinfo("Login Successful", "Login successful!")

login\_window.destroy()

main\_menu()

else:

messagebox.showerror("Login Failed", "Invalid username or password")

# Username entry field

username\_label = tk.Label(login\_window, text="Username:")

username\_label.pack()

username\_entry = tk.Entry(login\_window)

username\_entry.pack()

# Password entry field

password\_label = tk.Label(login\_window, text="Password:")

password\_label.pack()

password\_entry = tk.Entry(login\_window, show="\*")

password\_entry.pack()

# Login button

login\_button = tk.Button(login\_window, text="Login", command=handle\_login)

login\_button.pack()

login\_window.mainloop()

# Function to create main menu window

def main\_menu():

# Creating Tkinter window for main menu

main\_menu = tk.Tk()

main\_menu.title("Student Database Management System")

# Function to handle menu choices

def handle\_choice(choice):

if choice == 'Add New Student':

add\_student\_window()

elif choice == 'View Students':

view\_students\_window()

elif choice == 'Search Student':

search\_student\_window()

elif choice == 'Update Student':

update\_student\_window()

elif choice == 'Delete Student':

delete\_student\_window()

elif choice == 'Add Grades':

add\_grades\_window()

elif choice == 'View Grades':

view\_grades\_window()

elif choice == 'Add ECA':

add\_eca\_window()

elif choice == 'View ECA':

view\_eca\_window()

else:

main\_menu.destroy()

choices = [

"Add New Student",

"View Students",

"Search Student",

"Update Student",

"Delete Student",

"Add Grades",

"View Grades",

"Add ECA",

"View ECA",

"Quit"

]

# Creating buttons for menu choices

for choice in choices:

button = tk.Button(main\_menu, text=choice, command=lambda c=choice: handle\_choice(c))

button.pack()

main\_menu.mainloop()

# Add New Student Window

def add\_student\_window():

# Creating Tkinter window for adding new student

add\_student\_window = tk.Tk()

add\_student\_window.title("Add New Student")

# Function to save student data

def save\_student():

student\_data = [roll\_entry.get(), name\_entry.get(), age\_entry.get(), email\_entry.get(), phone\_entry.get()]

save\_student\_data(student\_data)

messagebox.showinfo("Success", "Student added successfully")

add\_student\_window.destroy()

# Entry fields for student data

roll\_label = tk.Label(add\_student\_window, text="Roll:")

roll\_label.pack()

roll\_entry = tk.Entry(add\_student\_window)

roll\_entry.pack()

name\_label = tk.Label(add\_student\_window, text="Name:")

name\_label.pack()

name\_entry = tk.Entry(add\_student\_window)

name\_entry.pack()

age\_label = tk.Label(add\_student\_window, text="Age:")

age\_label.pack()

age\_entry = tk.Entry(add\_student\_window)

age\_entry.pack()

email\_label = tk.Label(add\_student\_window, text="Email:")

email\_label.pack()

email\_entry = tk.Entry(add\_student\_window)

email\_entry.pack()

phone\_label = tk.Label(add\_student\_window, text="Phone:")

phone\_label.pack()

phone\_entry = tk.Entry(add\_student\_window)

phone\_entry.pack()

save\_button = tk.Button(add\_student\_window, text="Save", command=save\_student)

save\_button.pack()

add\_student\_window.mainloop()

# View Students Window

def view\_students\_window():

# Creating Tkinter window for viewing students

view\_students\_window = tk.Tk()

view\_students\_window.title("View Students")

view\_students\_window.geometry("800x600") # Increase window size

students\_list = tk.Listbox(view\_students\_window, width=100, height=30) # Increase listbox size

students\_list.pack()

# Populate listbox with student data

with open(student\_database, "r", encoding="utf-8") as f:

reader = csv.reader(f)

for row in reader:

students\_list.insert(tk.END, " | ".join(row))

view\_students\_window.mainloop()

# Search Student Window

def search\_student\_window():

# Creating Tkinter window for searching student

search\_student\_window = tk.Tk()

search\_student\_window.title("Search Student")

# Function to search student

def search\_student():

roll = roll\_entry.get()

with open(student\_database, "r", encoding="utf-8") as f:

reader = csv.reader(f)

for row in reader:

if row and row[0] == roll:

messagebox.showinfo("Student Found", "Name: {}\nAge: {}\nEmail: {}\nPhone: {}".format(row[1], row[2], row[3], row[4]))

return

messagebox.showinfo("Student Not Found", "Student with roll {} not found.".format(roll))

roll\_label = tk.Label(search\_student\_window, text="Roll:")

roll\_label.pack()

roll\_entry = tk.Entry(search\_student\_window)

roll\_entry.pack()

search\_button = tk.Button(search\_student\_window, text="Search", command=search\_student)

search\_button.pack()

search\_student\_window.mainloop()

# Update Student Window

def update\_student\_window():

# Creating Tkinter window for updating student

update\_student\_window = tk.Tk()

update\_student\_window.title("Update Student")

# Function to update student data

def update\_student():

roll = roll\_entry.get()

updated\_data = [name\_entry.get(), age\_entry.get(), email\_entry.get(), phone\_entry.get()]

rows = []

with open(student\_database, "r", encoding="utf-8") as f:

reader = csv.reader(f)

for row in reader:

if row and row[0] == roll:

rows.append(updated\_data)

else:

rows.append(row)

with open(student\_database, "w", newline='', encoding="utf-8") as f:

writer = csv.writer(f)

writer.writerows(rows)

messagebox.showinfo("Success", "Student with roll {} updated successfully.".format(roll))

update\_student\_window.destroy()

roll\_label = tk.Label(update\_student\_window, text="Roll:")

roll\_label.pack()

roll\_entry = tk.Entry(update\_student\_window)

roll\_entry.pack()

name\_label = tk.Label(update\_student\_window, text="Name:")

name\_label.pack()

name\_entry = tk.Entry(update\_student\_window)

name\_entry.pack()

age\_label = tk.Label(update\_student\_window, text="Age:")

age\_label.pack()

age\_entry = tk.Entry(update\_student\_window)

age\_entry.pack()

email\_label = tk.Label(update\_student\_window, text="Email:")

email\_label.pack()

email\_entry = tk.Entry(update\_student\_window)

email\_entry.pack()

phone\_label = tk.Label(update\_student\_window, text="Phone:")

phone\_label.pack()

phone\_entry = tk.Entry(update\_student\_window)

phone\_entry.pack()

update\_button = tk.Button(update\_student\_window, text="Update", command=update\_student)

update\_button.pack()

update\_student\_window.mainloop()

# Delete Student Window

def delete\_student\_window():

# Creating Tkinter window for deleting student

delete\_student\_window = tk.Tk()

delete\_student\_window.title("Delete Student")

# Function to delete student

def delete\_student():

roll = roll\_entry.get()

rows = []

with open(student\_database, "r", encoding="utf-8") as f:

reader = csv.reader(f)

for row in reader:

if row and row[0] == roll:

continue

rows.append(row)

with open(student\_database, "w", newline='', encoding="utf-8") as f:

writer = csv.writer(f)

writer.writerows(rows)

messagebox.showinfo("Success", "Student with roll {} deleted successfully.".format(roll))

delete\_student\_window.destroy()

roll\_label = tk.Label(delete\_student\_window, text="Roll:")

roll\_label.pack()

roll\_entry = tk.Entry(delete\_student\_window)

roll\_entry.pack()

delete\_button = tk.Button(delete\_student\_window, text="Delete", command=delete\_student)

delete\_button.pack()

delete\_student\_window.mainloop()

# Add Grades Window

def add\_grades\_window():

# Creating Tkinter window for adding grades

add\_grades\_window = tk.Tk()

add\_grades\_window.title("Add Grades")

# Function to save grades

def save\_grades():

roll = roll\_entry.get()

grade\_data = [

('Nepali', nepali\_entry.get()),

('Math', math\_entry.get()),

('Science', science\_entry.get()),

('English', english\_entry.get())

] # Example fields

save\_grade\_data(roll, grade\_data)

messagebox.showinfo("Success", "Grades added successfully")

add\_grades\_window.destroy()

roll\_label = tk.Label(add\_grades\_window, text="Roll:")

roll\_label.pack()

roll\_entry = tk.Entry(add\_grades\_window)

roll\_entry.pack()

nepali\_label = tk.Label(add\_grades\_window, text="Nepali Grade:")

nepali\_label.pack()

nepali\_entry = tk.Entry(add\_grades\_window)

nepali\_entry.pack()

math\_label = tk.Label(add\_grades\_window, text="Math Grade:")

math\_label.pack()

math\_entry = tk.Entry(add\_grades\_window)

math\_entry.pack()

science\_label = tk.Label(add\_grades\_window, text="Science Grade:")

science\_label.pack()

science\_entry = tk.Entry(add\_grades\_window)

science\_entry.pack()

english\_label = tk.Label(add\_grades\_window, text="English Grade:")

english\_label.pack()

english\_entry = tk.Entry(add\_grades\_window)

english\_entry.pack()

save\_button = tk.Button(add\_grades\_window, text="Save", command=save\_grades)

save\_button.pack()

add\_grades\_window.mainloop()

# View Grades Window

def view\_grades\_window():

# Creating Tkinter window for viewing grades

def delete\_grade():

selected\_index = grades\_list.curselection()

if selected\_index:

selected\_grade = grades\_list.get(selected\_index)

roll, \*grades = selected\_grade.split(" | ")

with open(grade\_database, "r", encoding="utf-8") as f:

reader = csv.reader(f)

rows = [row for row in reader if row and row[0] != roll]

with open(grade\_database, "w", newline='', encoding="utf-8") as f:

writer = csv.writer(f)

writer.writerows(rows)

messagebox.showinfo("Success", "Grade for student with roll {} deleted successfully.".format(roll))

view\_grades\_window.destroy()

view\_grades\_window()

view\_grades\_window = tk.Tk()

view\_grades\_window.title("View Grades")

grades\_list = tk.Listbox(view\_grades\_window, width=80, height=20)

grades\_list.pack(padx=10, pady=10)

# Populate listbox with grade data

with open(grade\_database, "r", encoding="utf-8") as f:

reader = csv.reader(f)

for row in reader:

grades\_list.insert(tk.END, " | ".join(row))

delete\_button = tk.Button(view\_grades\_window, text="Delete Selected Grade", command=delete\_grade)

delete\_button.pack()

view\_grades\_window.mainloop()

# Add ECA Window

def add\_eca\_window():

# Creating Tkinter window for adding ECA

add\_eca\_window = tk.Tk()

add\_eca\_window.title("Add ECA")

# Function to save ECA

def save\_eca():

roll = roll\_entry.get()

eca\_data = [

('Name', name\_entry.get()),

('ECA ', eca\_entry.get())

] # Example fields

save\_eca\_data(roll, eca\_data)

messagebox.showinfo("Success", "ECAs added successfully")

add\_eca\_window.destroy()

roll\_label = tk.Label(add\_eca\_window, text="Roll:")

roll\_label.pack()

roll\_entry = tk.Entry(add\_eca\_window)

roll\_entry.pack()

name\_label = tk.Label(add\_eca\_window, text="NAME:")

name\_label.pack()

name\_entry = tk.Entry(add\_eca\_window)

name\_entry.pack()

eca\_label = tk.Label(add\_eca\_window, text="ECA :")

eca\_label.pack()

eca\_entry = tk.Entry(add\_eca\_window)

eca\_entry.pack()

save\_button = tk.Button(add\_eca\_window, text="Save", command=save\_eca)

save\_button.pack()

add\_eca\_window.mainloop()

# View ECA Window

def view\_eca\_window():

# Creating Tkinter window for viewing ECA

def delete\_eca():

selected\_index = eca\_list.curselection()

if selected\_index:

selected\_eca = eca\_list.get(selected\_index)

roll, \*ecas = selected\_eca.split(" | ")

with open(eca\_database, "r", encoding="utf-8") as f:

reader = csv.reader(f)

rows = [row for row in reader if row and row[0] != roll]

with open(eca\_database, "w", newline='', encoding="utf-8") as f:

writer = csv.writer(f)

writer.writerows(rows)

messagebox.showinfo("Success", "ECA for student with roll {} deleted successfully.".format(roll))

view\_eca\_window.destroy()

view\_eca\_window()

view\_eca\_window = tk.Tk()

view\_eca\_window.title("View ECA")

eca\_list = tk.Listbox(view\_eca\_window, width=80, height=20)

eca\_list.pack(padx=10, pady=10)

# Populate listbox with ECA data

with open(eca\_database, "r", encoding="utf-8") as f:

reader = csv.reader(f)

for row in reader:

eca\_list.insert(tk.END, " | ".join(row))

delete\_button = tk.Button(view\_eca\_window, text="Delete Selected ECA", command=delete\_eca)

delete\_button.pack()

view\_eca\_window.mainloop()

# Function to save student data to file

def save\_student\_data(data):

with open(student\_database, "a", newline='', encoding="utf-8") as f:

writer = csv.writer(f)

writer.writerow(data)

# Function to save grade data to file

def save\_grade\_data(roll, data):

with open(grade\_database, "a", newline='', encoding="utf-8") as f:

writer = csv.writer(f)

for subject, grade in data:

writer.writerow([roll, subject, grade])

# Function to save ECA data to file

def save\_eca\_data(roll, data):

with open(eca\_database, "a", newline='', encoding="utf-8") as f:

writer = csv.writer(f)

for eca, name in data:

writer.writerow([roll, eca, name])

# Starting point of the program

login\_window()

# Displaying thank you message

print("-------------------------------")

print(" Thank you for using our system")

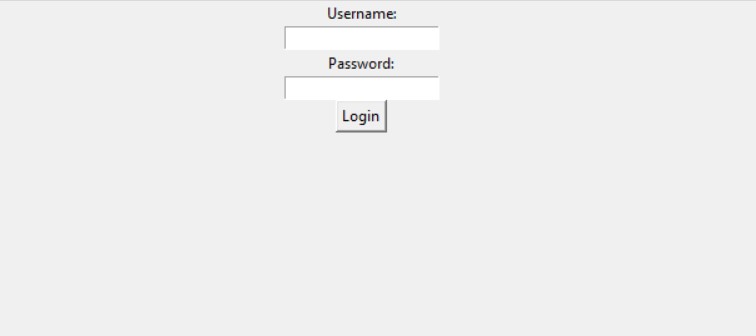
print("-------------------------------")

**Objective**

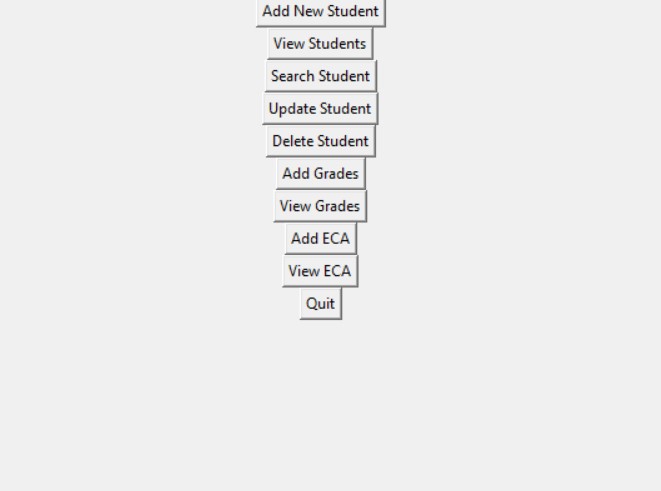
The objective of this project is to design and develop a comprehensive Student Profile System for the Fundamentals of Data Science Level 3 course at The British College. The system aims to facilitate efficient management of student information, academic records, and extracurricular activities (ECA) through the integration of fundamental data science principles and software development methodologies. By implementing object-oriented programming, modular design, file handling, and error handling techniques, our objective is to create a user-friendly and robust application that meets the specified requirements outlined in the project brief. Through this project, we seek to demonstrate our proficiency in applying theoretical concepts to practical software development tasks, thereby enhancing our understanding of data science fundamentals and honing our programming skills.

**Information of Project**

The interface with a option of prompting user to enter username and password appears.



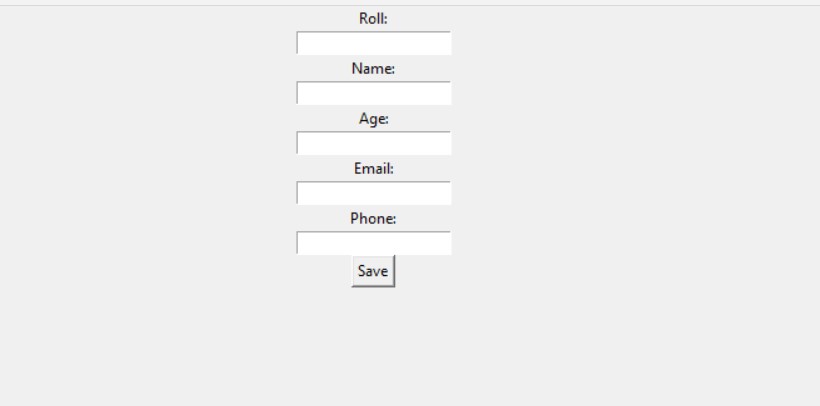
After the successful Login , the terminal displays various options regarding the students.



**Procedures: -**

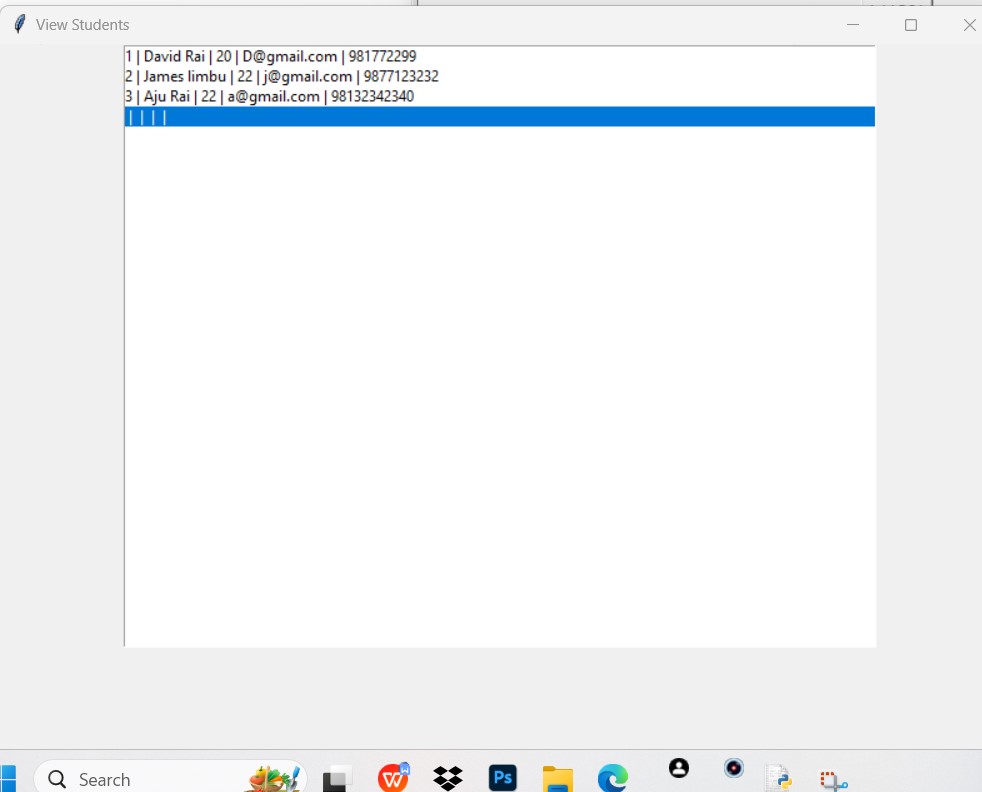
**Adding New Students:**

A specific form will display beneath all fields should be filled out completely. If there is a mistake in the entry process, the data won't be saved in the file.



**Displaying Students:**

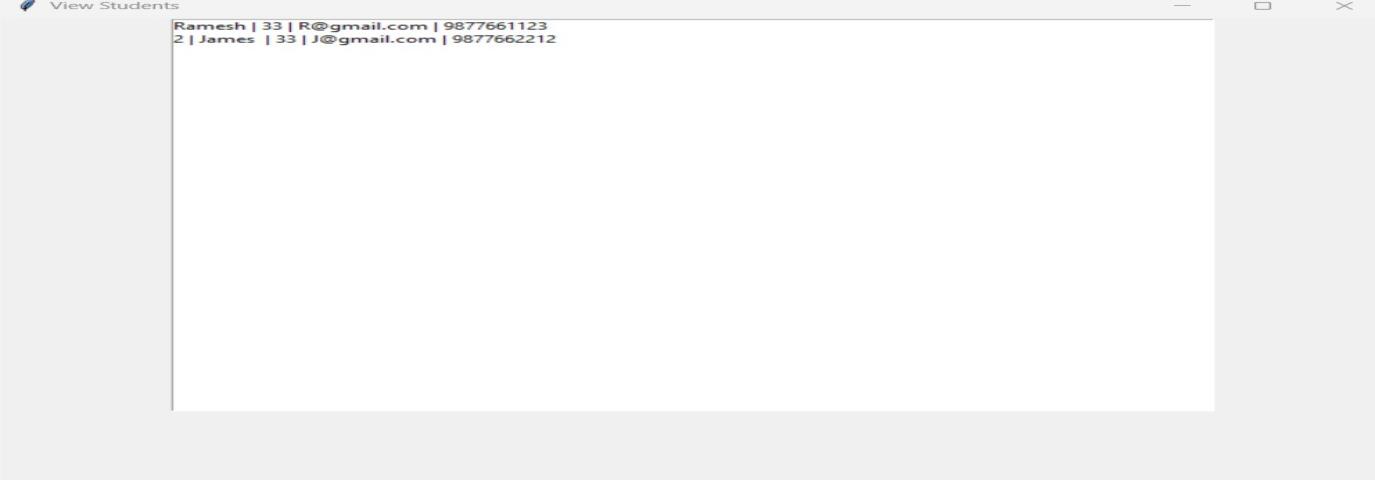
In this section, the total no. of added students is displayed where the user can enter multiple number of students.



**Update and Display Student Records:**

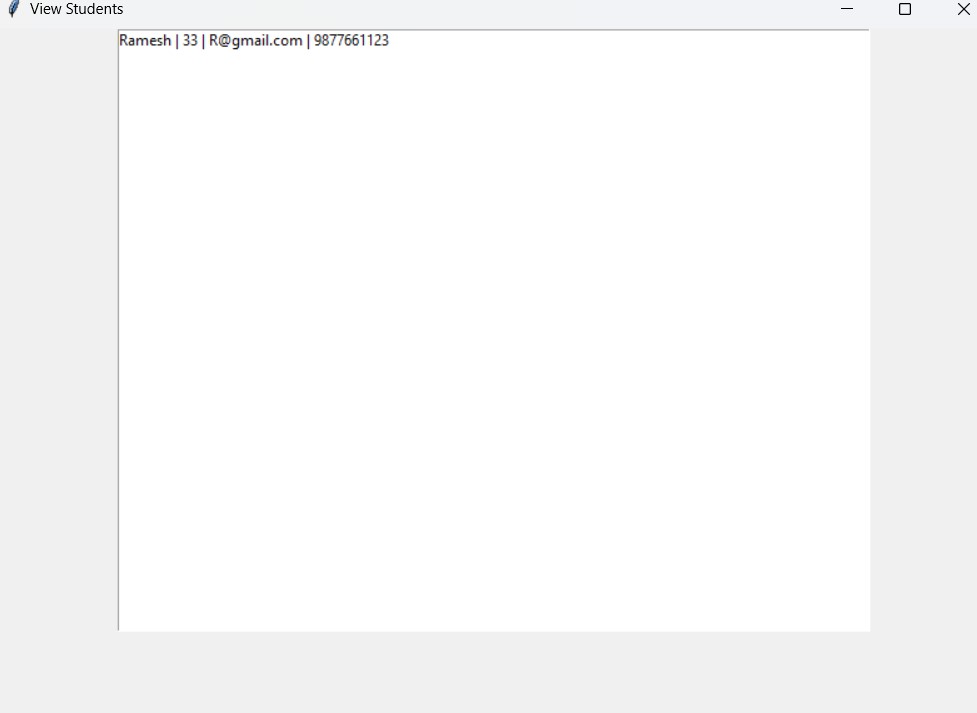
This system has the capacity to carry out all of the tasks necessary to build a successful student profile system where the user can add, update and delete the record of the students.

After updating the Student with roll no.1, the following result is displayed.



.

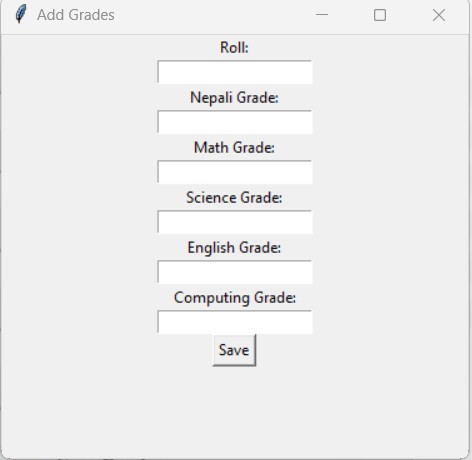
Similarly, After deleting the student with roll no.2 , the following result is obtained.



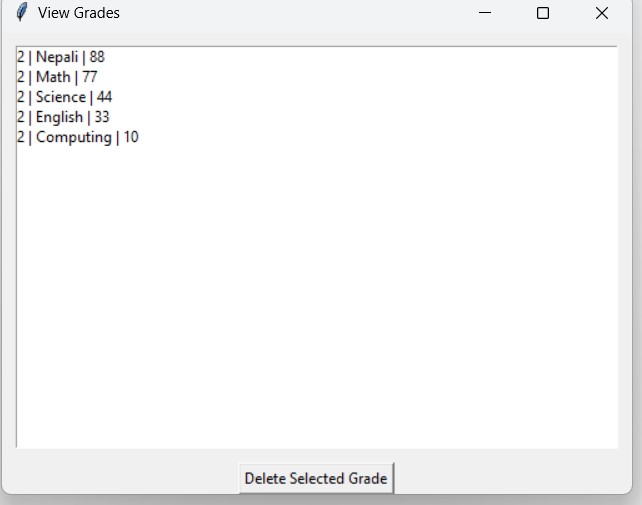
**Adding and displaying Grades:**

The program allows the user to add grades in accordance to the roll no.and the program also has a provision of deleting the grades.

The program prompts the user to add marks to different subjects relating to the roll no. of the students.

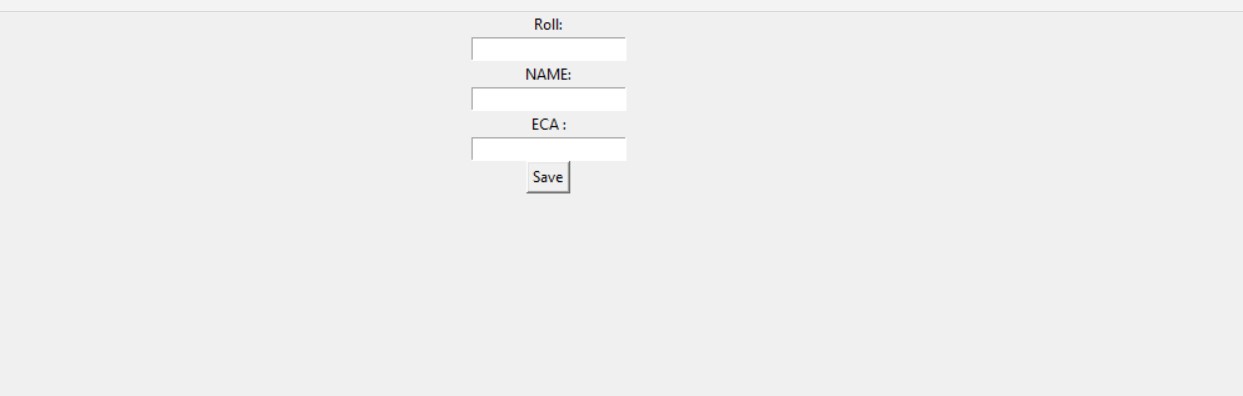


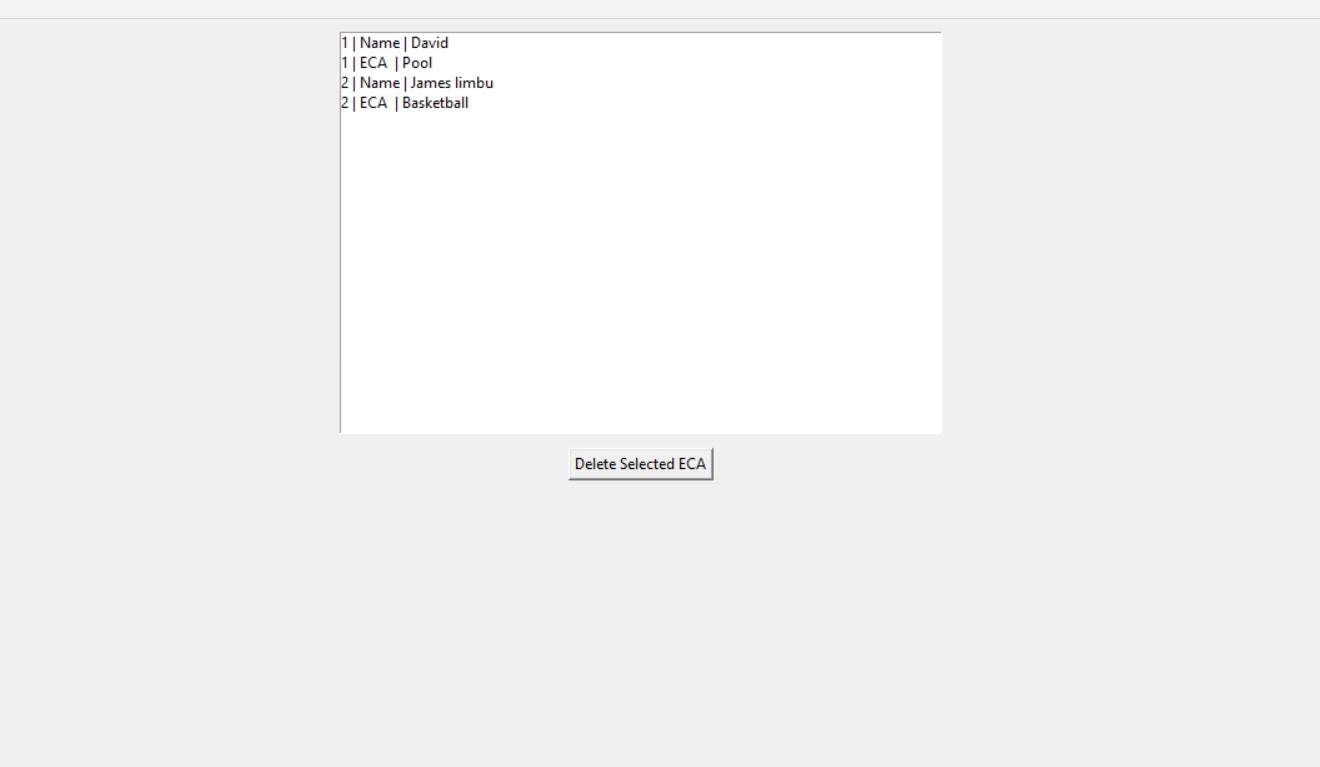
The program displays the total marks obtained by the students in accordance to their roll nos.



**Adding and displaying ECA:**

The program has another function of displaying the ECA of each students according to their respective roll nos.

****



**Exit**

To log out of the Student Profile system, choose quit and then you will get our of the pr

