Exam Result System

Project Report submitted in partial fulfillment of the requirement of ICTC 1101.3 and Introduction to Programming of the degree of BSc in Information Technology

Name – Y. Amadi Anuththara

Department – Department of Information and communication Technology

Contents

1. System Introduction

Exam Result system is a Python based project. The system presenting exam result of the student and calculate relevant results for analyze exam results. This project report is introduced how built exam result system using the Python.

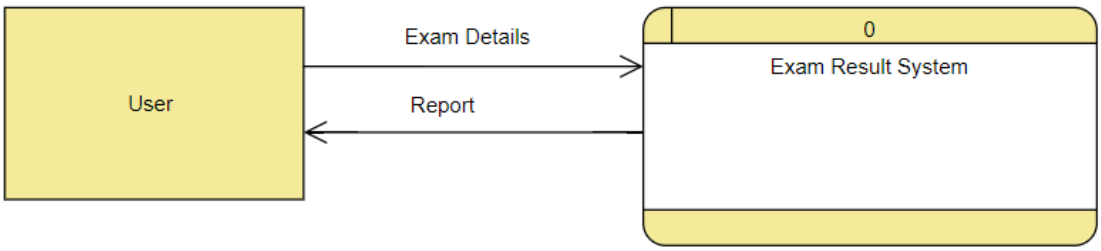
Student (user) can login to the system using their relevant details. Student have to input their user name and password for login to the system. User input data into the system. After It navigate the system in to different levels. This Exam Result System has three subsystem each of which has similar functionality and similar architecture.

There is main three levels in the system. First one is Primary Level, second one is Ordinary Level, third one is Advance Level.

After that exam results separately process through the system. Then we can calculate its total, average, for relevant level for each exam term or several. Finally, we can analyze this result for get direct idea.

Student examination and result computation systems can prove useful for educational institutions and other examination testing bodies. This exam result system is very helpful to get their exam result report relevant level with analysis. Student can understand their level according to exam results.

Student exam result systems can effectively take care of every aspect of exam results and of course faster and accurate result processing are some of the key features of the system.



2. System Objectives

1) Analyse Result

Exam result system is useful to calculate the exam result of the student. This system can be able to calculate total and average relevant student exam results.

2) Improvement on control and performance

This system is developed to analyse exam result. It is help to get direct idea about the exam result.

3) Save time

Student can get total and average easily and quickly. Then they can understand their position relevant the result.

4) Save cost

Student can use this when they want to get their analysis of result. This system hasn't limit. It works at any number of times

5) Store data

Exam result system is very useable system for student result records.

3. Functional Requirement

01. Collect data

Exam result system based on new data. We include Exam result of the student as an input. After that this new collecting data is process to get analysis result of the student.

02. Calculate data

Exam result of the student is calculated to perform student analysis for a term or multiple terms. It calculates the total of the subjects. After that find an average and grade. This calculation is helpful for the final analysis.

03. Analyze data

We need to analysis data to get direct idea about the system. It helps to get direct idea about the advance calculation.

04. Store data

Collected data of the system is update to gather a new data from the user. If we can add data and deleted the data, then we can update the system. This updated data store in a database.

05. Update data

Exam result system has a data base for store a data. Store data used to calculate and analysis to the exam results. We can store Student details, student result and student report into the data base.

4.Non - Functional Requirement

01. Usability

Exam result system is usable for perform the student result with its analysis. This system is simple to understand. Student can get result of the term several terms without a support.

02. Reliability

It can use continuously as much as it wants by a student to calculate and analyze their exam results.

03. Performance

Exam result system perform clearly without affecting user experience and all guidelines are given by the system to follow the program.

04. Availability

This is available for any kind of user who still in the school at any time when it needs.

05. Security

At the beginning system takes user details to verify the user. Therefore, student data are protected through continue the system security.

5. Software tools used

The whole project is divided in two parts the front end and the backend.

- Front end

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

The front end is designed using of Python. Python is used in many application domains. The Python Package Index lists thousands of third-party modules for Python. It can be used as a scripting language or can be compiled to byte-code for building large applications. It provides very high-level dynamic data types and supports dynamic type checking. Famous Desktop GUI is Tkinter in python. Some Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit. toolkits that are usable on several platforms are available separately:

Create the GUI application main window.

Python has a lot of GUI frameworks, but Tkinter is the only framework that's built into the Python standard library. Tkinter has several strengths. Visual elements are rendered using native operating system elements, so applications built with Tkinter look like they belong on the platform where they're run. Although Tkinter is considered the de-facto Python GUI framework, it's not without criticism. One notable criticism is that GUIs built with Tkinter look outdated. To develop GUI applications Tkinter provides widgets that are used to represent in the browser. Tkinter widgets are objects that are eventful and are represented in the browser such as textbox, button, etc. In Python, Tkinter widgets are standard GUI elements that are used for handling events through elements like button, frames, labels, etc.

- Backend

The backend is designed using SQLite which is used to design the databases

SQLite in general is a server-less database that we can use within almost all programming languages including Python. Server-less means there is no need to install a separate server to work with SQLite so we can connect directly with the database. SQLite is a lightweight database that can provide a relational database management system with zero-configuration because there is no need to configure or set up anything to use it.

SQLite is a self-contained, high-reliability, embedded, full-featured, public-domain, SQL database engine. It is the most used database engine on the world wide web. Python has a library to access SQLite databases, called sqlite3, intended for working with this database which has been included with Python package since version 2.5.

	StuID	StuName	Subject1	Subject2	Subject3	Subject4	Subject5	Subject6	Total	Average
1	100	Nimal	45	56	98	87	78	78	442	73.6666666…
2	10005	asdfg	78	45	67	67	56	56	369	61.5
3	10006	Nimali	78	89	67	78	56	98	466	77.6666666…
4	10008	Samadi	78	67	80	70	78	56	429	71.5
5	10050	Vimansa	78	98	70	67	78	80	471	78.5
6	10678	Tharuka	77	80	75	70	71	83	456	76
7	12345	adfdfg	23	34	45	45	45	76	268	44.6666666…

6. Frontend

6.1. Create the graphical user interface

This exam result system was created using four interfaces through a Tkinter in python. Exam result system main window, Primary window, Ordinary window and Advance window are created windows in this system.

6.1.1. Main window

Main window is the main part of the system. It was created using frames, buttons, entries, picture and etc. I used fames for tittle, picture, student name, student id and level buttons. There are three level buttons in this main window. Such as primary Level, Ordinary Level and Advance Level.

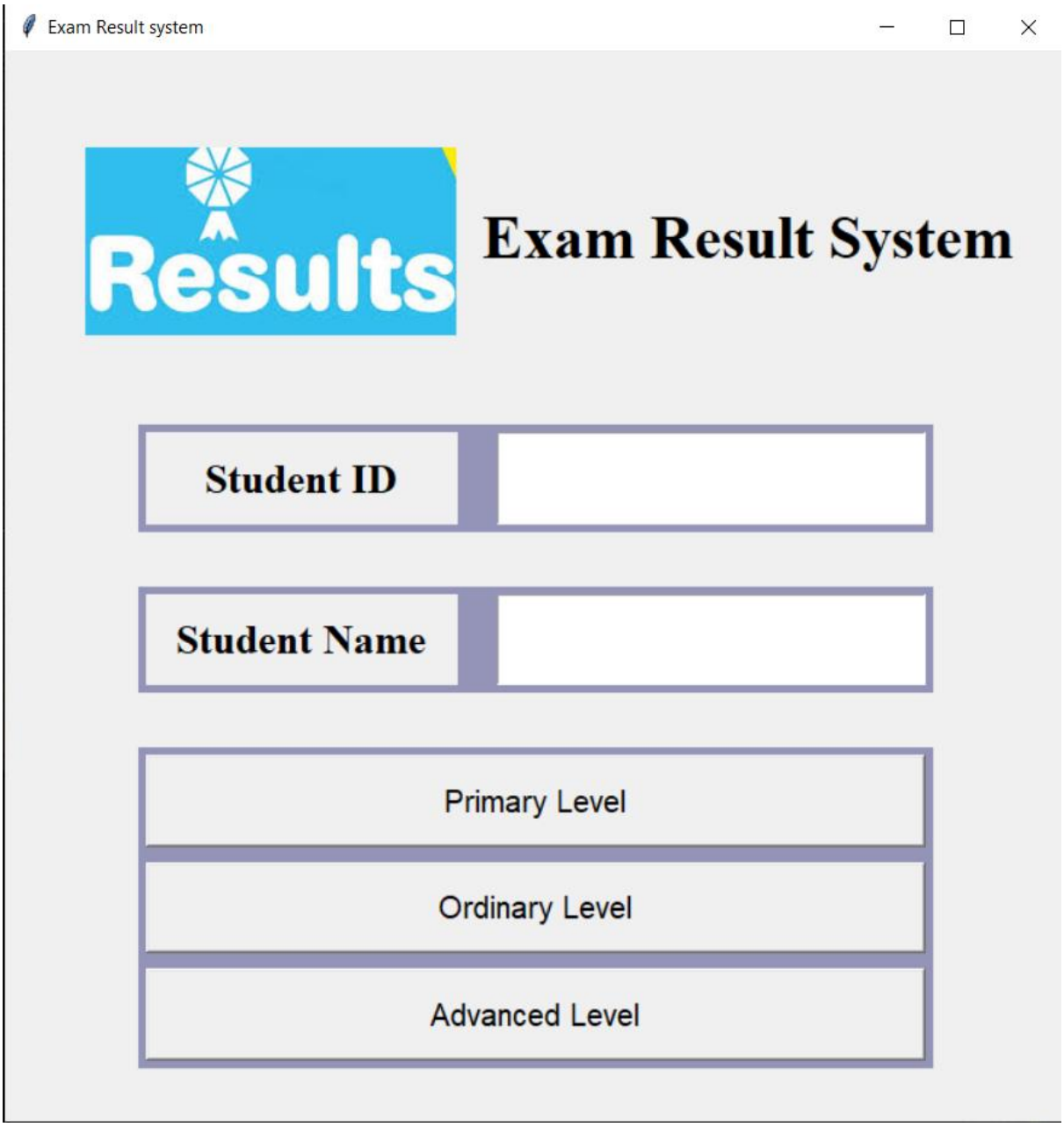
I used two entries to get user input data. Student Id and Student Name are entries of this main window.

I used following codes to create main window:

```
class MainWindow:

    def Image(self):
        self.Photo =PhotoImage(file="jpg.png")
        return self.Photo

    def __init__(self, master):


        StudentID = StringVar()
        StudentName = StringVar()


        self.master = master
        self.master.title("Exam Result system")

        # Main Text
        self.frameTitleMainText = tk.Frame(self.master)
        self.frameTitleMainText.place(relx=0.7, rely=0.05,                    relwidth=0.85,
relheight=0.25, anchor='n')

        self.labelTitleTextMain = tk.Label(self.frameTitleMainText,                    text="Exam Result
System", font="Times 30 bold")
```

```
        self.labelTitleTextMain.place(relwidth=1, relheigh=1)

        self.framePhoto = tk.Frame(self.master)
        self.framePhoto.place(relx=0.25, rely=0.09, relwidth=0.35,   relheight=0.35, anchor='n')

        self.labelPhoto = tk.Label(self.framePhoto, image=self.Image())
        self.labelPhoto.place(relwidth=1, relheigh=0.5)

        # Student Id
        self.frameStuID = tk.Frame(self.master, bg='#9494b8', bd=5)
        self.frameStuID.place(relx=0.5, rely=0.35, relwidth=0.75,                relheight=0.1, anchor='n')

        self.labelStuID = tk.Label(self.frameStuID, text="Student ID",            font="Times 20 bold")
        self.labelStuID.place(relwidth=0.4, relheight=1)

        self.entryStuID = tk.Entry(self.frameStuID,
                textvariable=StudentID)
        self.entryStuID.place(relx=0.45, relwidth=0.55, relheight=1)
        # Student Name
        self.frameStuName = tk.Frame(self.master, bg='#9494b8', bd=5)
        self.frameStuName.place(relx=0.5, rely=0.5, relwidth=0.75,                          relheight=0.1,
anchor='n')

        self.labelStuName = tk.Label(self.frameStuName, text="Student                        Name",
font="Times 20 bold")
        self.labelStuName.place(relwidth=0.4, relheight=1)

        self.entryStuName = tk.Entry(self.frameStuName,
                textvariable=StudentName)
        self.entryStuName.place(relx=0.45, relwidth=0.55, relheight=1)

        # Buttons (Primary,Ordinary,Advance)
        self.frameButtonMain = tk.Frame(self.master, bg='#9494b8',                          bd=5)
        self.frameButtonMain.place(relx=0.5, rely=0.65, relwidth=0.75,           relheight=0.3, anchor='n')

        self.buttonPrimary = tk.Button(self.frameButtonMain,                                text="Primary
Level", font=40,
                                command=lambda:
                                        self.Login_System_Primary())
        self.buttonPrimary.place(relheight=0.3, relwidth=1)

        self.buttonOrdinary = tk.Button(self.frameButtonMain,                               text="Ordinary
Level", font=40,
                                command=lambda:
                            self.Login_System_Ordinary())
        self.buttonOrdinary.place(rely=0.35, relheight=0.3, relwidth=1)

        self.buttonAdvanced = tk.Button(self.frameButtonMain,                               text="Advanced
Level", font=40,
                                command=lambda:
                                        self.Login_System_Advance())
        self.buttonAdvanced.place(rely=0.7, relheight=0.3, relwidth=1)
```

- Main Window user Interface



6.1.2. Primary window

Student can enter the primary window using the primary window button. I used frames for primary level tittle, subjects, total, average and Analyse button. There is a analyse button in this primary window. It helps to calculate total and average.

I used entries to get Subject results from the user (Student). There are 6 entries of this primary window. I used ten labels to input the text into the primary window.

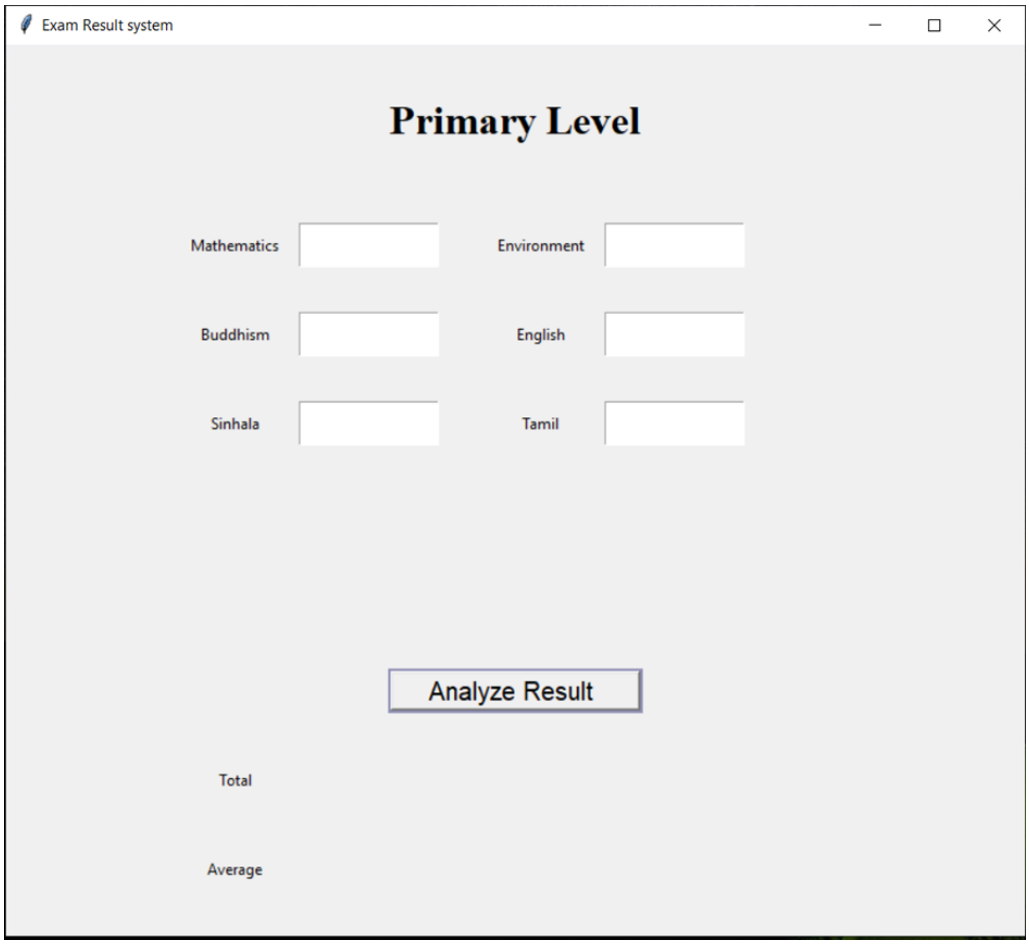
I used following codes to create Primary window:

```
class windowPrimary:

    def CalculateTotal(self):
        sum = int(self.entrySubject1.get()) + int(self.entrySubject2.get()) + int(self.entrySubject3.get()) + \
            int(self.entrySubject4.get()) + int(self.entrySubject5.get()) + int(self.entrySubject6.get())
        return sum

    def CalculateAverage(self):
        average = self.CalculateTotal()/6
        return average


    def analyseData(self):
        print("Total =", self.CalculateTotal())
        print("Average = ", self.CalculateAverage())
        self.textTotal.set(self.CalculateTotal())
        self.textAverage.set(self.CalculateAverage())
        databse.InsertPrimaryData(self.StudentID, self.StudentName,
                int(self.entrySubject1.get()), \
                        int(self.entrySubject2.get()),          int(self.entrySubject3.get()), \
                        int(self.entrySubject4.get()),          int(self.entrySubject5.get()), \
                        int(self.entrySubject6.get()),  self.CalculateTotal(), self.CalculateAverage())



    def __init__(self, master, StudentID, StudentName):
        self.master = master

        self.textTotal = StringVar()
        self.textAverage = StringVar()

        self.StudentID = StudentID
        self.StudentName = StudentName

        # Primary Title
        self.frameTitleText = tk.Frame(self.master)
        self.frameTitleText.place(relx=0.5, rely=0.01, relwidth=0.65,          relheight=0.15, anchor='n')

        self.labelTitleText = tk.Label(self.frameTitleText,                          text="Primary
Level", font="Times 24 bold")

        self.labelTitleText.place(relwidth=1, relheigh=1)

        # subjects
        self.frameSubject1 = tk.Frame(self.master)
        self.frameSubject1.place(relx=0.3, rely=0.20, relwidth=0.25,                 relheight=0.05,
anchor='n')

        self.labelSubject1 = tk.Label(self.frameSubject1,
                text="Mathematics")
        self.labelSubject1.place(relwidth=0.4, relheight=1)

        self.entrySubject1 = tk.Entry(self.frameSubject1)
```

```
        self.entrySubject1.place(relx=0.45, relwidth=0.55, relheight=1)

        self.frameSubject2 = tk.Frame(self.master)
        self.frameSubject2.place(relx=0.3, rely=0.30, relwidth=0.25,
                relheight=0.05, anchor='n')

        self.labelSubject2 = tk.Label(self.frameSubject2,
                text="Buddhism")
        self.labelSubject2.place(relwidth=0.4, relheight=1)

        self.entrySubject2 = tk.Entry(self.frameSubject2)
        self.entrySubject2.place(relx=0.45, relwidth=0.55, relheight=1)

        self.frameSubject3 = tk.Frame(self.master)
        self.frameSubject3.place(relx=0.3, rely=0.40, relwidth=0.25,
relheight=0.05, anchor='n')

        self.labelSubject3 = tk.Label(self.frameSubject3,
                text="Sinhala")
        self.labelSubject3.place(relwidth=0.4, relheight=1)

        self.entrySubject3 = tk.Entry(self.frameSubject3)
        self.entrySubject3.place(relx=0.45, relwidth=0.55, relheight=1)

        self.frameSubject4 = tk.Frame(self.master)
        self.frameSubject4.place(relx=0.6, rely=0.20, relwidth=0.25,                relheight=0.05, anchor='n')

        self.labelSubject4 = tk.Label(self.frameSubject4,
                text="Environment")
        self.labelSubject4.place(relwidth=0.4, relheight=1)

        self.entrySubject4 = tk.Entry(self.frameSubject4)
        self.entrySubject4.place(relx=0.45, relwidth=0.55, relheight=1)

        self.frameSubject5 = tk.Frame(self.master)
        self.frameSubject5.place(relx=0.6, rely=0.30, relwidth=0.25,                                relheight=0.05,
anchor='n')

        self.labelSubject5 = tk.Label(self.frameSubject5,
                text="English")
        self.labelSubject5.place(relwidth=0.4, relheight=1)

        self.entrySubject5 = tk.Entry(self.frameSubject5)
        self.entrySubject5.place(relx=0.45, relwidth=0.55, relheight=1)

        self.frameSubject6 = tk.Frame(self.master)
        self.frameSubject6.place(relx=0.6, rely=0.40, relwidth=0.25,                relheight=0.05, anchor='n')

        self.labelSubject6 = tk.Label(self.frameSubject6, text="Tamil")
        self.labelSubject6.place(relwidth=0.4, relheight=1)

        self.entrySubject6 = tk.Entry(self.frameSubject6)
        self.entrySubject6.place(relx=0.45, relwidth=0.55, relheight=1)

        # Button Analyze
        self.frameButtonAnalyze = tk.Frame(self.master, bg='#9494b8', bd=2)
        self.frameButtonAnalyze.place(relx=0.5, rely=0.70, relwidth=0.25,      relheight=0.05, anchor='n')
```

```
self.buttonPrimaryWinAna = tk.Button(self.frameButtonAnalyze, text="Analyze  Result ", font=20,
                                     command=lambda:
                                         self.analyseData())
self.buttonPrimaryWinAna.place(relheight=1, relwidth=1)


# Total and average
self.frameTotal = tk.Frame(self.master)
self.frameTotal.place(relx=0.3, rely=0.80, relwidth=0.25,                relheight=0.05, anchor='n')

self.labelTotal = tk.Label(self.frameTotal, text="Total")
self.labelTotal.place(relwidth=0.4, relheight=1)

self.labelEntryTotal = tk.Label(self.frameTotal,
        textvariable=self.textTotal)
self.labelEntryTotal.place(relx=0.45, relwidth=0.55,                                relheight=1)

self.frameAverage = tk.Frame(self.master)
self.frameAverage.place(relx=0.3, rely=0.90, relwidth=0.25,              relheight=0.05, anchor='n')

self.labelAverage = tk.Label(self.frameAverage, text="Average")
self.labelAverage.place(relwidth=0.4, relheight=1)

self.labelEntryAverage = tk.Label(self.frameAverage,
        textvariable=self.textAverage)
self.labelEntryAverage.place(relx=0.45, relwidth=0.55,                              relheight=1)
```

- Primary Level User Interface



6.1.3. Ordinary window

Student can enter the Ordinary window using the Ordinary window button. I used fames for Ordinary level tittle, subjects, total, average and Analyse button. There is a analyse button in this Ordinary window. It helps to calculate total and average.

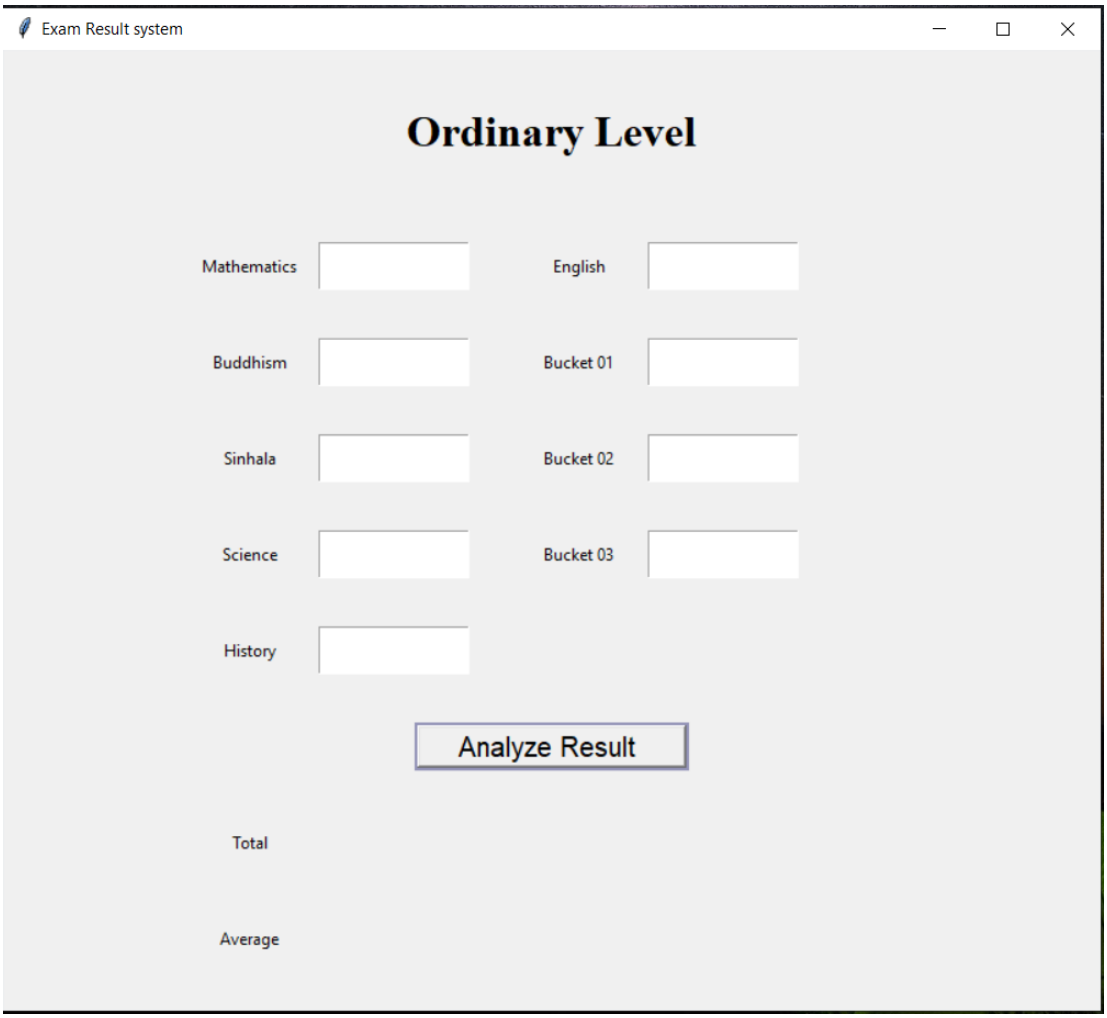
I used entries to get Subject results from the user (Student). There are 9 entries of this Ordinary window. I used thirteen labels to input the text into the Ordinary window.

I used following codes to create Ordinary Level window:

```
class windowAdvance:

    def CalculateTotal(self):
        sum = int(self.entrySubject1.get()) +                    int(self.entrySubject2.get()) +
                int(self.entrySubject3.get())
        return sum

    def CalculateAverage(self):
        average = self.CalculateTotal() / 3
        return average

    def analyseData(self):
        print("Total =", self.CalculateTotal())
        print("Average = ", self.CalculateAverage())
        self.textTotal.set(self.CalculateTotal())
        self.textAverage.set(self.CalculateAverage())
        databse.InsertAdvancedData(self.StudentID, self.StudentName,
                self.entrySubStream.get(), \
                        int(self.entrySubject1.get()),           int(self.entrySubject2.get()), \
                        int(self.entrySubject3.get()),                    self.CalculateTotal(),
self.CalculateAverage())


    def __init__(self, master, StudentID, StudentName):
        self.master = master

        self.textTotal = StringVar()
        self.textAverage = StringVar()

        self.StudentID = StudentID
        self.StudentName = StudentName

        # Advance Title
        self.frameTitleText = tk.Frame(self.master)
        self.frameTitleText.place(relx=0.5, rely=0.01, relwidth=0.65, relheight=0.15, anchor='n')

        self.labelTitleText = tk.Label(self.frameTitleText, text="Advance Level", font="Times 24 bold")
        self.labelTitleText.place(relwidth=1, relheigh=1)

        self.frameSubStream = tk.Frame(self.master)
        self.frameSubStream.place(relx=0.4, rely=0.20, relwidth=0.55, relheight=0.05, anchor='n')

        self.labelSubStream = tk.Label(self.frameSubStream, text="Subject Stream")
        self.labelSubStream.place(relwidth=0.3, relheight=1)

        self.entrySubStream = tk.Entry(self.frameSubStream, textvariable=StudentName)
        self.entrySubStream.place(relx=0.30, relwidth=0.55, relheight=1)


        # subjects
        self.frameSubject1 = tk.Frame(self.master)
        self.frameSubject1.place(relx=0.3, rely=0.30, relwidth=0.25, relheight=0.05, anchor='n')
```

```
        self.labelSubject1 = tk.Label(self.frameSubject1, text="Subject 01")
        self.labelSubject1.place(relwidth=0.4, relheight=1)

        self.entrySubject1 = tk.Entry(self.frameSubject1)
        self.entrySubject1.place(relx=0.45, relwidth=0.55, relheight=1)

        self.frameSubject2 = tk.Frame(self.master)
        self.frameSubject2.place(relx=0.3, rely=0.40, relwidth=0.25, relheight=0.05, anchor='n')

        self.labelSubject2 = tk.Label(self.frameSubject2, text="Subject 02")
        self.labelSubject2.place(relwidth=0.4, relheight=1)

        self.entrySubject2 = tk.Entry(self.frameSubject2)
        self.entrySubject2.place(relx=0.45, relwidth=0.55, relheight=1)

        self.frameSubject3 = tk.Frame(self.master)
        self.frameSubject3.place(relx=0.3, rely=0.50,
                                 relwidth=0.25,          relheight=0.05, anchor='n')

        self.labelSubject3 = tk.Label(self.frameSubject3,
                                 text="Subject 03")
        self.labelSubject3.place(relwidth=0.4, relheight=1)

        self.entrySubject3 = tk.Entry(self.frameSubject3)
        self.entrySubject3.place(relx=0.45, relwidth=0.55, relheight=1)

        # Button Analyze

        self.frameButtonAnalyze = tk.Frame(self.master, bg='#9494b8', bd=2)
        self.frameButtonAnalyze.place(relx=0.5, rely=0.70, relwidth=0.25, relheight=0.05, anchor='n')

        self.buttonPrimaryWin = tk.Button(self.frameButtonAnalyze, text="Analyze Result ", font=20,
                                    command=lambda: self.analyseData())
        self.buttonPrimaryWin.place(relheight=1, relwidth=1)

        # Total and Average
        self.frameTotal = tk.Frame(self.master)
        self.frameTotal.place(relx=0.3, rely=0.80, relwidth=0.25, relheight=0.05, anchor='n')

        self.labelTotal = tk.Label(self.frameTotal, text="Total")
        self.labelTotal.place(relwidth=0.4, relheight=1)

        self.labelEntryTotal = tk.Label(self.frameTotal,
            textvariable=self.textTotal)
        self.labelEntryTotal.place(relx=0.45, relwidth=0.55, relheight=1)

        self.frameAverage = tk.Frame(self.master)
        self.frameAverage.place(relx=0.3, rely=0.90, relwidth=0.25, relheight=0.05, anchor='n')

        self.labelAverage = tk.Label(self.frameAverage, text="Average")
        self.labelAverage.place(relwidth=0.4, relheight=1)

        self.labelEntryAverage = tk.Label(self.frameAverage,
            textvariable=self.textAverage)
        self.labelEntryAverage.place(relx=0.45, relwidth=0.55, relheight=1)
```

- Ordinary Level User Interface



6.1.4. Advance Level window

Student can enter the Advance Level window using the Advance Level window button. I used fames for Advance level tittle, subjects, total, average and Analyse button. There is a analyse button in this Advance Level window. It helps to calculate total and average.

I used entries to get Subject results from the user (Student). There are 4 entries of this primary window. I used eight labels to input the text into the primary window.

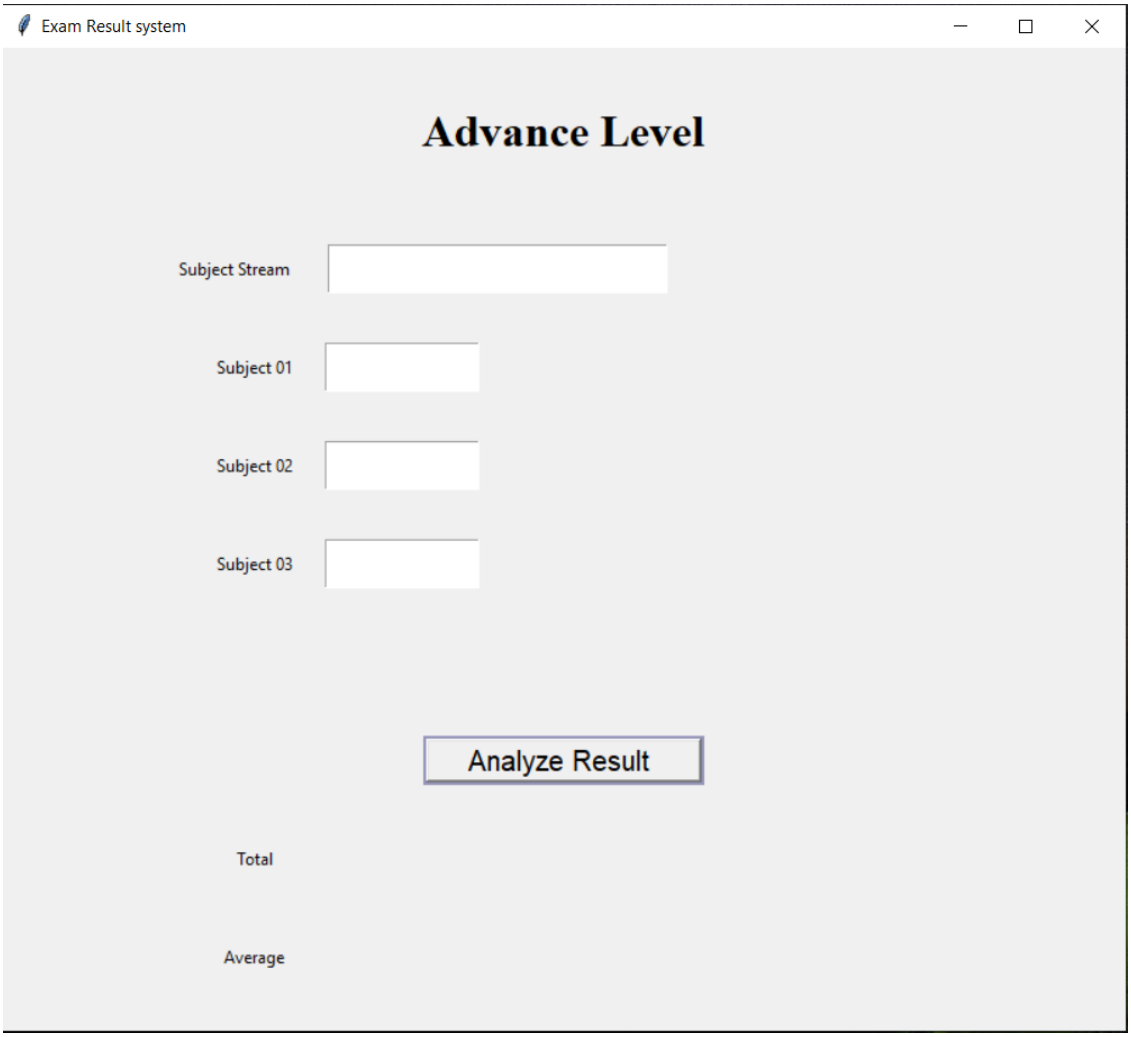
I used following codes to create Advance Level window:

```
class windowAdvance:

    def CalculateTotal(self):
        sum = int(self.entrySubject1.get()) +   int(self.entrySubject2.get()) + int(self.entrySubject3.get())
        return sum

    def CalculateAverage(self):
        average = self.CalculateTotal() / 3
        return average

    def analyseData(self):
        print("Total =", self.CalculateTotal())
        print("Average = ", self.CalculateAverage())
        self.textTotal.set(self.CalculateTotal())
        self.textAverage.set(self.CalculateAverage())
        databse.InsertAdvancedData(self.StudentID, self.StudentName,
                self.entrySubStream.get(), \
                        int(self.entrySubject1.get()), int(self.entrySubject2.get()), \
                        int(self.entrySubject3.get()),
                self.CalculateTotal(),self.CalculateAverage())


    def __init__(self, master, StudentID, StudentName):
        self.master = master

        self.textTotal = StringVar()
        self.textAverage = StringVar()

        self.StudentID = StudentID
        self.StudentName = StudentName

        # Advance Title
        self.frameTitleText = tk.Frame(self.master)
        self.frameTitleText.place(relx=0.5, rely=0.01, relwidth=0.65, relheight=0.15, anchor='n')

        self.labelTitleText = tk.Label(self.frameTitleText, text="Advance Level", font="Times 24 bold")
        self.labelTitleText.place(relwidth=1, relheigh=1)

        self.frameSubStream = tk.Frame(self.master)
        self.frameSubStream.place(relx=0.4, rely=0.20, relwidth=0.55,relheight=0.05, anchor='n')

        self.labelSubStream = tk.Label(self.frameSubStream, text="Subject Stream")
        self.labelSubStream.place(relwidth=0.3, relheight=1)
```

```
        self.entrySubStream = tk.Entry(self.frameSubStream,
                textvariable=StudentName)
        self.entrySubStream.place(relx=0.30, relwidth=0.55, relheight=1)

        # subjects
        self.frameSubject1 = tk.Frame(self.master)
        self.frameSubject1.place(relx=0.3, rely=0.30, relwidth=0.25, relheight=0.05, anchor='n')

        self.labelSubject1 = tk.Label(self.frameSubject1, text="Subject 01")
        self.labelSubject1.place(relwidth=0.4, relheight=1)

        self.entrySubject1 = tk.Entry(self.frameSubject1)
        self.entrySubject1.place(relx=0.45, relwidth=0.55, relheight=1)

        self.frameSubject2 = tk.Frame(self.master)
        self.frameSubject2.place(relx=0.3, rely=0.40, relwidth=0.25, relheight=0.05, anchor='n')

        self.labelSubject2 = tk.Label(self.frameSubject2, text="Subject 02")
        self.labelSubject2.place(relwidth=0.4, relheight=1)

        self.entrySubject2 = tk.Entry(self.frameSubject2)
        self.entrySubject2.place(relx=0.45, relwidth=0.55, relheight=1)

        self.frameSubject3 = tk.Frame(self.master)
        self.frameSubject3.place(relx=0.3, rely=0.50, relwidth=0.25, relheight=0.05, anchor='n')

        self.labelSubject3 = tk.Label(self.frameSubject3, text="Subject 03")
        self.labelSubject3.place(relwidth=0.4, relheight=1)

        self.entrySubject3 = tk.Entry(self.frameSubject3)
        self.entrySubject3.place(relx=0.45, relwidth=0.55, relheight=1)

        # Button Analyze

        self.frameButtonAnalyze = tk.Frame(self.master, bg='#9494b8', bd=2)
        self.frameButtonAnalyze.place(relx=0.5, rely=0.70, relwidth=0.25, relheight=0.05, anchor='n')

        self.buttonPrimaryWin = tk.Button(self.frameButtonAnalyze, text="Analyze Result ", font=20,
                                          command=lambda: self.analyseData())
        self.buttonPrimaryWin.place(relheight=1, relwidth=1)

        # Total and Average
        self.frameTotal = tk.Frame(self.master)
        self.frameTotal.place(relx=0.3, rely=0.80, relwidth=0.25, relheight=0.05, anchor='n')

        self.labelTotal = tk.Label(self.frameTotal, text="Total")
        self.labelTotal.place(relwidth=0.4, relheight=1)

        self.labelEntryTotal = tk.Label(self.frameTotal,
                textvariable=self.textTotal)
        self.labelEntryTotal.place(relx=0.45, relwidth=0.55, relheight=1)

        self.frameAverage = tk.Frame(self.master)
        self.frameAverage.place(relx=0.3, rely=0.90, relwidth=0.25, relheight=0.05, anchor='n')

        self.labelAverage = tk.Label(self.frameAverage, text="Average")
        self.labelAverage.place(relwidth=0.4, relheight=1)
```

```
self.labelEntryAverage = tk.Label(self.frameAverage, textvariable=self.textAverage)
self.labelEntryAverage.place(relx=0.45, relwidth=0.55, relheight=1)
```

- Advance Level User Interface



7.Backend

7.1. Create the database tables

we need to create a backend of the system which is the database. All the tables in the database for this exam result system include exam results for relevant student' level. There are three tables created for include student result. These tables are created initially when the exam result system is deployed. Information is not input into the database at the beginning.

Inside my server folder, created a new Python file named create_database.py and add the following code:

Primary Level created database

```
import sqlite3

con = sqlite3.connect("Student_Databases.db")
cur = con.cursor()


def PrimaryData():
   cur.execute("CREATE TABLE IF NOT EXISTS primary_student( StuID INTEGER PRIMARY KEY,
StuName TEXT,\
               Subject1 INTEGER, \
               Subject2 INTEGER, \
               Subject3 INTEGER, \
               Subject4 INTEGER, \
               Subject5 INTEGER, \
               Subject6 INTEGER, \
               Total INTEGER, Average INTEGER)")
   con.commit()

PrimaryData()
```

Next, def functions were created to Insert the student result into the table. Therefore, add following code to insert them.

Primary Level data insert to the database

```
def InsertOrdinaryData(StuID, StuName, Subject1, Subject2, Subject3, Subject4, Subject5, Subject6,
Subject7, Subject8, Subject9, StuTot, StuAver):

    cur.execute("INSERT INTO ordinary_student VALUES (?,?,?,?,?,?,?,?,?,?,?,?,?)",\
                (StuID, StuName, Subject1, Subject2, Subject3, Subject4, Subject5, Subject6, Subject7,
Subject8, Subject9, StuTot, StuAver))
    con.commit()
```

After Import database.py to Main file in python. The entry data is entered into the database using a function.

```
def analyseData(self):

    self.textTotal.set(self.CalculateTotal())
    self.textAverage.set(self.CalculateAverage())
    databse.InsertPrimaryData(self.StudentID, self.StudentName, int(self.entrySubject1.get()), \
                              int(self.entrySubject2.get()), int(self.entrySubject3.get()), \
                              int(self.entrySubject4.get()), int(self.entrySubject5.get()), \
                              int(self.entrySubject6.get()), self.CalculateTotal(), self.CalculateAverage())
```

It is executed by placing a command inside a button.

This is example of primary button command:

```
self.buttonPrimaryWinAna = tk.Button(self.frameButtonAnalyze, text="Analyze Result ", font=20,
                                     command=lambda: self.analyseData())
```

Created databases with included data

- Primary Level Database

	StuID	StuName	Subject1	Subject2	Subject3	Subject4	Subject5	Subject6	Total	Average
1	100	Nimal	45	56	98	87	78	78	442	73.66666666666667
2	10005	asdfg	78	45	67	67	56	56	369	61.5
3	10006	Nimali	78	89	67	78	56	98	466	77.66666666666667
4	10008	Samadi	78	67	80	70	78	56	429	71.5
5	10050	Vimansa	78	98	70	67	78	80	471	78.5
6	10678	Tharuka	77	80	75	70	71	83	456	76
7	12345	adfdfg	23	34	45	45	45	76	268	44.666666666666664
8	17904	Naween	84	78	96	74	56	78	466	77.66666666666667

- Ordinary Level Database

	StuID	StuName	Subject1	Subject2 1	Subject3	Subject4	..	..	..	..	..	Total	Average
1	1234	fgdfyh	39	90	68	89	78	90	78	89	78	699	77.6666666…
2	32400	Anushka	65	89	75	95	70	83	74	54	69	674	74.8888888…
3	43256	Anusha	96	85	45	65	35	74	45	65	45	555	61.6666666…
4	23456	Saduni	85	78	75	45	96	85	65	75	56	660	73.3333333…
5	40087	Rangi	85	78	95	75	65	95	65	65	78	701	77.8888888…
6	38796	Gayani	89	45	78	65	75	85	75	95	78	685	76.1111111…

- Ordinary Level Database

	StuID	StuName	SubjectStream	Subject1	Subject2	Subject3	Total	Av...
1	1234	asdf	art	97	78	89	264	88
2	50006	Lakshan	Commerce	78	95	78	251	83.66666…
3	54567	Ashan	Teach	67	87	60	214	71.33333…
4	60005	Danusha	Science Stream	78	67	89	234	78
5	70000	mulshan	commrce	45	78	69	192	64
6	70111	Sandeep	Maths	85	74	65	224	74.66666…
7	78567	mulshan	Art	98	75	84	257	85.66666…

7.2. Calculation

calculated student exam result is the main purpose of the system and output the calculated data. The exam result system is calculated total and average relevant the student exam results.

So total and average are calculated using def function.

Example :

Function for calculate total and average

```
def CalculateTotal(self):
    sum = int(self.entrySubject1.get()) + int(self.entrySubject2.get()) + int(self.entrySubject3.get()) + \
        int(self.entrySubject4.get()) + int(self.entrySubject5.get()) + int(self.entrySubject6.get())
    return sum

def CalculateAverage(self):
    average = self.CalculateTotal()/6
    return average
```

After that total and average converted as a string variable for show them through a label.

```
self.textTotal = StringVar()
self.textAverage = StringVar()
```

Finally, converted total and average call through a text variable.

Example:

Primary Level Total:

```
self.labelEntryTotal = tk.Label(self.frameTotal,
textvariable=self.textTotal)
self.labelEntryTotal.place(relx=0.45, relwidth=0.55, relheight=1)
```

Primary Level Average:

```
self.labelEntryAverage = tk.Label(self.frameAverage, textvariable=self.textAverage)
self.labelEntryAverage.place(relx=0.45, relwidth=0.55, relheight=1)
```

8. Conclusion

The python gives us a simple and reliable way to create the Exam result system. It provides powerful functionalities and concise syntax to help programmers deal with the database, and the inner logic. The experience of developing the system also helped me learning a lot of knowledge about creating system Python and SQ Lite.

In short, this system will bring great user experience to students. Once this system passes the testing phase, it can be used to serve students and Analyze exam result in short time. This system is very helpful for both the student and teachers. It is very helpful for the institutes to use for calculated complex exam result.

9. Reference

- W3schools.com. 2020. *Python Tutorial*. [online] Available at: https://www.w3schools.com/python/default.asp [Accessed 25 December 2020].
- tutorialspoint.com. "Python GUI Programming

  (Tkinter)." Www.Tutorialspoint.com, 2019,

  www.tutorialspoint.com/python/python_gui_programming.htm.
- "SQLite - Python - Tutorialspoint."

  www.Tutorialspoint.com

  , www.tutorialspoint.com/sqlite/sqlite_python.htm.

  Accessed 25 Dec. 2020.