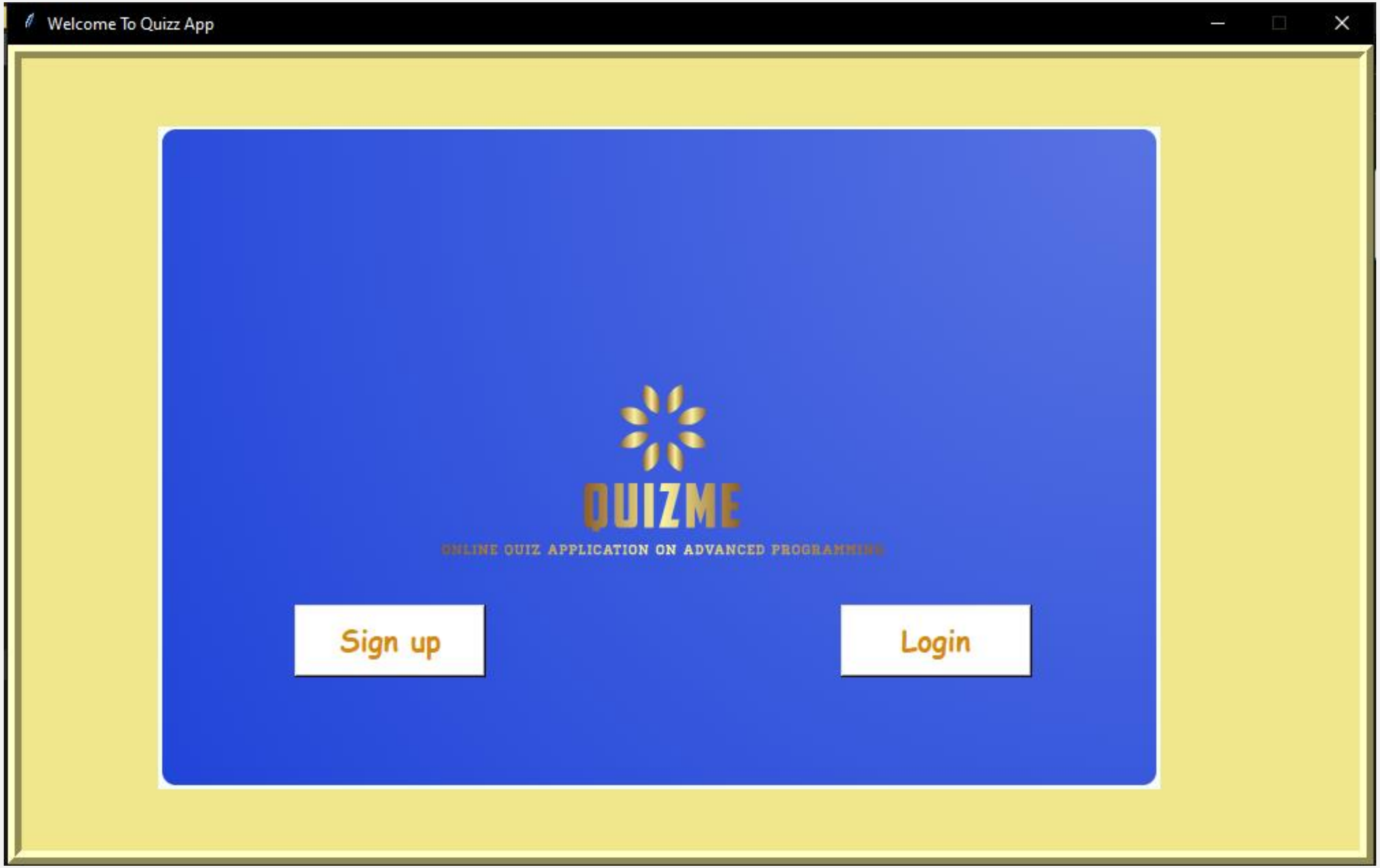
QUIZME ONLINE QUIZ APPLICATION



MODULES USED FOR THE QUIZ APPLICATION

According to our programming project our main task was to build a python-based GUI Quiz application. Basically, it was developed to ask multiple-choice questions from the user and collect user answers and finally display the results. We use different modules and methods and, we used python classes for the whole project development.

- **tkinter module**

  Used to create the graphical user interface of the application

- **random module**

  Used to shuffle the questions from question collection

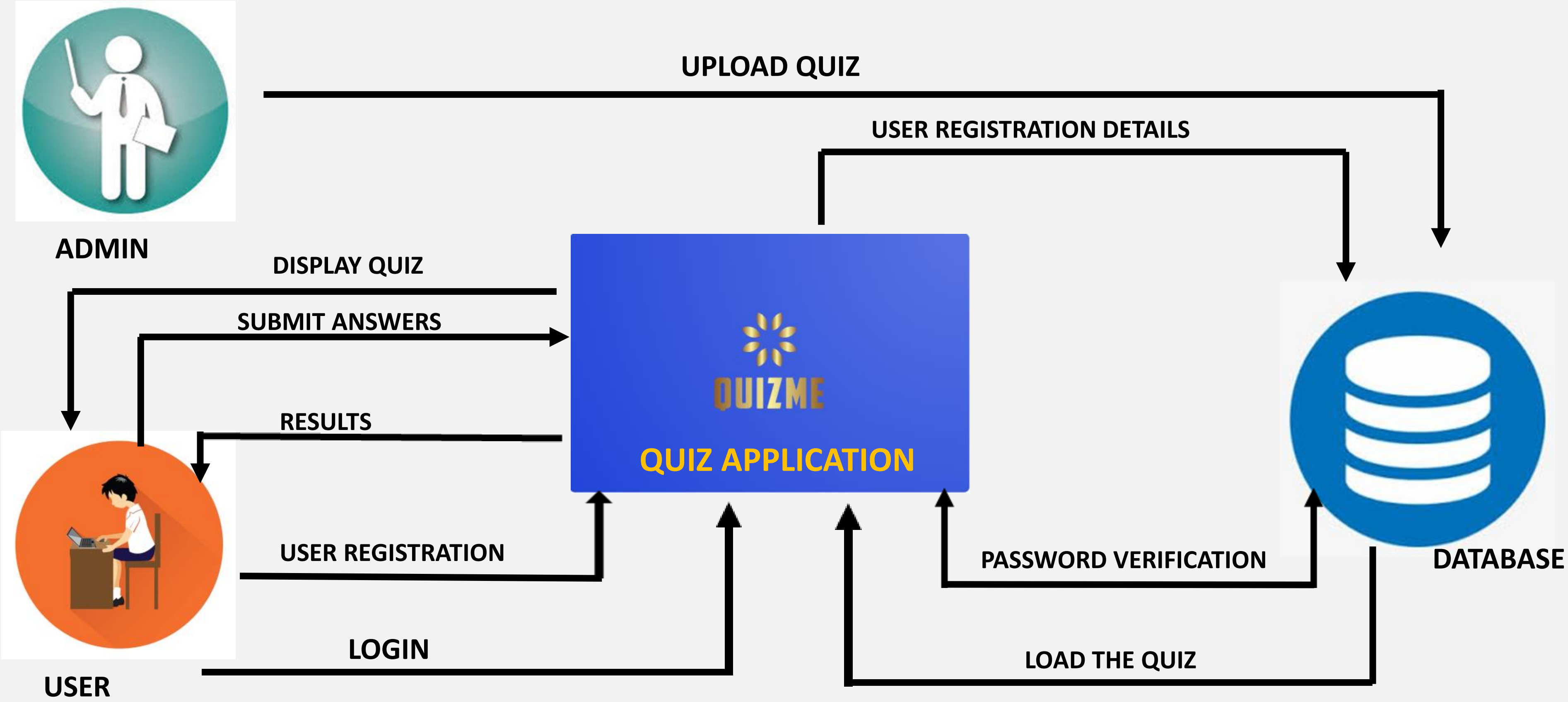
- **mysql.connector**

  This is a standardized database driver for python platforms and development. It enables python programs to access MySQL databases.

- **csv module**

  Used to retrieve data from spreadsheet files (Microsoft excel)

SYSTEM ARCHITECTURE



INTERFACE DESIGNING

CONCEPTS AND TOOLS
PYTHON CODE

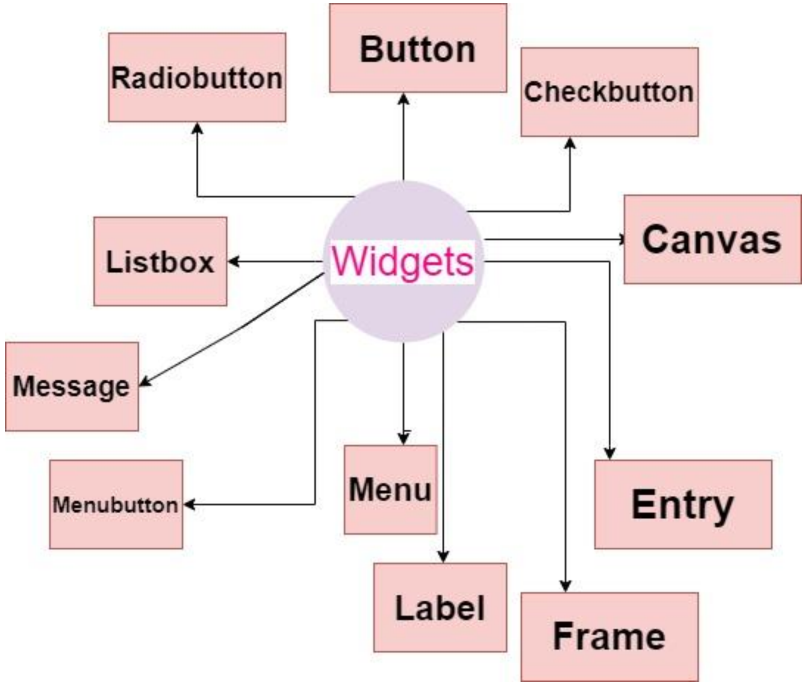
Tkinter module

Tkinter Programming

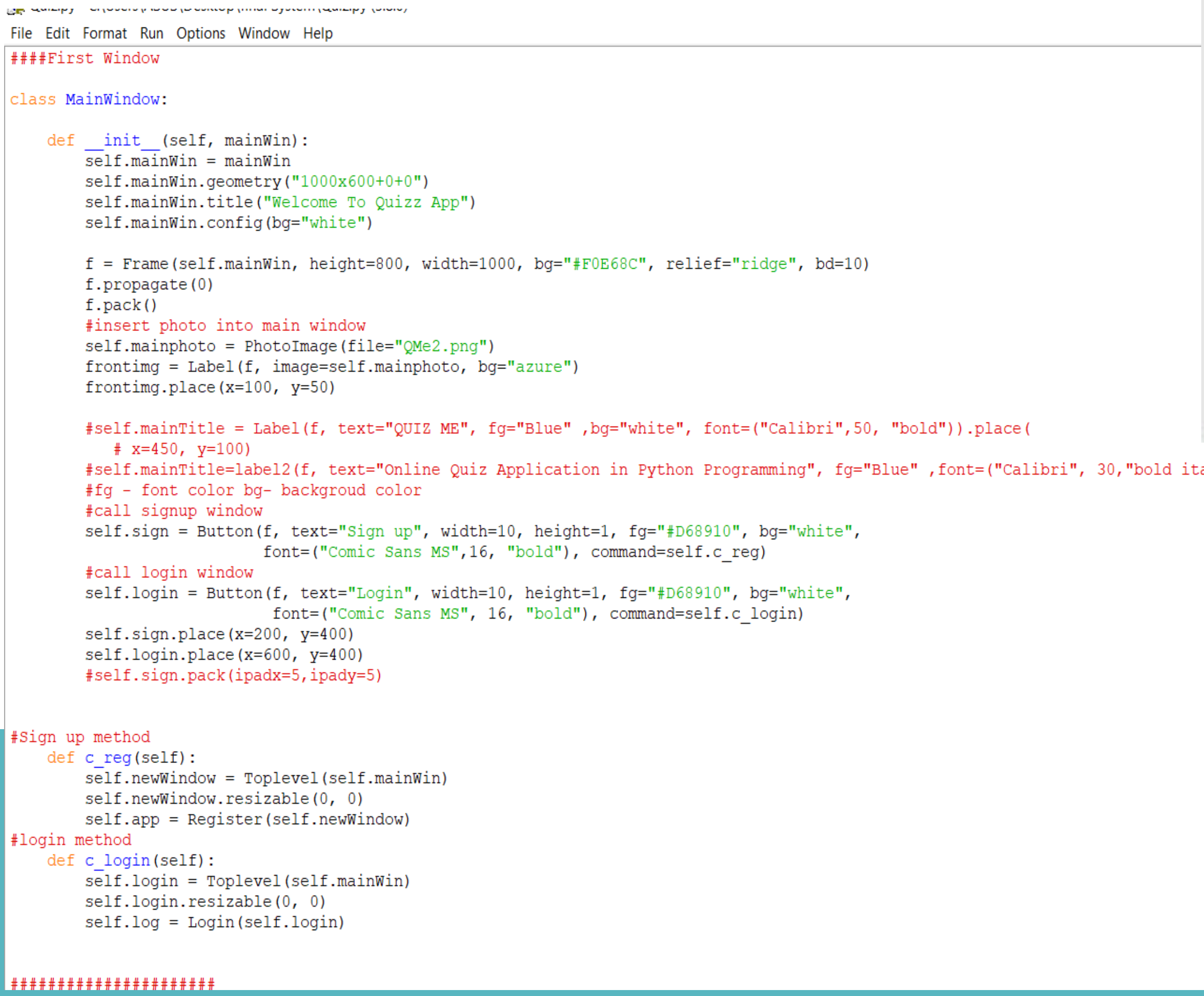
Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

Tkinter Widgets

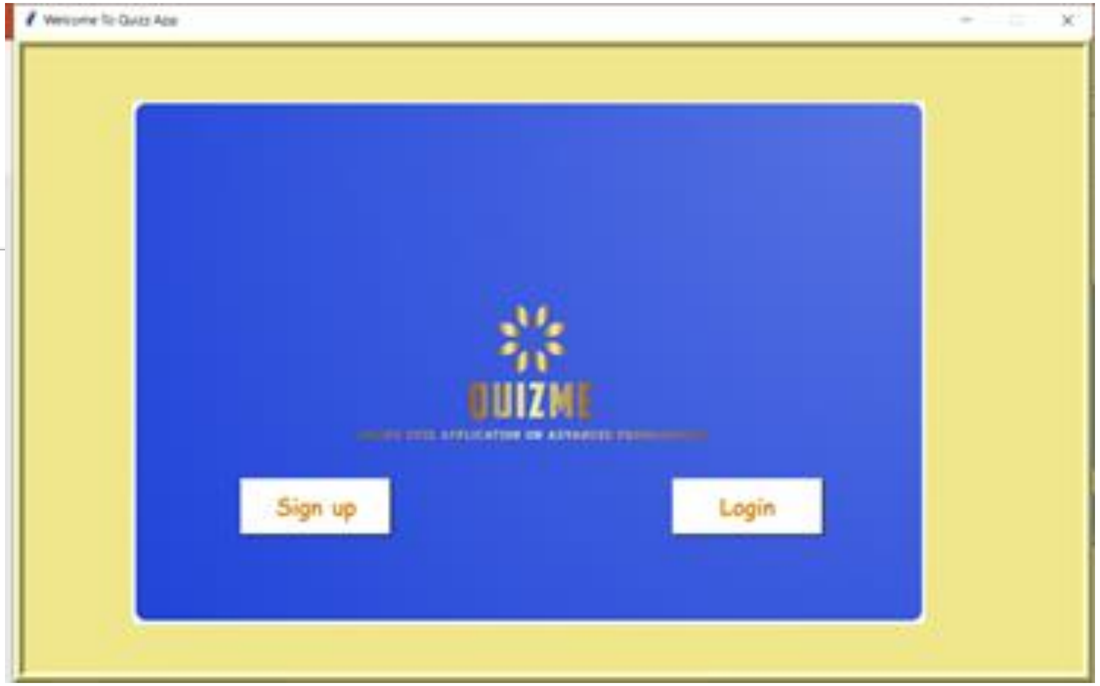
Tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets.

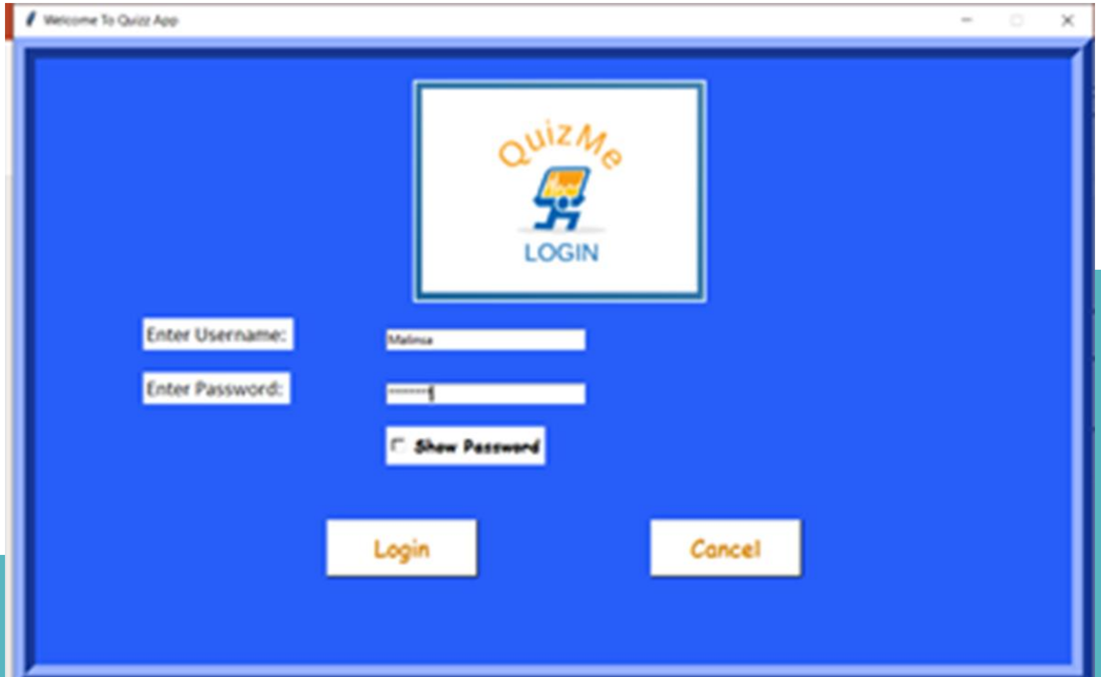


MAIN WINDOW

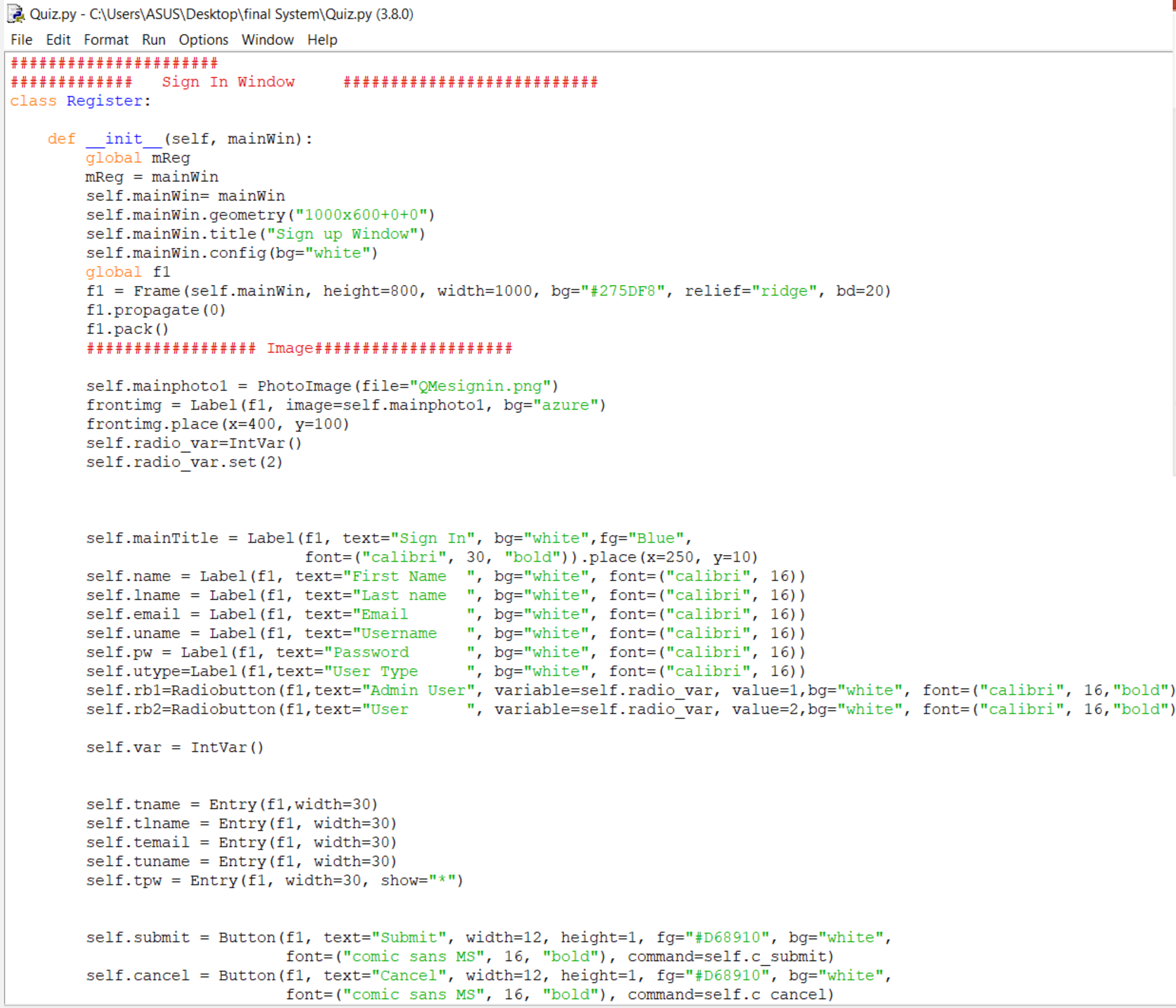


```
####First Window

class MainWindow:

    def __init__(self, mainWin):
        self.mainWin = mainWin
        self.mainWin.geometry("1000x600+0+0")
        self.mainWin.title("Welcome To Quizz App")
        self.mainWin.config(bg="white")

        f = Frame(self.mainWin, height=800, width=1000, bg="#F0E68C", relief="ridge", bd=10)
        f.propagate(0)
        f.pack()
        #insert photo into main window
        self.mainphoto = PhotoImage(file="QMe2.png")
        frontimg = Label(f, image=self.mainphoto, bg="azure")
        frontimg.place(x=100, y=50)

        #self.mainTitle = Label(f, text="QUIZ ME", fg="Blue" ,bg="white", font=("Calibri",50, "bold")).place(
           # x=450, y=100)
        #self.mainTitle=label2(f, text="Online Quiz Application in Python Programming", fg="Blue" ,font=("Calibri", 30,"bold italic")).place(x=50, y=100)
        #fg - font color bg- backgroud color
        #call signup window
        self.sign = Button(f, text="Sign up", width=10, height=1, fg="#D68910", bg="white",
                           font=("Comic Sans MS",16, "bold"), command=self.c_reg)
        #call login window
        self.login = Button(f, text="Login", width=10, height=1, fg="#D68910", bg="white",
                            font=("Comic Sans MS", 16, "bold"), command=self.c_login)
        self.sign.place(x=200, y=400)
        self.login.place(x=600, y=400)
        #self.sign.pack(ipadx=5,ipady=5)


#Sign up method
    def c_reg(self):
        self.newWindow = Toplevel(self.mainWin)
        self.newWindow.resizable(0, 0)
        self.app = Register(self.newWindow)
#login method
    def c_login(self):
        self.login = Toplevel(self.mainWin)
        self.login.resizable(0, 0)
        self.log = Login(self.login)


######################
```



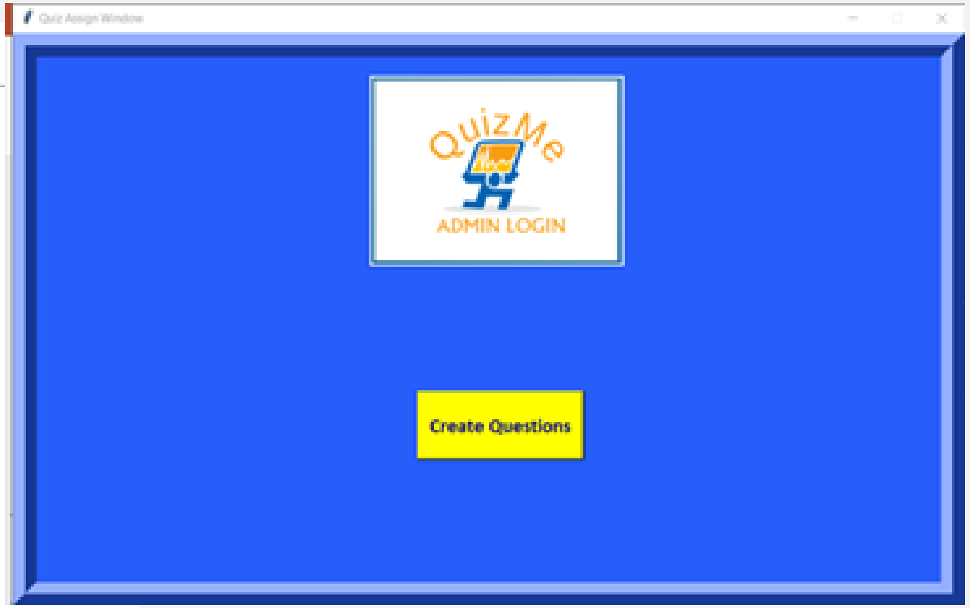


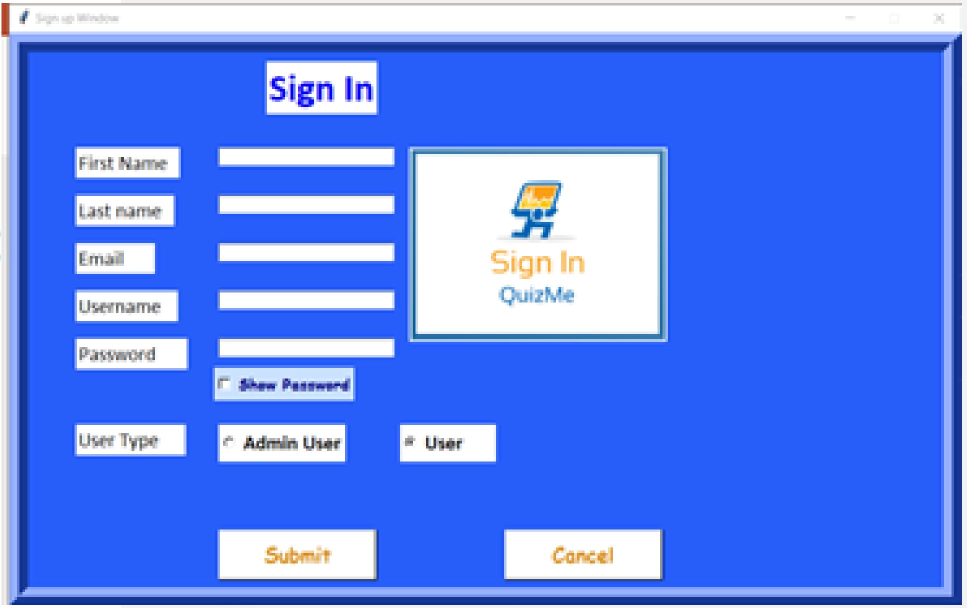
USER LOGIN AND ADMINISTRATOR LOGIN



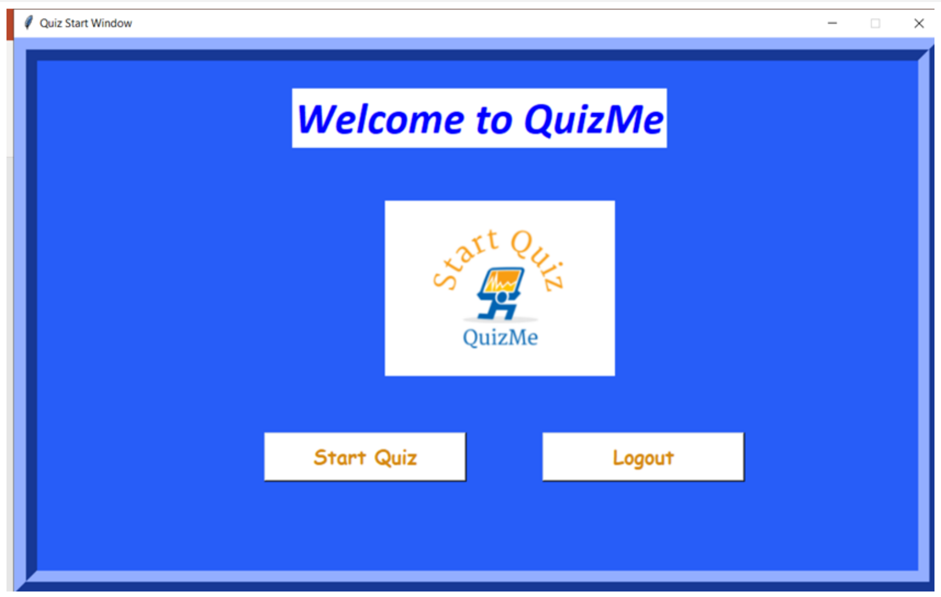
Quiz.py - C:\Users\ASUS\Desktop\final System\Quiz.py (3.8.0)
File Edit Format Run Options Window Help

```
######################
############   Sign In Window     ###########################
class Register:

    def __init__(self, mainWin):
        global mReg
        mReg = mainWin
        self.mainWin= mainWin
        self.mainWin.geometry("1000x600+0+0")
        self.mainWin.title("Sign up Window")
        self.mainWin.config(bg="white")
        global f1
        f1 = Frame(self.mainWin, height=800, width=1000, bg="#275DF8", relief="ridge", bd=20)
        f1.propagate(0)
        f1.pack()
        ################# Image####################

        self.mainphoto1 = PhotoImage(file="QMesignin.png")
        frontimg = Label(f1, image=self.mainphoto1, bg="azure")
        frontimg.place(x=400, y=100)
        self.radio_var=IntVar()
        self.radio_var.set(2)



        self.mainTitle = Label(f1, text="Sign In", bg="white",fg="Blue",
                               font=("calibri", 30, "bold")).place(x=250, y=10)
        self.name = Label(f1, text="First Name  ", bg="white", font=("calibri", 16))
        self.lname = Label(f1, text="Last name  ", bg="white", font=("calibri", 16))
        self.email = Label(f1, text="Email      ", bg="white", font=("calibri", 16))
        self.uname = Label(f1, text="Username   ", bg="white", font=("calibri", 16))
        self.pw = Label(f1, text="Password     ", bg="white", font=("calibri", 16))
        self.utype=Label(f1,text="User Type    ", bg="white", font=("calibri", 16))
        self.rb1=Radiobutton(f1,text="Admin User", variable=self.radio_var, value=1,bg="white", font=("calibri", 16,"bold"))
        self.rb2=Radiobutton(f1,text="User       ", variable=self.radio_var, value=2,bg="white", font=("calibri", 16,"bold"))

        self.var = IntVar()


        self.tname = Entry(f1,width=30)
        self.tlname = Entry(f1, width=30)
        self.temail = Entry(f1, width=30)
        self.tuname = Entry(f1, width=30)
        self.tpw = Entry(f1, width=30, show="*")


        self.submit = Button(f1, text="Submit", width=12, height=1, fg="#D68910", bg="white",
                             font=("comic sans MS", 16, "bold"), command=self.c_submit)
        self.cancel = Button(f1, text="Cancel", width=12, height=1, fg="#D68910", bg="white",
                             font=("comic sans MS", 16, "bold"), command=self.c_cancel)
```

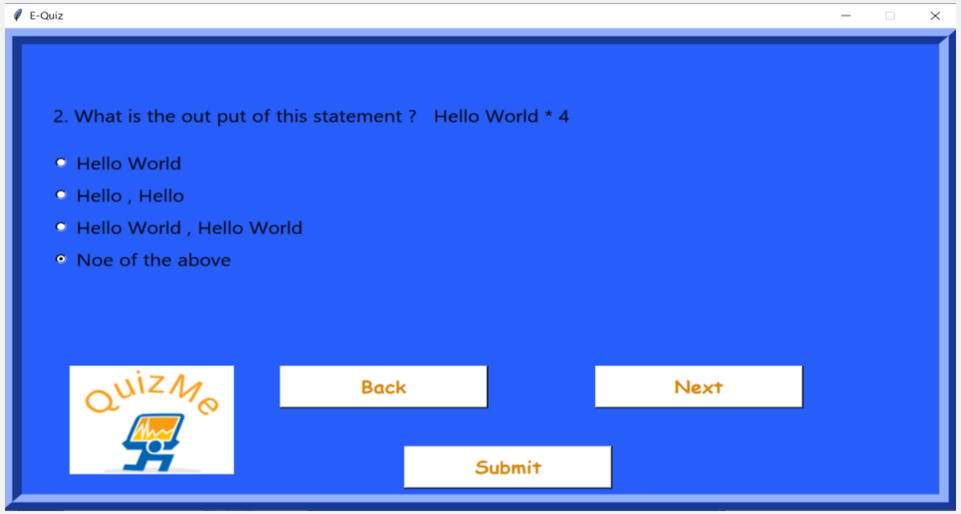




START QUIZ WINDOW



QUIZ WINDOW



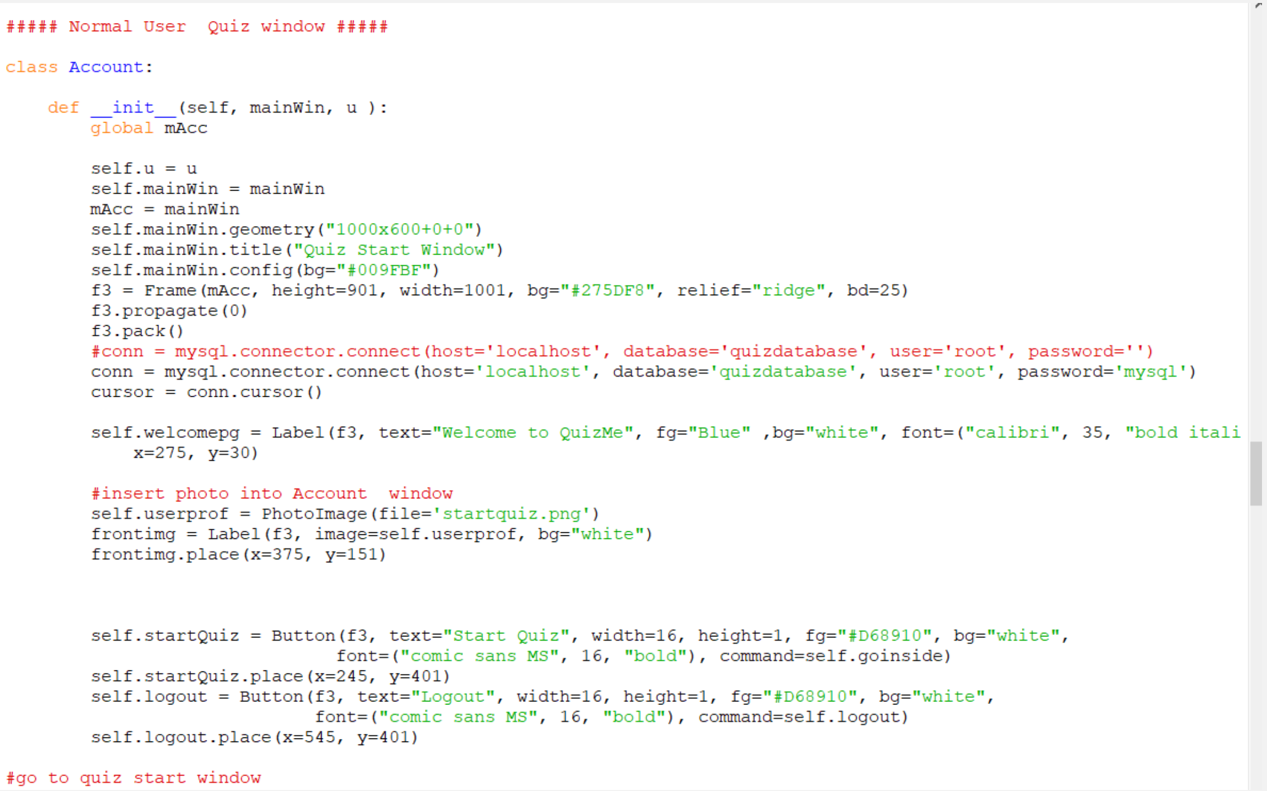
2. What is the out put of this statement ?   Hello World * 4

○ Hello World

○ Hello , Hello

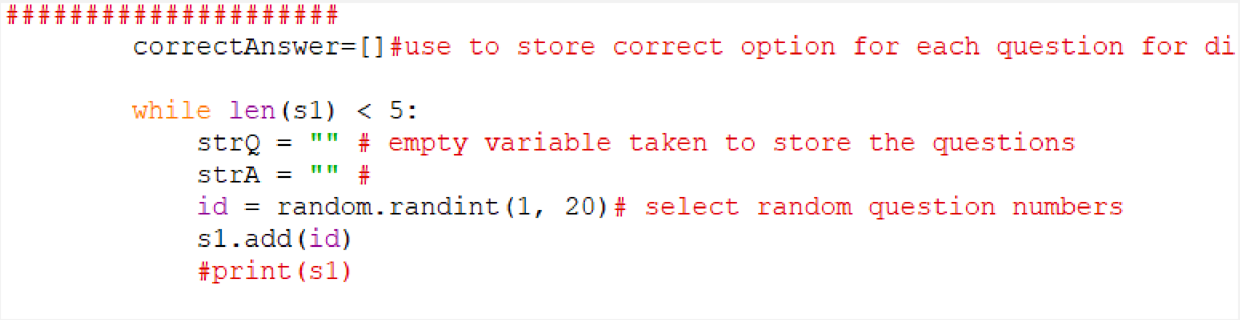
○ Hello World , Hello World

○ Noe of the above



```
##### Normal User  Quiz window #####

class Account:

    def __init__(self, mainWin, u ):
        global mAcc

        self.u = u
        self.mainWin = mainWin
        mAcc = mainWin
        self.mainWin.geometry("1000x600+0+0")
        self.mainWin.title("Quiz Start Window")
        self.mainWin.config(bg="#009FBF")
        f3 = Frame(mAcc, height=901, width=1001, bg="#275DF8", relief="ridge", bd=25)
        f3.propagate(0)
        f3.pack()
        #conn = mysql.connector.connect(host='localhost', database='quizdatabase', user='root', password='')
        conn = mysql.connector.connect(host='localhost', database='quizdatabase', user='root', password='mysql')
        cursor = conn.cursor()

        self.welcomepg = Label(f3, text="Welcome to QuizMe", fg="Blue" ,bg="white", font=("calibri", 35, "bold itali
            x=275, y=30)

        #insert photo into Account  window
        self.userprof = PhotoImage(file='startquiz.png')
        frontimg = Label(f3, image=self.userprof, bg="white")
        frontimg.place(x=375, y=151)



        self.startQuiz = Button(f3, text="Start Quiz", width=16, height=1, fg="#D68910", bg="white",
                                font=("comic sans MS", 16, "bold"), command=self.goinside)
        self.startQuiz.place(x=245, y=401)
        self.logout = Button(f3, text="Logout", width=16, height=1, fg="#D68910", bg="white",
                              font=("comic sans MS", 16, "bold"), command=self.logout)
        self.logout.place(x=545, y=401)

#go to quiz start window
```

Assign randomly selected five questions for each user to answer



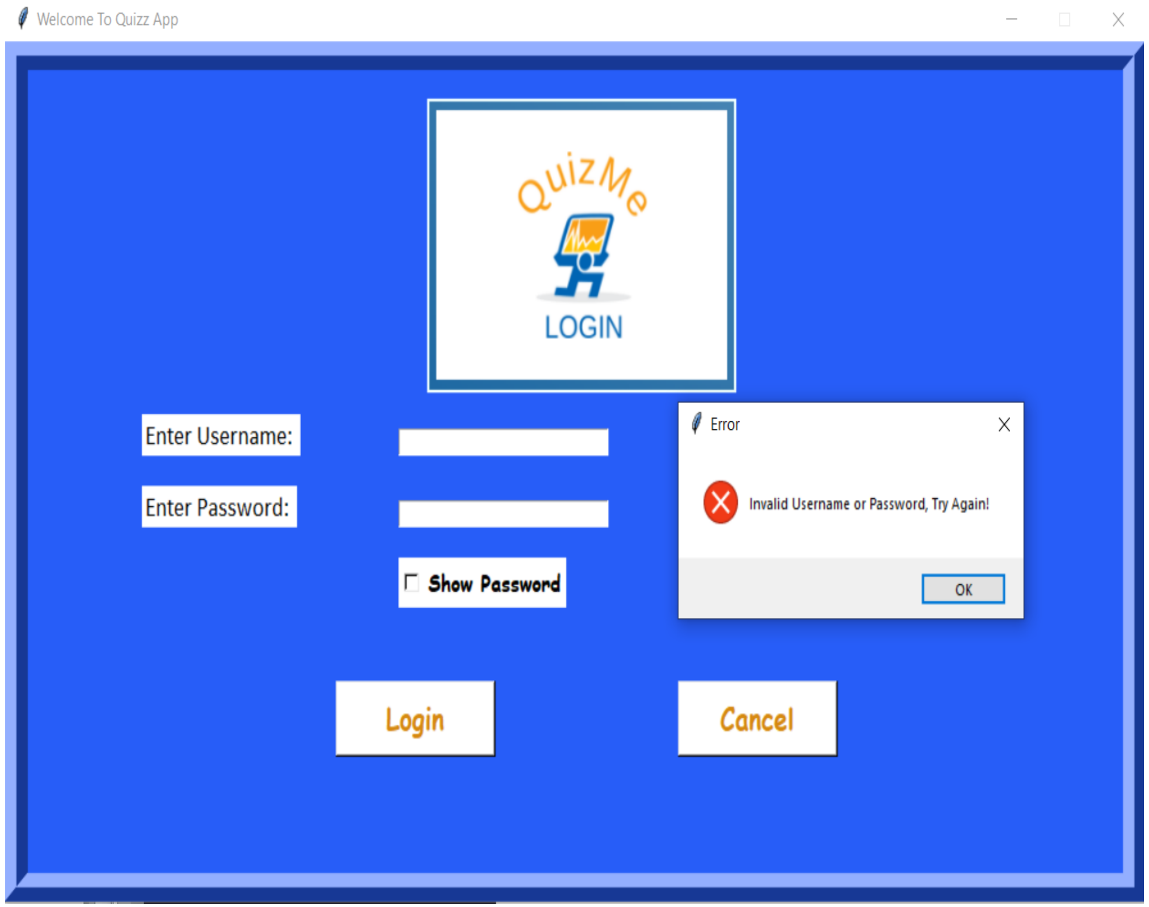
```
####################
        correctAnswer=[]#use to store correct option for each question for di

        while len(s1) < 5:
            strQ = "" # empty variable taken to store the questions
            strA = "" #
            id = random.randint(1, 20)# select random question numbers
            s1.add(id)
            #print(s1)
```

QUIZ VALIDATION

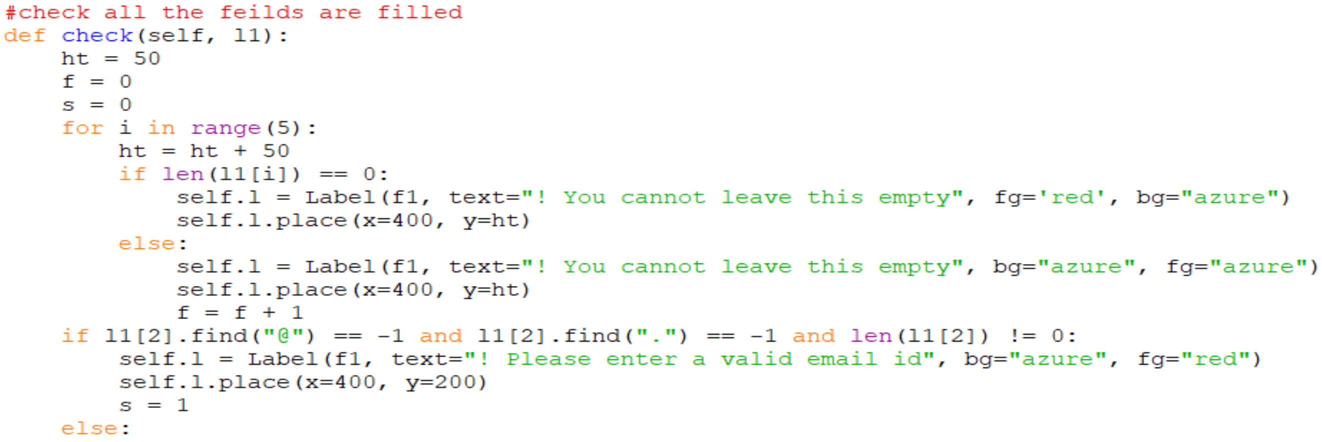
CONCEPTS AND TOOLS
PYTHON CODE

Display error messages if ,

- Entered wrong password or user name
- Required fields are empty
- Entered less number of characters to password
- Entered invalid email address format



Invalid Username or Password, Try Again!



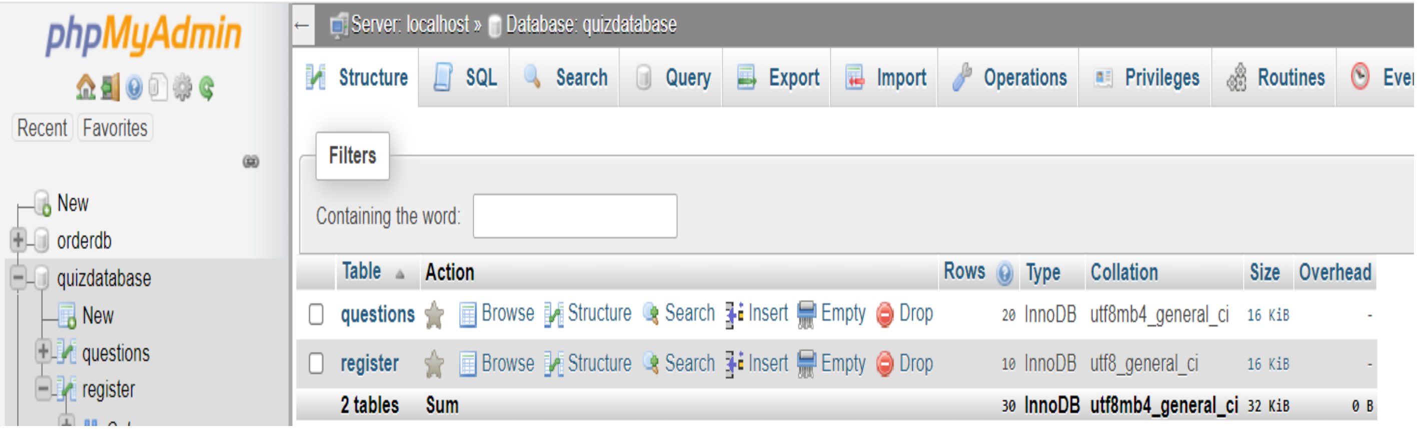
```
#check all the feilds are filled
def check(self, l1):
    ht = 50
    f = 0
    s = 0
    for i in range(5):
        ht = ht + 50
        if len(l1[i]) == 0:
            self.l = Label(f1, text="! You cannot leave this empty", fg='red', bg="azure")
            self.l.place(x=400, y=ht)
        else:
            self.l = Label(f1, text="! You cannot leave this empty", bg="azure", fg="azure")
            self.l.place(x=400, y=ht)
            f = f + 1
    if l1[2].find("@") == -1 and l1[2].find(".") == -1 and len(l1[2]) != 0:
        self.l = Label(f1, text="! Please enter a valid email id", bg="azure", fg="red")
        self.l.place(x=400, y=200)
        s = 1
    else:
```

DATABASE DESIGNING

CONCEPTS AND TOOLS
PYTHON CODE

There are two tables

- Question Table
- Register Table

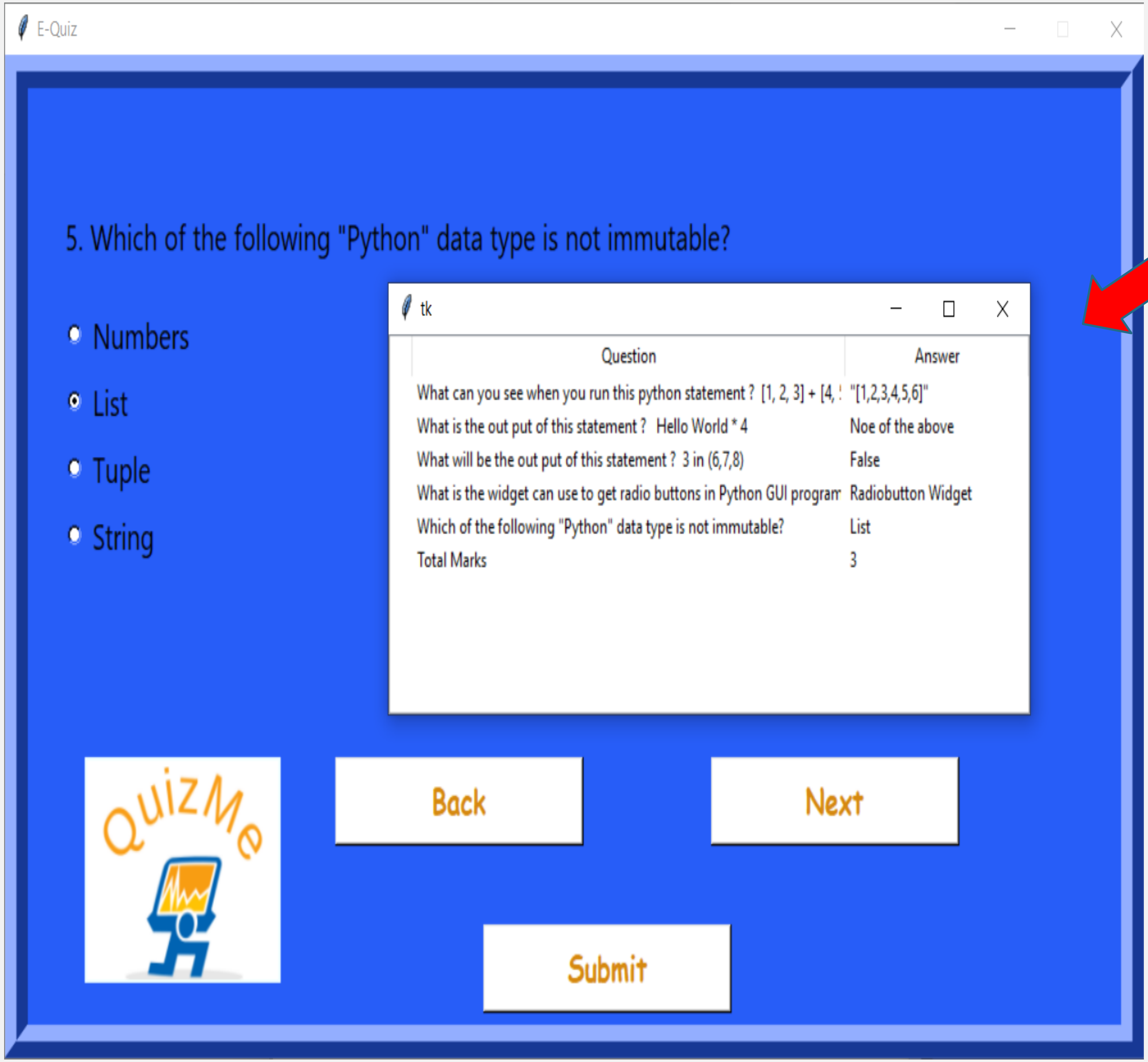


Containing the word:

	Table	Action	Rows	Type	Collation	Size	Overhead
□	questions	★ Browse Structure Search Insert Empty Drop	20	InnoDB	utf8mb4_general_ci	16 KiB	-
□	register	★ Browse Structure Search Insert Empty Drop	10	InnoDB	utf8_general_ci	16 KiB	-
	2 tables	Sum	30	InnoDB	utf8mb4_general_ci	32 KiB	0 B

Database connection

```
####  Connect to the database
        db_connection = mysql.connector.connect(
        host="localhost",
        user="root",
        passwd="mysql",
        #passwd="",
        database="quizdatabase"

        )
```



5. Which of the following "Python" data type is not immutable?

○ Numbers

● List

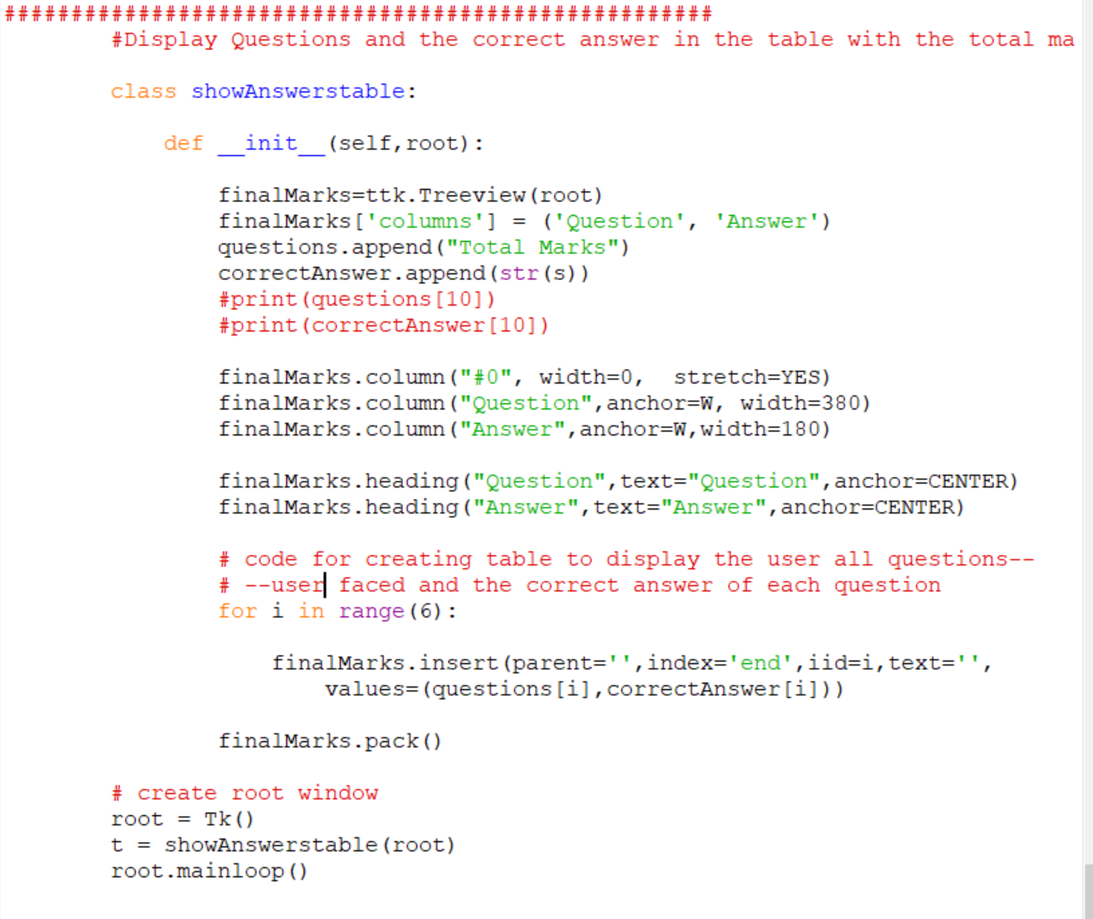
○ Tuple

○ String

Question	Answer
What can you see when you run this python statement ? [1, 2, 3] + [4, !	"[1,2,3,4,5,6]"
What is the out put of this statement ? Hello World * 4	Noe of the above
What will be the out put of this statement ? 3 in (6,7,8)	False
What is the widget can use to get radio buttons in Python GUI program	Radiobutton Widget
Which of the following "Python" data type is not immutable?	List
Total Marks	3

Display Attempted questions and correct answer of each question with total marks earned by the user at the end of the quiz in a tabular form

For that tabular display we use “ttk” library



```
######################################################
        #Display Questions and the correct answer in the table with the total ma

        class showAnswerstable:

            def __init__(self,root):

                finalMarks=ttk.Treeview(root)
                finalMarks['columns'] = ('Question', 'Answer')
                questions.append("Total Marks")
                correctAnswer.append(str(s))
                #print(questions[10])
                #print(correctAnswer[10])

                finalMarks.column("#0", width=0,  stretch=YES)
                finalMarks.column("Question",anchor=W, width=380)
                finalMarks.column("Answer",anchor=W,width=180)

                finalMarks.heading("Question",text="Question",anchor=CENTER)
                finalMarks.heading("Answer",text="Answer",anchor=CENTER)

                # code for creating table to display the user all questions--
                # --user faced and the correct answer of each question
                for i in range(6):

                    finalMarks.insert(parent='',index='end',iid=i,text='',
                        values=(questions[i],correctAnswer[i]))

                finalMarks.pack()

        # create root window
        root = Tk()
        t = showAnswerstable(root)
        root.mainloop()
```