

Table of Contents

S. No.	Description	Page No.
1	Acknowledgement	2
2	Introduction	3
3	Objective	3
4	Proposed System	4
5	Hardware and Software Requirements	5
6	Technology Used	6
7	Database Used	11
8	Source Code	13
9	Output	59
10	Bibliography	61

ACKNOWLEDGMENT

I would like to express my special thanks of gratitude to my teacher **Mr. Gopal Singh** as well as our **Principal Mr. Biswajit Dutta** who gave me the golden opportunity to do this wonderful project on the topic ' Online Quiz System' which also helped me in doing a lot of research and I came to know about so many new things.

Secondly, I would also like to thank my parents and friends who helped me a lot in finishing this project within the time limit. I am making this project not only for marks but to also increase my knowledge.

I express deep sense of gratitude to almighty God for giving me strength for the successful completion of the project.

I gratefully acknowledge the contribution of the individuals who contributed in bringing this project up to this level, who continues to look after me despite my flaws.

The guidance and support received from all the members who contributed and who are contributing to this project, was vital for the success of the project. I am grateful for their constant support and help.

Student Name
(Omkaar)

Online Quiz System

INTRODUCTION

The Online Quiz system is basically a GUI & CSV-based project done with help of python language. this project is very useful for the School and Colleges to maintain record of student's regular progress in each subject. This project is multifield project, so that it can be modified for various purposes.

OBJECTIVES OF THE PROJECT

The objective of this project is to let the students apply the programming knowledge into a real- world situation/problem and exposed the students how programming skills helps in developing a good software.

- Write programs utilizing modern software tools.
- Apply object-oriented programming principles effectively when developing small to medium sized projects.
- Write effective procedural code to solve small to medium sized problems.
- Students will demonstrate a breadth of knowledge in computer science, as exemplified in the areas of systems, theory and software development.

Students will demonstrate ability to conduct research or applied Informatics Practices project, requiring writing and presentation skills which exemplify scholarly style in the field of computer science.

PROPOSED SYSTEM

Today one cannot afford to rely on the fallible human beings of be really wants to stand against today's merciless competition where not to wise saying "to err is human" no longer valid, it's outdated to rationalize your mistake. So, to keep pace with time, to bring about the best result without malfunctioning and greater efficiency so to replace the unending heaps of files with a much-sophisticated hard disk of the computer.

One has to use the data management software. Software has been an ascent in atomization various organisations. Many software products working are now in markets, which have helped in making the organizations work easier and efficiently. Data management initially had to maintain a lot of ledgers and a lot of paper work has to be done but now software product on this organization has made their work faster and easier. Now only this software has to be loaded on the computer and work can be done.

This prevents a lot of time and money. The work becomes fully automated and any information regarding the organization can be obtained by clicking the button. Moreover, now it's an age of computers of and automating such an organization gives the better look.

HARDWARE AND SOFTWARE REQUIREMENTS

- I. OPERATING SYSTEM : WINDOWS 10 AND ABOVE
- II. PROCESSOR : PENTIUM DUAL CORE OR ABOVE
ATHALON (3800+- 4200+ DUAL CORE)
- III. MOTHERBOARD : 1.845 OR 915,995 FOR PENTIUM OR MSI
K9MM-V VIA K8M800+8237R PLUS
CHIPSET FOR AMD ATHALON
- IV. PROGRAMMING LANGUAGE : PYTHON 3.3.7 OR ABOVE
- V. DATABASE : MySQL 5.5.27 OR ABOVE

TECHNOLOGY USED

Python

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.



Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming.

Guido van Rossum began working on Python in the late 1980s as a successor to the ABC



programming language and first released it in 1991 as Python 0.9.0.

Python 2.0 was released in 2000 and introduced new features such as list comprehensions, cycle-detecting garbage collection, reference counting, and Unicode

support. Python 3.0, released in 2008, was a major revision

that is not completely backward-compatible with earlier versions. Python 2 was discontinued with version 2.7.18 in 2020.

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 is used for:

- web development (server-side),
- software development,
- mathematics,
- system scripting.

What can Python do?

- ✓ Python can be used on a server to create web applications.
- ✓ Python can be used alongside software to create workflows.
- ✓ Python can connect to database systems. It can also read and modify files.
- ✓ Python can be used to handle big data and perform complex mathematics.
- ✓ Python can be used for rapid prototyping, or for production-ready software development.

Why Python?

- ✓ Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- ✓ Python has a simple syntax similar to the English language.
- ✓ Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- ✓ Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.

- ✓ Python can be treated in a procedural way, an object-oriented way or a functional way.

It is possible to write Python in an Integrated Development Environment, such as Thonny, Pycharm, Netbeans or Eclipse which are particularly useful when managing larger collections of Python files.

Python Syntax compared to other programming languages

- ❖ Python was designed for readability, and has some similarities to the English language with influence from mathematics.
- ❖ Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
- ❖ Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

MySQL

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation.



The MySQL website (<http://www.mysql.com/>) provides the latest information about MySQL software.

MySQL is a database management system.

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

MySQL databases are relational.

A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and “pointers” between different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data.

The SQL part of “MySQL” stands for “Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax.

SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, “SQL-92” refers to the standard released in 1992, “SQL:1999” refers to the standard released in 1999, and “SQL:2003” refers to the current version of the standard. We use the phrase “the SQL standard” to mean the current version of the SQL Standard at any time.

MySQL software is Open Source.

Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License).

DATABASE USED

Comma Separated Values (C.S.V) File

CSV File Name: users.csv

example of users.csv

username	password
admin	admin@1234
omkaar	1234
Krishna Bansal	123456

CSV File Name: user_progress.csv

example of user_progress.csv

Username	Score	Subject
omkaar	7	Science
admin	9	History
Krishna Bansal	6	Current Affairs

CSV File Name: Krishna.csv

example of Krishna.csv

Subject	Question ID	Question	OptionA	OptionB	OptionC	OptionD	Correct Option
General knowledge	GEN01	What is the capital of France?	Paris	London	Berlin	Madrid	Option A
Science	SCI01	What is H2O commonly known as?	Salt	Water	Hydrogen	Oxygen	Option B
History	HIS01	Who was the first president of the United States?	George Washington	Thomas Jefferson	Abraham Lincoln	John Adams	Option A

SOURCE CODE

```
from tkinter import *  
from tkinter import messagebox  
import pandas as pd  
import matplotlib.pyplot as plt  
from PIL import Image,ImageTk
```

username=NONE

#helping Functions

```
def get_length(df,sub,user):  
  
    id_df=df[(df['Subject']==sub) & (df['Username']==user)]  
  
    length=len(id_df)  
  
    ls=[j for j in range(1,length+1)]  
  
    return ls
```

```
def get_subject(df,sub,user):
```

```
Scoredf=df[(df['Subject']==sub) & (df['Username']==user)]
```

```
Scoredf=Scoredf['Score']
```

```
return Scoredf
```

```
def get_max(ls, df, user):
```

```
    results = []
```

```
    for index in range(len(ls)):
```

```
        Sub_max = df[(df['Subject'] == ls[index]) & (df['Username'] == user)]
```

```
        if not Sub_max.empty:
```

```
            results.append(max(Sub_max['Score']))
```

```
        else:
```

```
            results.append(0) # Default score if no matching records
```

```
    return results
```

```
#Chart creating functions
```

```
def create_piechart(score):
```

```
    plt.pie([score,10-score],labels=["Right","Wrong"])
```

```
    plt.legend()
```

```
    plt.savefig('Images/piechart')
```

```
plt.close()
```

```
def create_barchart(quiz_subjects,user_progress_data,username):
```

```
sub=["G.K","Science","Maths","History","Geo","Lit.","Tech","Sports",  
"Ent.","Mytho","C.A","Pop Cul."]
```

```
plt.barh(sub,get_max(quiz_subjects,user_progress_data,username))
```

```
plt.yticks(fontsize=8)
```

```
plt.ylabel('Subjects')
```

```
plt.xlabel('Maximum Score')
```

```
plt.savefig('Images/Barchart')
```

```
plt.close()
```

```
def create_linechart(subject,user_progress_data,username):
```

```
plt.plot(get_length(user_progress_data, subject, username),  
get_subject(user_progress_data, subject, username), label=subject)
```

```
plt.legend()
```

```
plt.savefig('Images/LineChart')
```

```
plt.close()
```

```
#button function & backend
```

def add_user(): #Signing Up and adding user in respective csv file

global username

username = signup_username_ent.get().strip()

password = signup_password_ent.get().strip()

if not username or not password:

messagebox.showwarning('Fill the fields', 'Please enter both username and password.')

return

if username in user_pass_data['username'].values:

messagebox.showinfo('Username already exists', f'Username '{username}' is already taken. Please choose a different one.')

else:

user_pass_data.loc[len(user_pass_data)] = [username, password]

user_pass_data.to_csv('Data/users.csv', index=False)

signup_username_ent.delete(0, END)

signup_password_ent.delete(0, END)

messagebox.showinfo('Success', 'User registered successfully!')

show_frame(Login)


```

def login_user(): #Login in user

    global username

    username = login_username_ent.get().strip()

    password = login_password_ent.get().strip()


    if not username or not password:

        messagebox.showwarning('Fill the fields', 'Please enter both username
and password.')

        return


    user_row = user_pass_data[user_pass_data['username'] == username]

    if not user_row.empty and user_row.iloc[0]['password'] == password:

        messagebox.showinfo('Success', 'Login successful!')

        show_frame(Declar)

    else:

        messagebox.showwarning('login failed', 'please enter correct username
and password')


def increase_qno(): #increasing qustion number

    global qno

```

qno+=1

def Show_Question(id): #Showing question for the first instance

global qno

global globalid

global section_question

globalid = id # Update the global ID for the subject

Filter questions for the selected subject

**section_question = questions_data[questions_data['Subject'] ==
id].sample(10).reset_index(drop=True)**

Check if `qno` is within valid range

question_text = f'{qno}. {section_question.loc[qno - 1, 'Question']}'

Question_label.configure(text=question_text)

Update options

opt1.configure(text=section_question.loc[qno - 1, "OptionA"])

opt2.configure(text=section_question.loc[qno - 1, "OptionB"])

```
opt3.configure(text=section_question.loc[qno - 1, "OptionC"])
```

```
opt4.configure(text=section_question.loc[qno - 1, "OptionD"])
```

```
def show_Questions_forcont(): #Showing question after first instance
```

```
    global qno
```

```
    global section_question
```

```
    if qno<=len(section_question):
```

```
        question_text = f"{qno}. {section_question.loc[qno - 1, 'Question']}"
```

```
        Question_label.configure(text=question_text)
```

```
        # Update options
```

```
        opt1.configure(text=section_question.loc[qno - 1, "OptionA"])
```

```
        opt2.configure(text=section_question.loc[qno - 1, "OptionB"])
```

```
        opt3.configure(text=section_question.loc[qno - 1, "OptionC"])
```

```
        opt4.configure(text=section_question.loc[qno - 1, "OptionD"])
```

```
        opt1.configure(bg='#2eff70')
```

```
        opt2.configure(bg='#2eff70')
```

```
        opt3.configure(bg='#2eff70')
```

```
        opt4.configure(bg='#2eff70')
```

else:

Progress_recorded()

show_stats()

def check(opt):

global res

global section_question

global score

section_ans=section_question

if opt==section_ans.loc[qno-1,'Correct_Option']:

res=1

else:

res=0

def color_button(p):

if p=='Option A':

opt1.configure(bg='red')

opt2.configure(bg='#2eff70')

opt3.configure(bg='#2eff70')

opt4.configure(bg='#2eff70')

elif p=='Option B':

```
opt1.configure(bg='#2eff70')
```

```
opt2.configure(bg='red')
```

```
opt3.configure(bg='#2eff70')
```

```
opt4.configure(bg='#2eff70')
```

```
elif p=='Option C':
```

```
opt1.configure(bg='#2eff70')
```

```
opt2.configure(bg='#2eff70')
```

```
opt3.configure(bg='red')
```

```
opt4.configure(bg='#2eff70')
```

```
elif p=='Option D':
```

```
opt1.configure(bg='#2eff70')
```

```
opt2.configure(bg='#2eff70')
```

```
opt3.configure(bg='#2eff70')
```

```
opt4.configure(bg='red')
```

```
def score_add():
```

```
    global score
```

```
    score+=int(res)
```

```
    score_label.config(text=f'Score: {score}')
```

```
def Progress_recorded():
```

```

global username, score, globalid, user_progress_data, quiz_subjects

new_row = pd.DataFrame([[username, score, globalid]],
columns=["Username", "Score", "Subject"])

user_progress_data = pd.concat([user_progress_data, new_row],
ignore_index=True)

user_progress_data.to_csv('Data/user_progress.csv', index=False)

create_piechart(score)

create_barchart(quiz_subjects,user_progress_data,username)

```

```

def show_stats(): #Showing Statistics frame

```

```

    global pie

    Stats_label.config(text=f"Congrats {username} for completing this
Quiz!!!")

    Final_score_label.configure(text=f"Score: {score}")

    pie=Image.open('Images/piechart.png')

    pie = pie.resize((400, 400))

    pie = ImageTk.PhotoImage(pie)

    piechart_label.config(image=pie,bg='#E9EDF1')

    show_frame(Stats)

```

```

def bargraph_show():

```

```

    global bargraph

```

```
bargraph=Image.open('Images/Barchart.png')
```

```
bargraph = bargraph.resize((400, 400))
```

```
bargraph = ImageTk.PhotoImage(bargraph)
```

```
Barchart_Image.config(image=bargraph)
```

```
def show_graph(pid):
```

```
    global username,user_progress_data,Linegraph
```

```
    create_linechart(pid,user_progress_data,username)
```

```
    Linegraph=Image.open('Images/LineChart.png')
```

```
    Linegraph = Linegraph.resize((400, 400))
```

```
    Linegraph = ImageTk.PhotoImage(Linegraph)
```

```
    Linegraph_Image.config(image=Linegraph)
```

```
def exit():
```

```
    root.destroy()
```

```
def restart():
```

```
    global qno, score, section_question, res, username,
```

```
    globalid,optname,user_response
```

```
    # Reset all necessary global variables
```

```
qno = 1

score = 0

section_question = None

res = None

globalid = None

optname=None

user_response=pd.DataFrame(columns=['Response'])


# Reset the score label and question display

score_label.config(text=f"Score: {score}")

Question_label.config(text="")


# Reset the options buttons

opt1.config(bg="#2eff70", text="")

opt2.config(bg="#2eff70", text="")

opt3.config(bg="#2eff70", text="")

opt4.config(bg="#2eff70", text="")


# Reset the submit button

submitbtn.config(state=NORMAL)
```


show_frame(Declar)

def getting_option_name(df,no,str):

txt=df.loc[no,str]

txt=txt.replace(' ','')

option_name=df.loc[no,txt]

return option_name

def option_name(label):

global optname

optname=label

def user_response_name(df,no,str):

txt=df.loc[no-1,str]

return txt

def adding_user_response():

global section_question,user_response,optname,qno

user_response.loc[qno-

1]=[user_response_name(section_question,qno,optname)]

```
def questionplate():
```

```
    global section_question,user_response
```

```
correct_answer_1.configure(text=getting_option_name(section_question,0,'  
Correct_Option'))
```

```
correct_answer_2.configure(text=getting_option_name(section_question,1,'  
Correct_Option'))
```

```
correct_answer_3.configure(text=getting_option_name(section_question,2,'  
Correct_Option'))
```

```
correct_answer_4.configure(text=getting_option_name(section_question,3,'  
Correct_Option'))
```

```
correct_answer_5.configure(text=getting_option_name(section_question,4,'  
Correct_Option'))
```

```
correct_answer_6.configure(text=getting_option_name(section_question,5,'  
Correct_Option'))
```

```
correct_answer_7.configure(text=getting_option_name(section_question,6,'  
Correct_Option'))
```

```
correct_answer_8.configure(text=getting_option_name(section_question,7,'  
Correct_Option'))
```

```
correct_answer_9.configure(text=getting_option_name(section_question,8,'
Correct_Option'))
```

```
correct_answer_10.configure(text=getting_option_name(section_question,9,'
Correct_Option'))
```

```
user_answer_1.configure(text=user_response.loc[0,'Response'])
```

```
user_answer_2.configure(text=user_response.loc[1,'Response'])
```

```
user_answer_3.configure(text=user_response.loc[2,'Response'])
```

```
user_answer_4.configure(text=user_response.loc[3,'Response'])
```

```
user_answer_5.configure(text=user_response.loc[4,'Response'])
```

```
user_answer_6.configure(text=user_response.loc[5,'Response'])
```

```
user_answer_7.configure(text=user_response.loc[6,'Response'])
```

```
user_answer_8.configure(text=user_response.loc[7,'Response'])
```

```
user_answer_9.configure(text=user_response.loc[8,'Response'])
```

```
user_answer_10.configure(text=user_response.loc[9,'Response'])
```

```
def color_wr():
```

```
    if user_answer_1.cget('text')==correct_answer_1.cget('text'):
```

```
        user_answer_1.configure(bg='#54f780')
```

```
    else:
```

```
user_answer_1.configure(bg='#fa6446')
```

```
if user_answer_2.cget('text')==correct_answer_2.cget('text'):
```

```
    user_answer_2.configure(bg='#54f780')
```

```
else:
```

```
    user_answer_2.configure(bg='#fa6446')
```

```
if user_answer_3.cget('text')==correct_answer_3.cget('text'):
```

```
    user_answer_3.configure(bg='#54f780')
```

```
else:
```

```
    user_answer_3.configure(bg='#fa6446')
```

```
if user_answer_4.cget('text')==correct_answer_4.cget('text'):
```

```
    user_answer_4.configure(bg='#54f780')
```

```
else:
```

```
    user_answer_4.configure(bg='#fa6446')
```

```
if user_answer_5.cget('text')==correct_answer_5.cget('text'):
```

```
    user_answer_5.configure(bg='#54f780')
```

```
else:
```

```
    user_answer_5.configure(bg='#fa6446')
```

if user_answer_6.cget('text')==correct_answer_6.cget('text'):

user_answer_6.configure(bg='#54f780')

else:

user_answer_6.configure(bg='#fa6446')

if user_answer_7.cget('text')==correct_answer_7.cget('text'):

user_answer_7.configure(bg='#54f780')

else:

user_answer_7.configure(bg='#fa6446')

if user_answer_8.cget('text')==correct_answer_8.cget('text'):

user_answer_8.configure(bg='#54f780')

else:

user_answer_8.configure(bg='#fa6446')

if user_answer_9.cget('text')==correct_answer_9.cget('text'):

user_answer_9.configure(bg='#54f780')

else:

user_answer_9.configure(bg='#fa6446')

if user_answer_10.cget('text')==correct_answer_10.cget('text'):

user_answer_10.configure(bg='#54f780')

else:

user_answer_10.configure(bg='#fa6446')

def boo():

global ch

ch=1

def check_none(choice_flag):

if choice_flag == 0: # No option selected

messagebox.showwarning("No Option Selected", "Please choose an option before submitting.")

return False # Stop execution

return True # Proceed

def restart_characterflag():

global ch

ch=0

user_pass_data=pd.read_csv('Data/users.csv')

questions_data=pd.read_csv('Data/krishna.csv')

user_progress_data=pd.read_csv('Data/user_progress.csv')

List of subjects for the quiz

```
quiz_subjects = [  
    "General Knowledge",  
    "Science",  
    "Mathematics",  
    "History",  
    "Geography",  
    "Literature",  
    "Technology",  
    "Sports",  
    "Entertainment",  
    "Mythology",  
    "Current Affairs",  
    "Pop Culture"  
]
```

qno=1

globalid=NONE

score=0

section_question=NONE

res=NONE

user_response=pd.DataFrame(columns=['Response'])

optname=None

ch=0

pie=None

bar=None

line=None

fon="vendana"

root = Tk()

root.geometry("900x600")

root.configure(background="#E9EDF1")

root.resizable(0, 0)

root.title('QUIZ')

Frames

Welcome = Frame(root, background="#E9EDF1")

Credit_page=Frame(root,background='#E9EDF1')


```

Login = Frame(root, background="#E9EDF1")

Signup = Frame(root, background="#E9EDF1")

Declar = Frame(root, background="#E9EDF1")

Question = Frame(root, background="#E9EDF1")

Stats=Frame(root,background='#ffffff')

Scorechart_frame=Frame(root,background='#E9EDF1')

Barchart=Frame(root,background='#ffffff')

SProgress=Frame(root,background="#E9EDF1")

LinegraphFrame=Frame(root,background='#ffffff')


for frame in (Welcome, Credit_page,Signup,
Login,Declar,Question,Stats,Scorechart_frame,Barchart,SProgress,Linegra
phFrame):

    frame.grid(row=0, column=0, sticky="nsew")


def show_frame(frame):

    frame.tkraise()


# Show the Welcome page initially

show_frame(Welcome)

```

Welcome Page

welcome_label = Label(

Welcome, text="Welcome to Quiz", font=(fon, 36, "bold"),
bg="#E9EDF1"

)

welcome_label.pack(pady=95,padx=250) # Centered horizontally

play_btn = Button(

Welcome,

text="Play",

font=(fon, 20),

width=15,

height=2,

bg="#D6DBDF",

command=lambda: show_frame(Signup)

)

play_btn.pack(pady=(100,0))

credit_btn=Button(Welcome,

```

text="Credits",

font=(fon, 17),

width=15,

height=2,

bg="#D6DBDF",

command=lambda: show_frame(Credit_page))

credit_btn.pack(pady=(56,0))

#Credit Page

GUI_cred=Label(Credit_page,text="GUI: Omkaar",font=(fon,18))

GUI_cred.pack(pady=(150,10))

Backend_cred=Label(Credit_page,text="Backend: Omkaar & Krishna
bansal",font=(fon,18))

Backend_cred.pack(pady=(10,10))

Datacollection_cred=Label(Credit_page,text="Data Collection: Krishna
Bansal",font=(fon,18))

Datacollection_cred.pack(pady=(10,10))

# Sign Up Page

signup_label = Label(

    Signup, text="Sign Up Page", font=(fon, 36, "bold"), bg="#E9EDF1"

```

```
)  
  
signup_label.pack(pady=9,padx=250) # Centered horizontally
```

```
signup_username_txt=Label(  
  
    Signup,  
  
    text="username",  
  
    font=(fon,28),  
  
    bg="#E9EDF1"
```

```
)  
  
signup_username_txt.pack(pady=(79,0),padx=(250,0),anchor="w")
```

```
signup_username_ent=Entry(  
  
    Signup,  
  
    width=20,  
  
    font=(fon,22)
```

```
)  
  
signup_username_ent.pack(pady=(13,0),padx=0)
```

```
signup_password_txt=Label(  
  
    Signup,  
  
    text="password",
```

```
font=(fon,28),

bg="#E9EDF1"

)

signup_password_txt.pack(pady=(25,0),padx=(250,0),anchor="w")


signup_password_ent=Entry(

    Signup,

    width=20,

    font=(fon,22),

    show="*"

)

signup_password_ent.pack(pady=(13,10),padx=0)


subm_btn=Button(

    Signup,

    text="Submit",

    bg="#5582BE",

    width=20,

    font=(fon,20),

    command=lambda :add_user()

)
```

```
subm_btn.pack(pady=(20,90))
```

```
login_frame = Frame(Signup, bg="#E9EDF1")
```

```
login_frame.pack(pady=(10, 0))
```

```
login_txt=Label(
```

```
    login_frame,
```

```
    text="already have credentials",
```

```
    bg="#E9EDF1",
```

```
    fg="#5582BE",
```

```
    font=(fon,12)
```

```
)
```

```
login_txt.pack(side="left", padx=(0, 5))
```

```
login_btn=Button(
```

```
    login_frame,
```

```
    text="login",
```

```
    command=lambda: show_frame(Login)
```

```
)
```

```
login_btn.pack(side="left")
```

#Login

login_label = Label(

Login, text="login Page", font=(fon, 36, "bold"), bg="#E9EDF1"

)

login_label.pack(pady=9,padx=250) # Centered horizontally

login_username_txt=Label(

Login,

text="username",

font=(fon,28),

bg="#E9EDF1"

)

login_username_txt.pack(pady=(79,0),padx=(250,0),anchor="w")

login_username_ent=Entry(

Login,

width=20,

```
font=(fon,22)

)

login_username_ent.pack(pady=(13,0),padx=0)

login_password_txt=Label(

    Login,

    text="password",

    font=(fon,28),

    bg="#E9EDF1",

)

login_password_txt.pack(pady=(25,0),padx=(250,0),anchor="w")

login_password_ent=Entry(

    Login,

    width=20,

    font=(fon,22),

    show='*'

)

login_password_ent.pack(pady=(13,10),padx=0)

sub_btn=Button(
```



```
Login,  
  
text="Submit",  
  
bg="#5582BE",  
  
width=20,  
  
font=(fon,20),  
  
command=lambda :login_user()  
  
)  
  
sub_btn.pack(pady=(20,90))  
  
  
  
signup_frame = Frame(Login, bg="#E9EDF1")  
  
signup_frame.pack(pady=(10, 0))  
  
  
  
signup_txt=Label(  
  
    signup_frame,  
  
    text="Don't have credentials",  
  
    bg="#E9EDF1",  
  
    fg="#5582BE",  
  
    font=(fon,12)  
  
)  
  
signup_txt.pack(side="left", padx=(0, 5))
```

```
signup_btn=Button(  
    signup_frame,  
    text="signup",  
    command=lambda: show_frame(Signup)  
)
```

```
signup_btn.pack(side="left")
```

```
#Declare
```

```
# Declare Frame
```

```
declar_label = Label(Declar, text="Choose the subject", font=(fon, 45),  
bg="#E9EDF1")
```

```
declar_label.pack()
```

```
btnframe1 = Frame(Declar, bg="#E9EDF1")
```

```
btnframe1.pack(pady=(70, 0))
```

```
# Buttons with symmetrical distance using grid
```

```
btn1 = Button(btnframe1, text=str(quiz_subjects[0]), width=15, font=(fon,  
20),bg='#C0EF76',command=lambda :  
(show_frame(Question),Show_Question('General Knowledge')))  
btn1.grid(row=0, column=0, padx=20, pady=30)
```

```
btn2 = Button(btnframe1, text=str(quiz_subjects[1]), width=15, font=(fon,  
20),bg='#C0EF76',command=lambda :  
(show_frame(Question),Show_Question('Science')))  
btn2.grid(row=0, column=1, padx=20, pady=30)
```

```
btn3 = Button(btnframe1, text=str(quiz_subjects[2]), width=15, font=(fon,  
20),bg='#C0EF76',command=lambda :  
(show_frame(Question),Show_Question('Mathematics')))  
btn3.grid(row=0, column=2, padx=20, pady=30)
```

```
btn4 = Button(btnframe1, text=str(quiz_subjects[3]), width=15, font=(fon,  
20),bg='#C0EF76',command=lambda :  
(show_frame(Question),Show_Question('History')))  
btn4.grid(row=1, column=0, padx=20, pady=30)
```

```
btn5 = Button(btnframe1, text=str(quiz_subjects[4]), width=15, font=(fon,  
20),bg='#C0EF76',command=lambda :  
(show_frame(Question),Show_Question('Geography')))
```

```
btn5.grid(row=1, column=1, padx=20, pady=30)
```

```
btn6 = Button(btnframe1, text=str(quiz_subjects[5]), width=15, font=(fon,  
20),bg='#C0EF76',command=lambda :  
(show_frame(Question),Show_Question('Literature')))
```

```
btn6.grid(row=1, column=2, padx=20, pady=30)
```

```
btn7 = Button(btnframe1, text=str(quiz_subjects[6]), width=15, font=(fon,  
20),bg='#C0EF76',command=lambda :  
(show_frame(Question),Show_Question('Technology')))
```

```
btn7.grid(row=2, column=0, padx=20, pady=30)
```

```
btn8 = Button(btnframe1, text=str(quiz_subjects[7]), width=15, font=(fon,  
20),bg='#C0EF76',command=lambda :  
(show_frame(Question),Show_Question('Sports')))
```

```
btn8.grid(row=2, column=1, padx=20, pady=30)
```

```
btn9 = Button(btnframe1, text=str(quiz_subjects[8]), width=15, font=(fon,  
20),bg='#C0EF76',command=lambda :  
(show_frame(Question),Show_Question('Entertainment')))
```

```
btn9.grid(row=2, column=2, padx=20, pady=30)
```

```
btn10 = Button(btnframe1, text=str(quiz_subjects[9]), width=15, font=(fon,  
20),bg='#C0EF76',command=lambda :  
(show_frame(Question),Show_Question('Mythology')))  
btn10.grid(row=3, column=0, padx=20, pady=30)
```

```
btn11 = Button(btnframe1, text=str(quiz_subjects[10]), width=15, font=(fon,  
20),bg='#C0EF76',command=lambda :  
(show_frame(Question),Show_Question('Current Affairs')))  
btn11.grid(row=3, column=1, padx=20, pady=30)
```

```
btn12 = Button(btnframe1, text=str(quiz_subjects[11]), width=15, font=(fon,  
20),bg='#C0EF76',command=lambda :  
(show_frame(Question),Show_Question('Pop Culture')))  
btn12.grid(row=3, column=2, padx=20, pady=30)
```

#Question

```
Question_label=Label(Question,text=str(qno)  
+''',bg='#E9EDF1',font=(fon,22),wraplength=900,justify="left")  
Question_label.pack(anchor='w',padx=(4,6),pady=(10,5))
```

```
score_label=Label(Question,text=("Score:  
"+str(score)),bg="#E9EDF1",font=(fon,18))
```

```
score_label.pack(anchor='e')
```

```
optionframe=Frame(Question,bg="#E9EDF1")
```

```
optionframe.pack(padx=(120,0),anchor='w',pady=(120,0))
```

```
opt1=Button(optionframe,text="",anchor='w',font=(fon,18),bg="#2eff70",w  
idth=17,command=lambda :(check('Option A'),color_button('Option  
A'),option_name('OptionA'),boo()))
```

```
opt1.grid(row=0,column=0)
```

```
opt2=Button(optionframe,text="",anchor='w',font=(fon,18),bg="#2eff70",w  
idth=17,command=lambda :(check('Option B'),color_button('Option  
B'),option_name('OptionB'),boo()))
```

```
opt2.grid(row=0,column=2,padx=(120,0))
```

```
opt3=Button(optionframe,text="",anchor='w',font=(fon,18),bg="#2eff70",w  
idth=17,command=lambda :(check('Option C'),color_button('Option  
C'),option_name('OptionC'),boo()))
```

```
opt3.grid(row=1,column=0,pady=(45,0))
```

```
opt4=Button(optionframe,text="",anchor='w',font=(fon,18),bg="#2eff70",width=17,command=lambda :(check('Option D'),color_button('Option D'),option_name('OptionD'),boo()))
```

```
opt4.grid(row=1,column=2,padx=(120,0),pady=(45,0))
```

```
submitbtn=Button(Question,text='Submit',font=(fon,20),width=17,bg="#cef522",
```

```
        command=lambda:(None if not check_none(ch) else  
(adding_user_response(), score_add(), increase_qno(),  
show_Questions_forcont(),restart_characterflag())))
```

```
submitbtn.pack(anchor="s",pady=(50))
```

```
#stats
```

```
Stats_label=Label(Stats,text="",font=(fon,20),bg="#ffffff")
```

```
Stats_label.pack()
```

```
Final_score_label=Label(Stats,text='score',font=(fon,24),bg="#ffffff")
```

```
Final_score_label.pack(anchor='w',padx=(45,7),pady=(23,0))
```

```
graph_frame=Frame(Stats,bg='#ffffff')
```

```
graph_frame.pack()
```

```
piechart_label=Label(graph_frame,image=pie)
```

```
piechart_label.grid(row=0,column=0,padx=(3,7))
```

```
graphbtn_frame=Frame(graph_frame,bg="#ffffff")
```

```
graphbtn_frame.grid(row=0,column=1)
```

```
ScoreBoard_btn=Button(graphbtn_frame,text='View your  
Answers',font=(fon,14),command=lambda:
```

```
(show_frame(Scorechart_frame),questionplate(),color_wr()))
```

```
ScoreBoard_btn.grid(row=0,column=0,padx=(7,9),pady=(0,10))
```

```
barbtn=Button(graphbtn_frame,text='View your highest in each  
subject',font=(fon,14),command=lambda :
```

```
(bargraph_show(),show_frame(Barchart)))
```

```
barbtn.grid(row=1,column=0,padx=(7,9),pady=(0,10))
```

```
linebtn=Button(graphbtn_frame,text='View your progress in each  
subject',font=(fon,14),command=lambda :(show_frame(SProgress)))
```

```
linebtn.grid(row=2,column=0,padx=(7,9),pady=(5,8))
```



```
restartbtn=Button(graphbtn_frame,text='Back',font=(fon,14),command=la  
mbda :restart())
```

```
restartbtn.grid(row=3,column=0,padx=(7,9),pady=(5,8))
```

```
exitbtn=Button(graphbtn_frame,text='exit',font=(fon,14),command=lambda  
: exit())
```

```
exitbtn.grid(row=4,column=0,padx=(7,9),pady=(5,8))
```

```
#Scorechart
```

```
scorechart = Frame(Scorechart_frame, width=60)
```

```
scorechart.pack(anchor='center', pady=(134, 0))
```

```
# Question Number Label
```

```
Question_number_label = Label(scorechart, text="Question  
number",width=14,font=(fon,15))
```

```
Question_number_label.grid(row=0, column=0)
```

```
# Question Numbers (1 to 10)
```

```
for i in range(1, 11):
```

```
    question_number = Label(scorechart, text=i, width=14, font=(fon,  
15),bg='#E9EDF1')
```

```
question_number.grid(row=i, column=0)
```

```
# User Answer Label
```

```
user_answer_label = Label(scorechart, text="Your  
Answer",width=18,font=(fon,15),bg='#E9EDF1')
```

```
user_answer_label.grid(row=0, column=1)
```

```
user_answer_1 = Label(scorechart, text='abdracadraabr', width=18,  
font=(fon, 15))
```

```
user_answer_1.grid(row=1, column=1)
```

```
user_answer_2= Label(scorechart, text='abdracadraabr', width=18,  
font=(fon, 15))
```

```
user_answer_2.grid(row=2, column=1)
```

```
user_answer_3 = Label(scorechart, text='abdracadraabr', width=18,  
font=(fon, 15))
```

```
user_answer_3.grid(row=3, column=1)
```

```
user_answer_4 = Label(scorechart, text='abdracadraabr', width=18,  
font=(fon, 15))
```

```
user_answer_4.grid(row=4, column=1)
```

```
user_answer_5 = Label(scorechart, text='abdracadraabr', width=18,  
font=(fon, 15))
```

```
user_answer_5.grid(row=5, column=1)
```

```
user_answer_6 = Label(scorechart, text='abdracadraabr', width=18,  
font=(fon, 15))
```

```
user_answer_6.grid(row=6, column=1)
```

```
user_answer_7 = Label(scorechart, text='abdracadraabr', width=18,  
font=(fon, 15))
```

```
user_answer_7.grid(row=7, column=1)
```

```
user_answer_8 = Label(scorechart, text='abdracadraabr', width=18,  
font=(fon, 15))
```

```
user_answer_8.grid(row=8, column=1)
```

```
user_answer_9 = Label(scorechart, text='abdracadraabr', width=18,  
font=(fon, 15))
```

```
user_answer_9.grid(row=9, column=1)
```

```
user_answer_10 = Label(scorechart, text='abdracadraabr', width=18,  
font=(fon, 15))
```

```
user_answer_10.grid(row=10, column=1)
```

```
Correct_answer_label = Label(scorechart, text="Correct  
Answer",width=18,font=(fon,15),bg='#E9EDF1')
```

```
Correct_answer_label.grid(row=0, column=2)
```

```
correct_answer_1 = Label(scorechart, text='abdracadraabr',
anchor='w',width=18, font=(fon, 15),bg='#54f780')

correct_answer_1.grid(row=1, column=2)

correct_answer_2= Label(scorechart, text='abdracadraabr',anchor='w',
width=18, font=(fon, 15),bg='#54f780')

correct_answer_2.grid(row=2, column=2)

correct_answer_3 = Label(scorechart, text='abdracadraabr',anchor='w',
width=18, font=(fon, 15),bg='#54f780')

correct_answer_3.grid(row=3, column=2)

correct_answer_4 = Label(scorechart, text='abdracadraabr',anchor='w',
width=18, font=(fon, 15),bg='#54f780')

correct_answer_4.grid(row=4, column=2)

correct_answer_5 = Label(scorechart, text='abdracadraabr',anchor='w',
width=18, font=(fon, 15),bg='#54f780')

correct_answer_5.grid(row=5, column=2)

correct_answer_6 = Label(scorechart, text='abdracadraabr',anchor='w',
width=18, font=(fon, 15),bg='#54f780')

correct_answer_6.grid(row=6, column=2)

correct_answer_7 = Label(scorechart, text='abdracadraabr',anchor='w',
width=18, font=(fon, 15),bg='#54f780')

correct_answer_7.grid(row=7, column=2)

correct_answer_8 = Label(scorechart, text='abdracadraabr',anchor='w',
width=18, font=(fon, 15),bg='#54f780')
```

```
correct_answer_8.grid(row=8, column=2)
```

```
correct_answer_9 = Label(scorechart, text='abdracadraabr',anchor='w',  
width=18, font=(fon, 15),bg='#54f780')
```

```
correct_answer_9.grid(row=9, column=2)
```

```
correct_answer_10 = Label(scorechart, text='abdracadraabr',anchor='w',  
width=18, font=(fon, 15),bg='#54f780')
```

```
correct_answer_10.grid(row=10, column=2)
```

```
Cbackbtn=Button(scorechart,bg='blue',text="Back",command=lambda :sh  
ow_frame(Stats))
```

```
Cbackbtn.grid(row=11,column=1,pady=(5,8))
```

```
#Barchart
```

```
bargraph=None
```

```
Barchart_Image=Label(Barchart,image=bargraph)
```

```
Barchart_Image.pack()
```

```
backbtn=Button(Barchart,bg='blue',text="Back",command=lambda :show  
_frame(Stats))
```

```
backbtn.pack(anchor='s')
```

#SProgress

Pbtnframe = Frame(SProgress, bg="#E9EDF1")

Pbtnframe.pack(pady=(70, 0))

**Pbtn1 = Button(Pbtnframe, text=str(quiz_subjects[0]), width=15, font=(fon,
20),bg='#C0EF76'**

,command=lambda :

(show_frame(LinegraphFrame),show_graph(quiz_subjects[0])))

Pbtn1.grid(row=0, column=0, padx=20, pady=30)

**Pbtn2 = Button(Pbtnframe, text=str(quiz_subjects[1]), width=15, font=(fon,
20),bg='#C0EF76'**

,command=lambda :

(show_frame(LinegraphFrame),show_graph(quiz_subjects[1])))

Pbtn2.grid(row=0, column=1, padx=20, pady=30)

**Pbtn3 = Button(Pbtnframe, text=str(quiz_subjects[2]), width=15, font=(fon,
20),bg='#C0EF76'**

,command=lambda :

(show_frame(LinegraphFrame),show_graph(quiz_subjects[2])))

Pbtn3.grid(row=0, column=2, padx=20, pady=30)

```
Pbtn4 = Button(Pbtnframe, text=str(quiz_subjects[3]), width=15, font=(fon,  
20),bg='#C0EF76'
```

```
        ,command=lambda :  
(show_frame(LinegraphFrame),show_graph(quiz_subjects[3]))
```

```
Pbtn4.grid(row=1, column=0, padx=20, pady=30)
```

```
Pbtn5 = Button(Pbtnframe, text=str(quiz_subjects[4]), width=15, font=(fon,  
20),bg='#C0EF76'
```

```
        ,command=lambda :  
(show_frame(LinegraphFrame),show_graph(quiz_subjects[4]))
```

```
Pbtn5.grid(row=1, column=1, padx=20, pady=30)
```

```
Pbtn6 = Button(Pbtnframe, text=str(quiz_subjects[5]), width=15, font=(fon,  
20),bg='#C0EF76'
```

```
        ,command=lambda :  
(show_frame(LinegraphFrame),show_graph(quiz_subjects[5]))
```

```
Pbtn6.grid(row=1, column=2, padx=20, pady=30)
```

```
Pbtn7 = Button(Pbtnframe, text=str(quiz_subjects[6]), width=15, font=(fon,  
20),bg='#C0EF76'
```

```
        ,command=lambda :  
(show_frame(LinegraphFrame),show_graph(quiz_subjects[6]))
```

```
Pbtn7.grid(row=2, column=0, padx=20, pady=30)
```

```
Pbtn8 = Button(Pbtnframe, text=str(quiz_subjects[7]), width=15, font=(fon,  
20),bg='#C0EF76'
```

```
,command=lambda :
```

```
(show_frame(LinegraphFrame),show_graph(quiz_subjects[7]))
```

```
Pbtn8.grid(row=2, column=1, padx=20, pady=30)
```

```
Pbtn9 = Button(Pbtnframe, text=str(quiz_subjects[8]), width=15, font=(fon,  
20),bg='#C0EF76'
```

```
,command=lambda :
```

```
(show_frame(LinegraphFrame),show_graph(quiz_subjects[8]))
```

```
Pbtn9.grid(row=2, column=2, padx=20, pady=30)
```

```
Pbtn10 = Button(Pbtnframe, text=str(quiz_subjects[9]), width=15,  
font=(fon, 20),bg='#C0EF76'
```

```
,command=lambda :
```

```
(show_frame(LinegraphFrame),show_graph(quiz_subjects[9]))
```

```
Pbtn10.grid(row=3, column=0, padx=20, pady=30)
```

```
Pbtn11 = Button(Pbtnframe, text=str(quiz_subjects[10]), width=15,  
font=(fon, 20),bg='#C0EF76'
```



```
,command=lambda :  
(show_frame(LinegraphFrame),show_graph(quiz_subjects[10]))  
  
Pbtn11.grid(row=3, column=1, padx=20, pady=30)  
  
Pbtn12 = Button(Pbtnframe, text=str(quiz_subjects[11]), width=15,  
font=(fon, 20),bg='#C0EF76'
```

```
,command=lambda :  
(show_frame(LinegraphFrame),show_graph(quiz_subjects[11]))  
  
Pbtn12.grid(row=3, column=2, padx=20, pady=30)
```

```
Pbackbtn=Button(SProgress,bg='blue',text="Back"
```

```
,command=lambda :show_frame(Stats))  
  
Pbackbtn.pack()
```

```
#LinegraphFrame
```

```
Linegraph=None
```

```
Linegraph_Image=Label(LinegraphFrame,image=Linegraph)
```

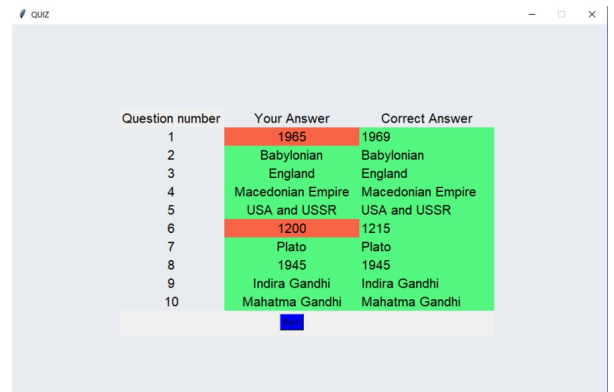
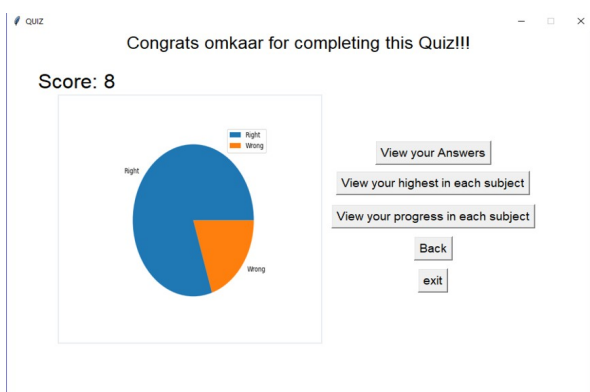
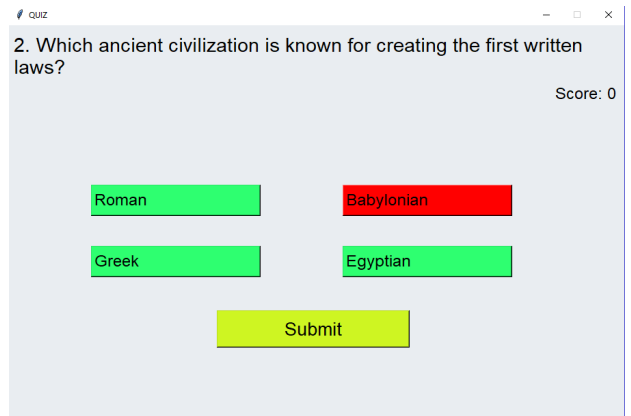
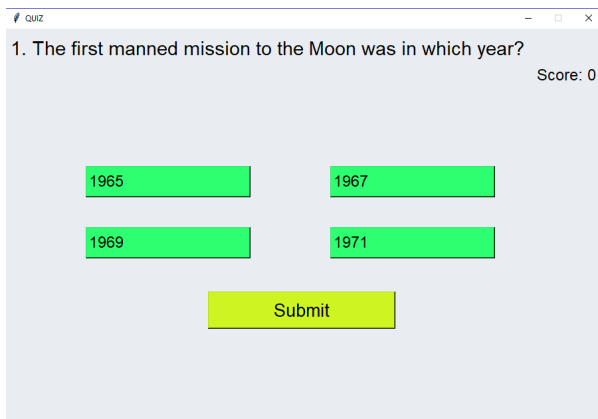
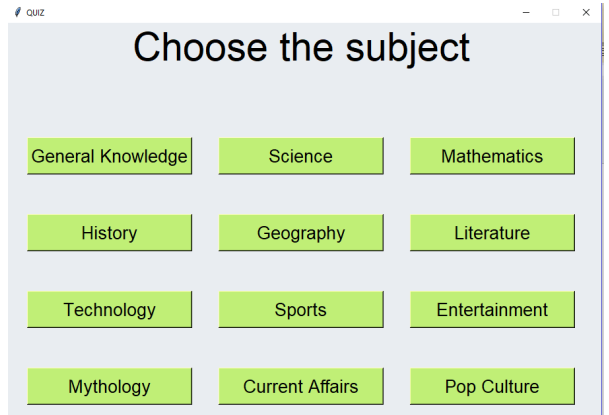
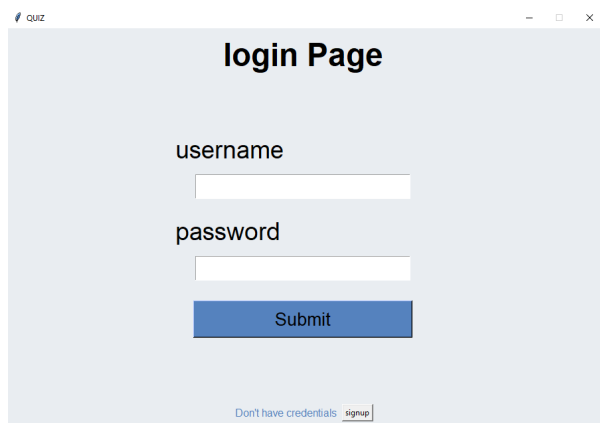
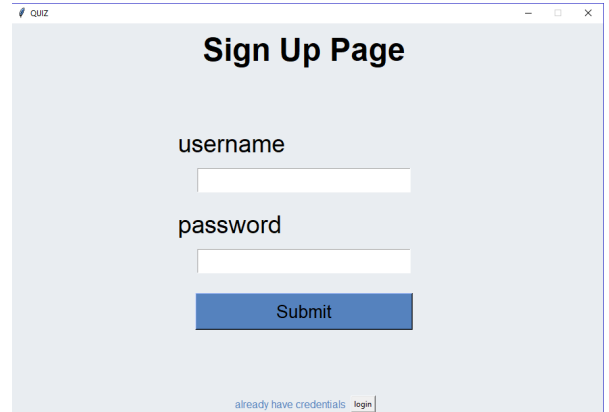
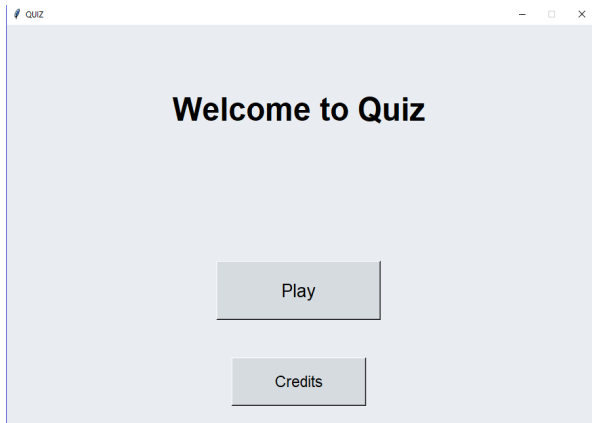
```
Linegraph_Image.grid(row=0,column=0)
```

```
Lbackbtn=Button(LinegraphFrame,bg='blue',text='Back',command=lamb  
da :show_frame(SProgress))
```

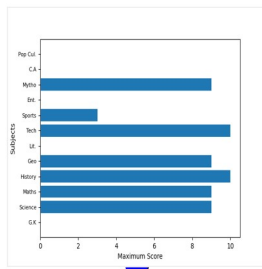
```
Lbackbtn.grid(row=1,column=0)
```

```
root.mainloop()
```

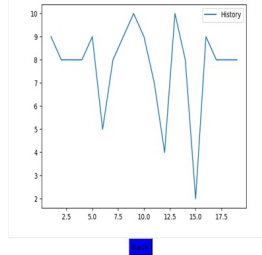
OUTPUT



QUIZ



QUIZ



QUIZ

GUI: Omkaar
Backend: Omkaar & Krishna bansal
Data Collection: Krishna Bansal

BIBLIOGRAPHY

Books

- Informatics Practices With Python - Class XI By : Sumita Arora
- Informatics Practices With Python - Class XII By : Sumita Arora
- Informatics Practices With Python - Class XII By : Preeti Arora

Websites

- <https://www.dev.mysql.com>
- <https://www.python.org>
- <https://www.w3resource.com>
