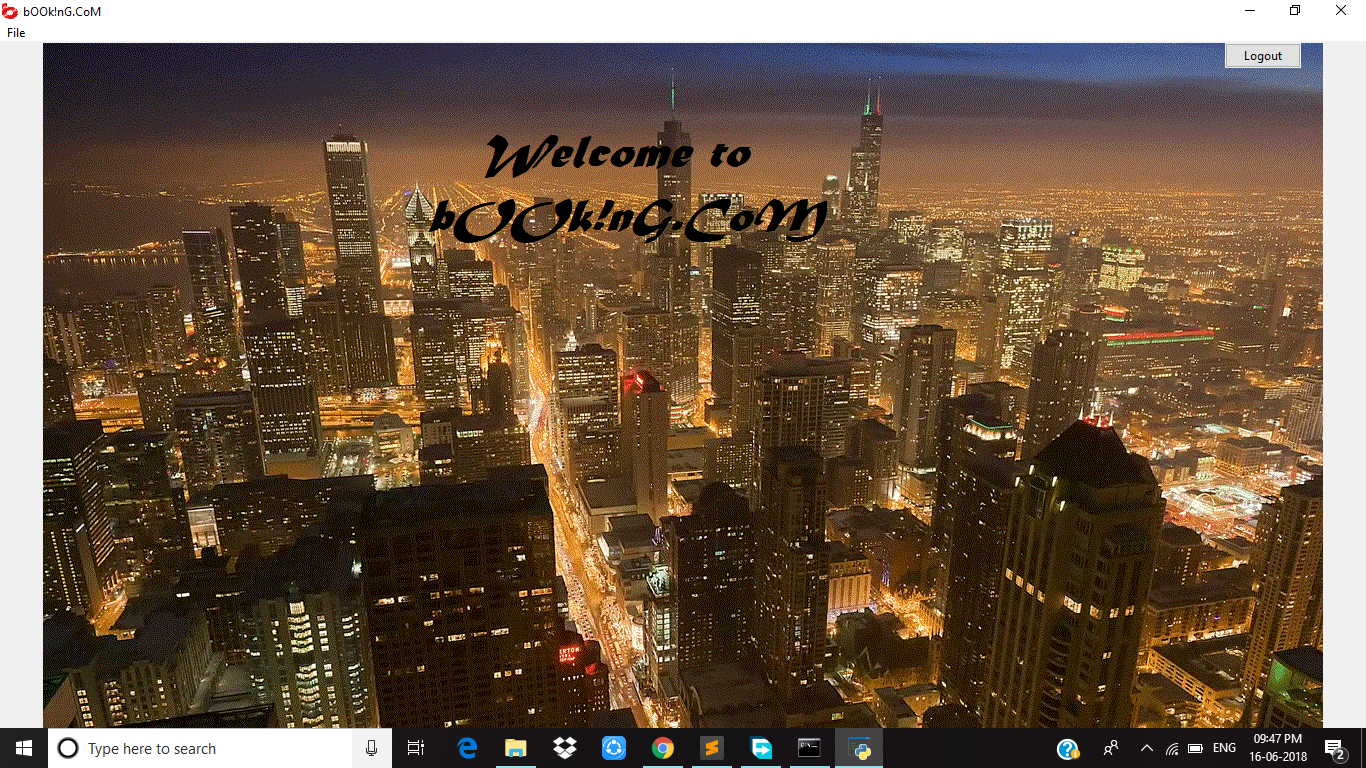
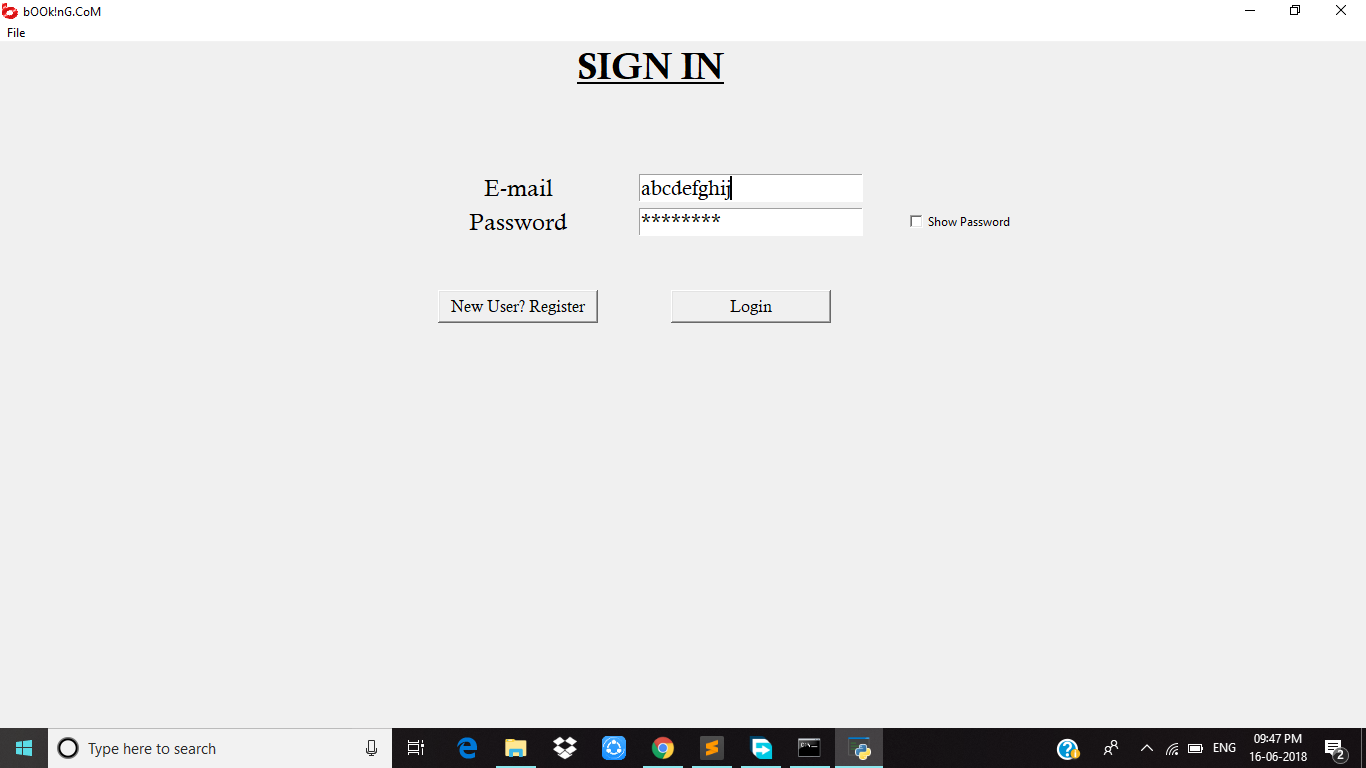
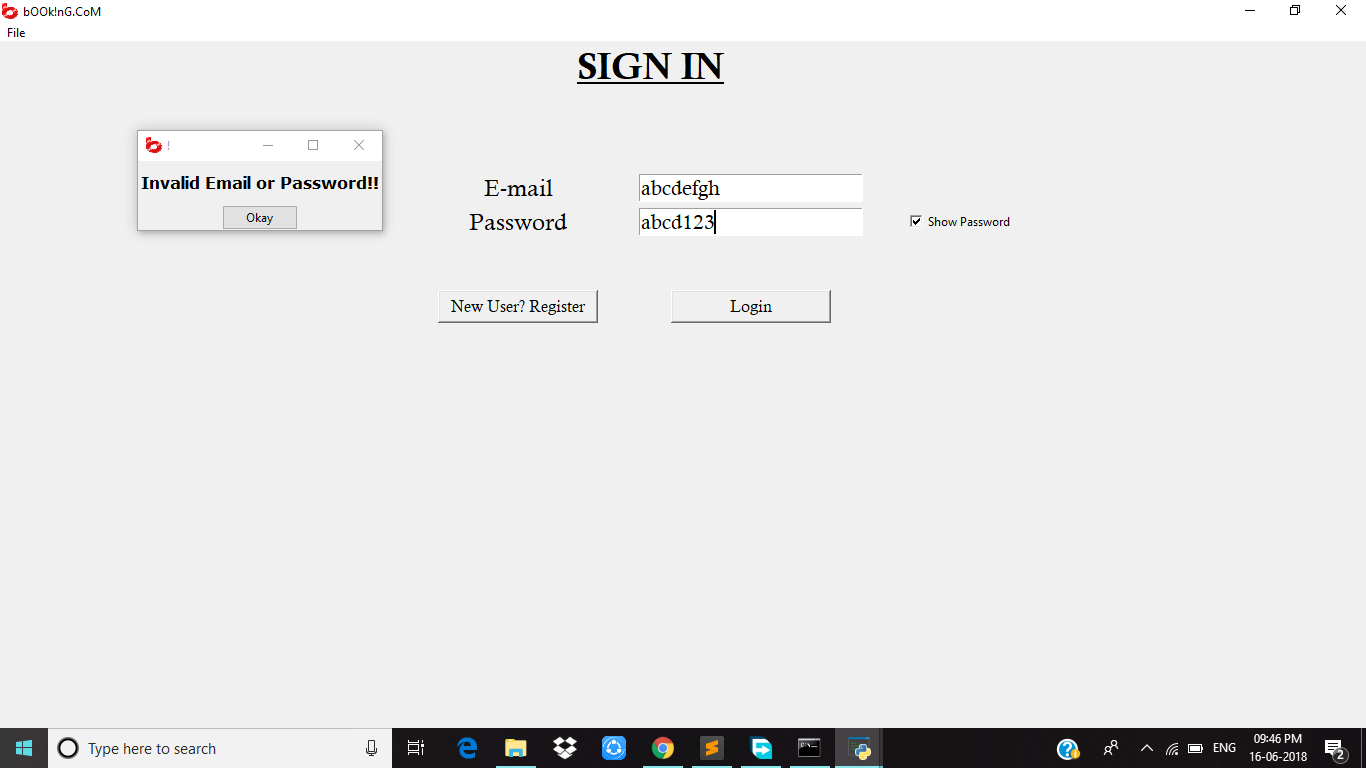
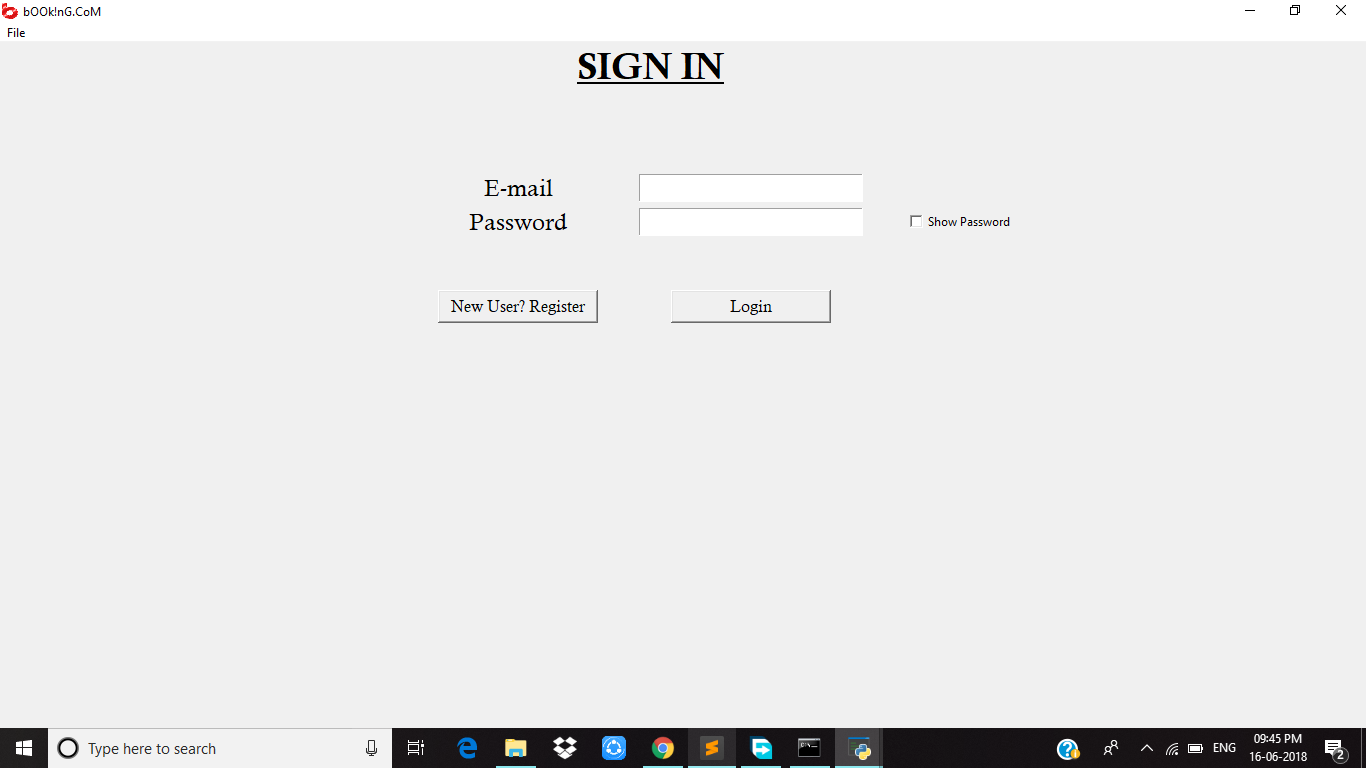
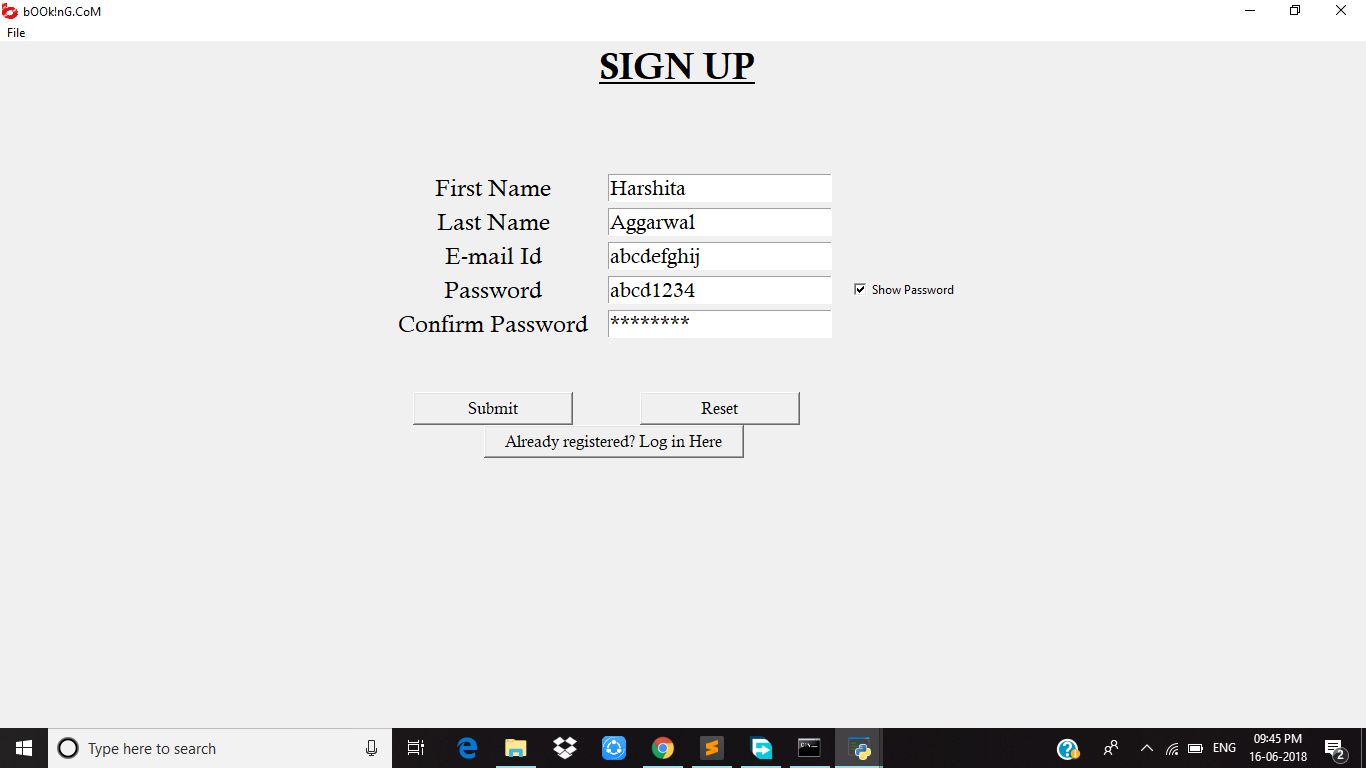
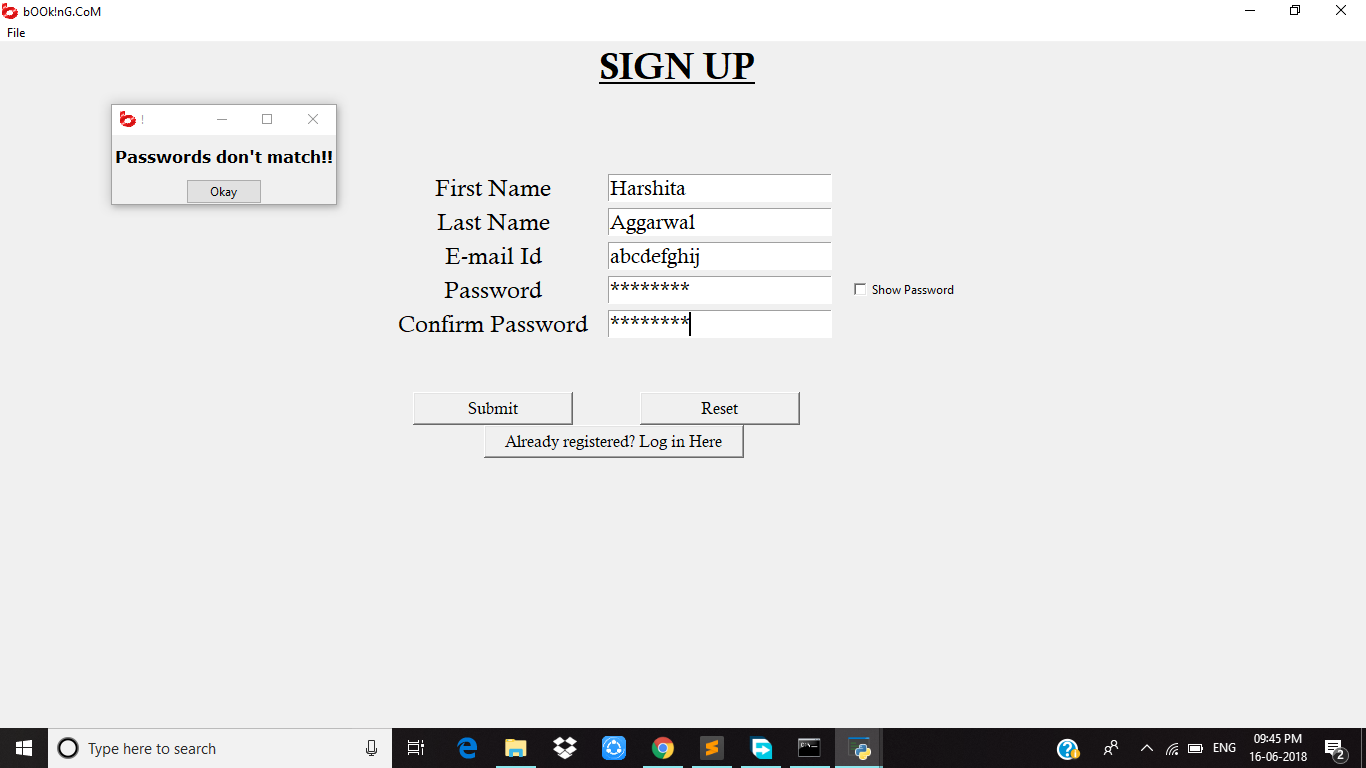
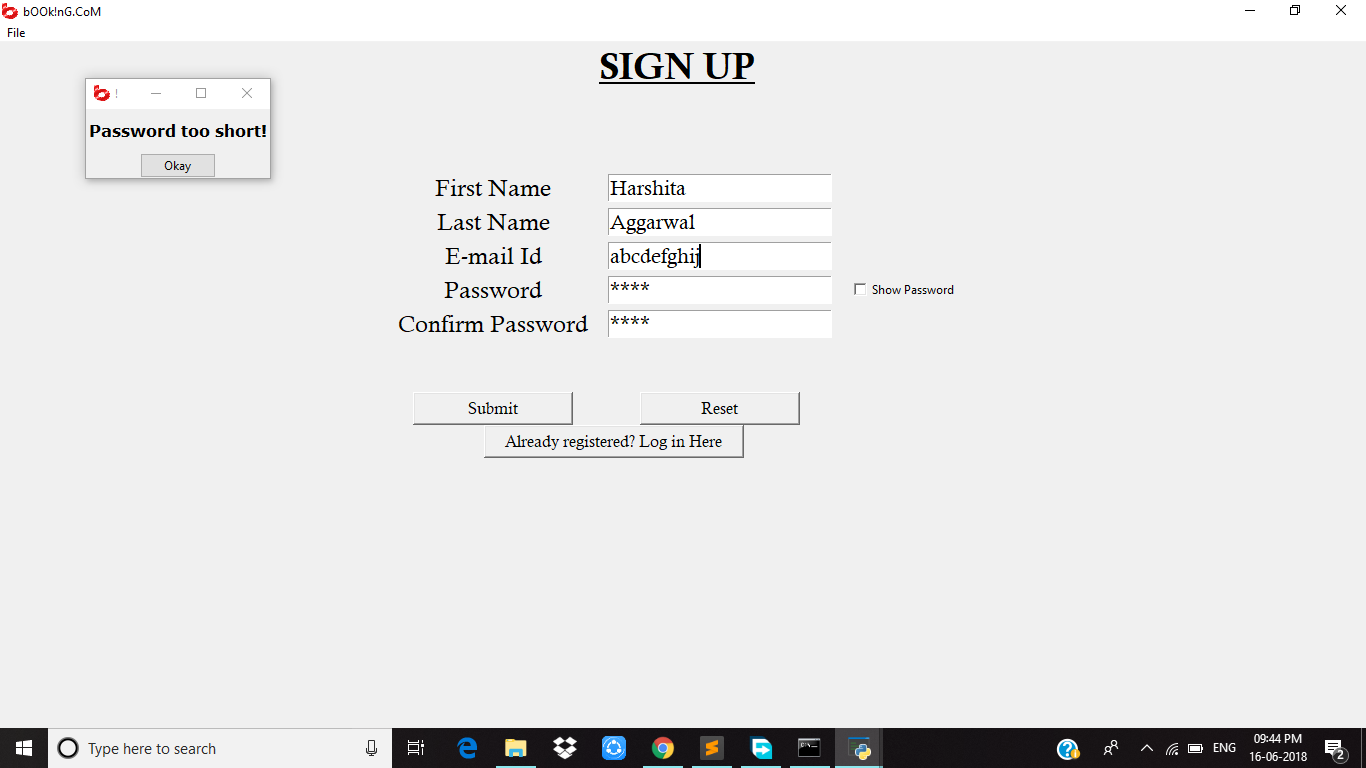
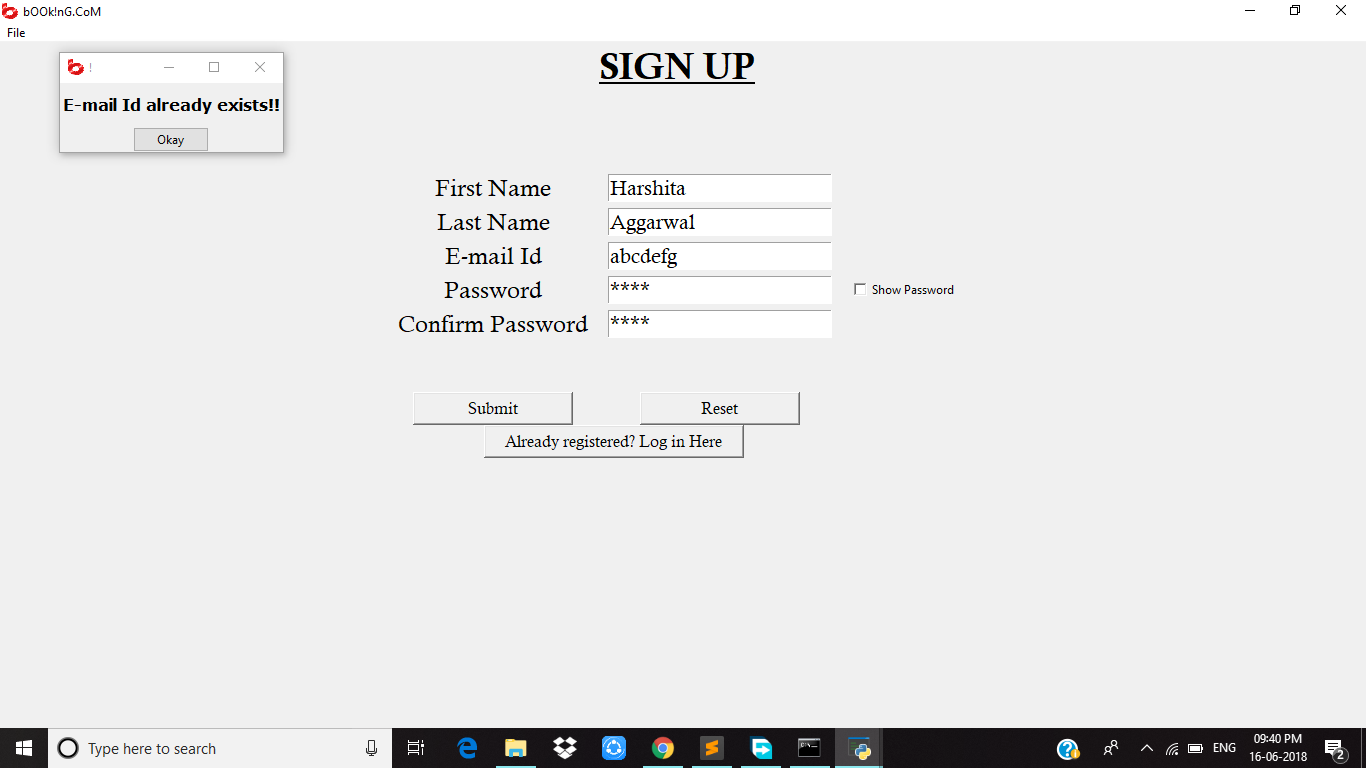
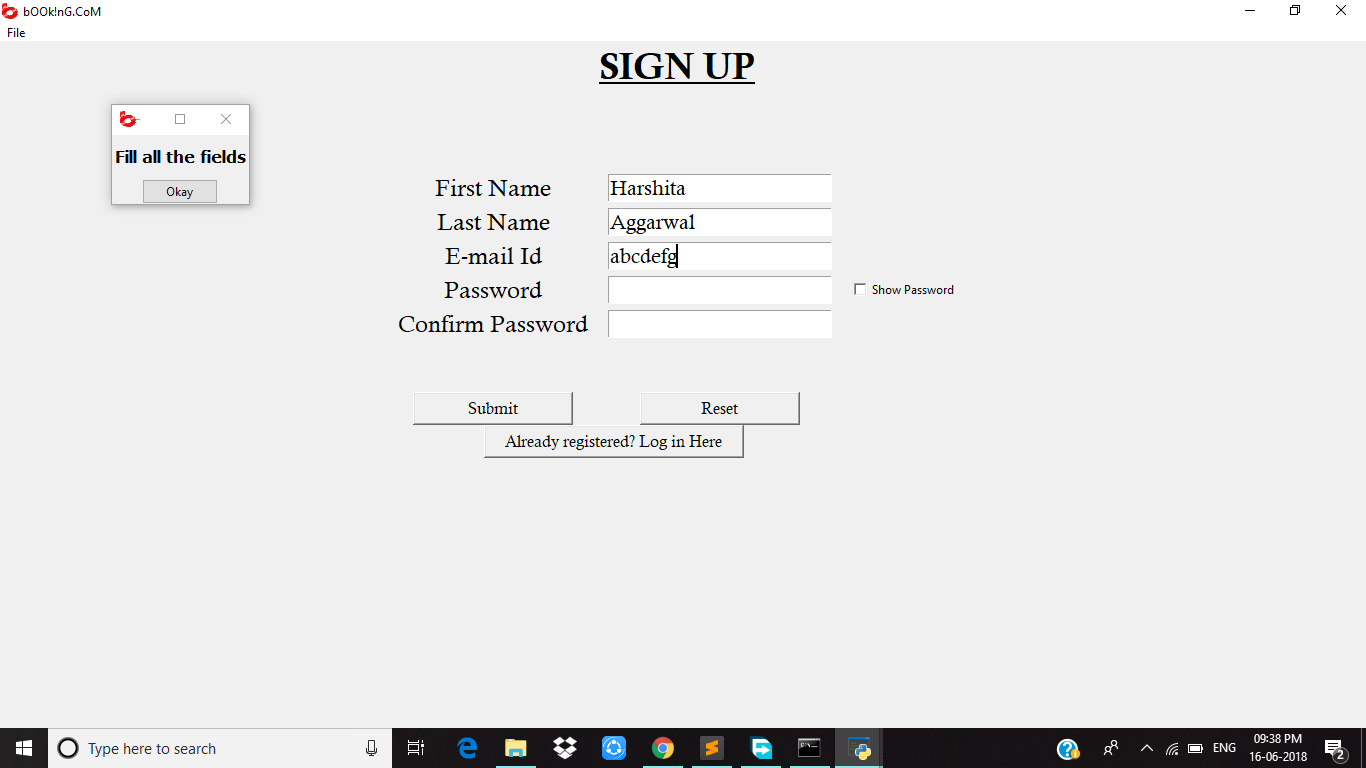
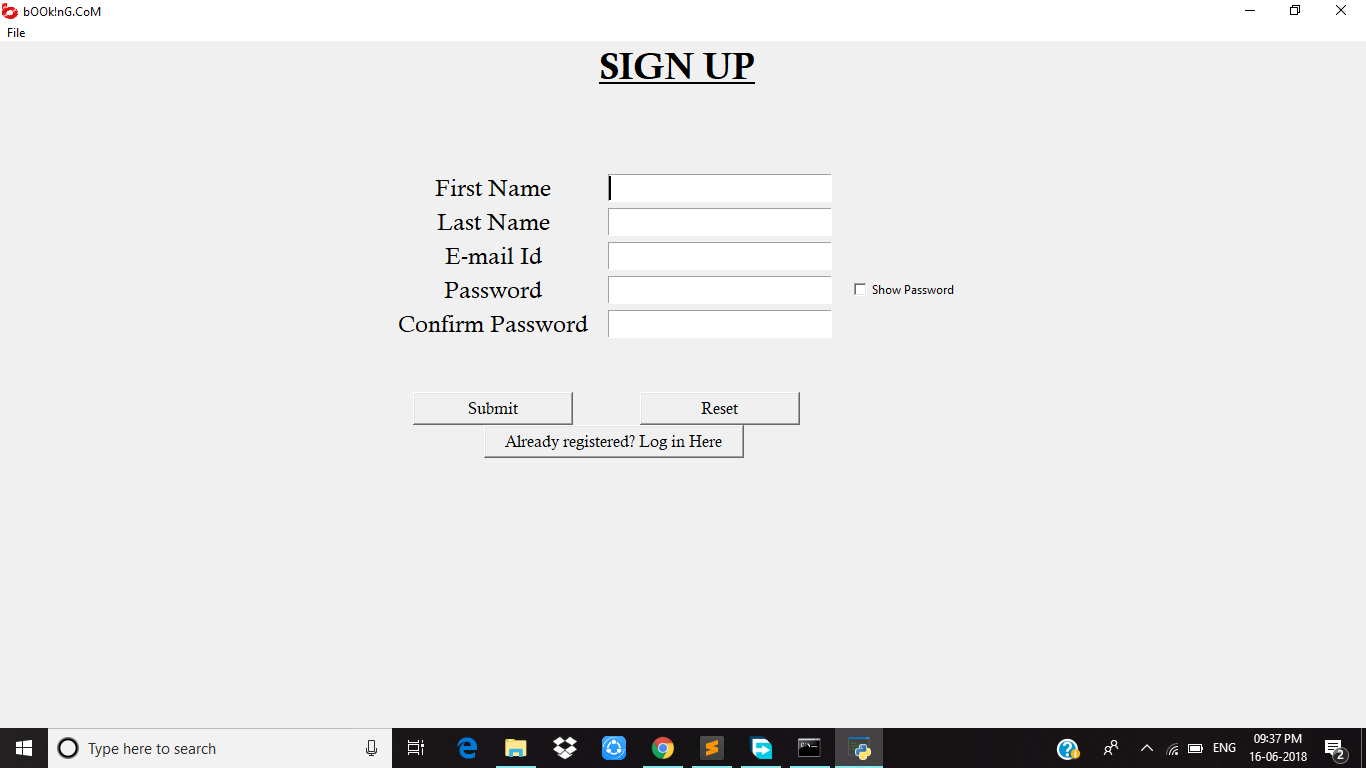
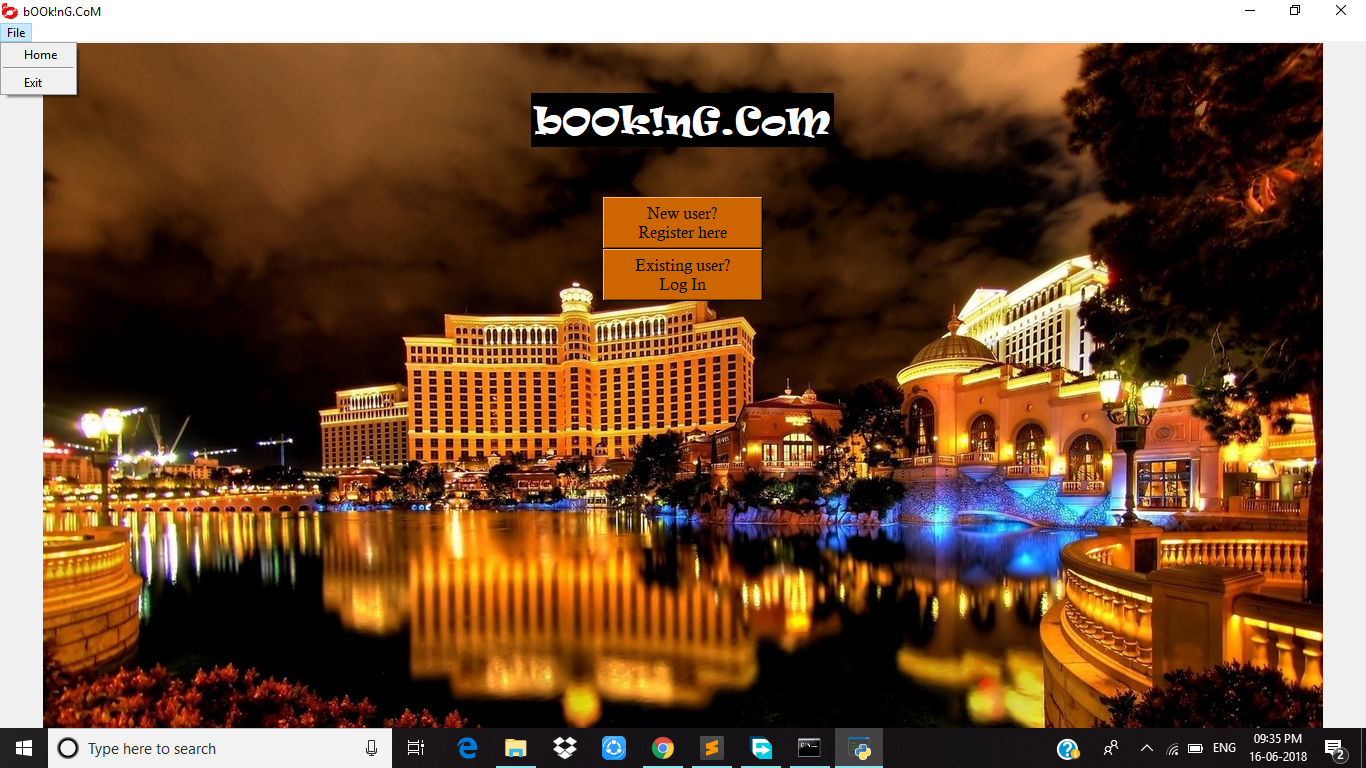
GUI-Booking.com



**CODE:**

import tkinter as tk

from tkinter import ttk

from tkinter import Entry, Toplevel, Canvas

from tkinter import Frame, Entry, Radiobutton, Checkbutton, Text, Listbox, Tk, Label, StringVar,Menu,IntVar

# import tkMessageBox as messagebox

from PIL import Image, ImageDraw, ImageTk, ImageFont

import json

import pickle

class bookingApp(Tk):

def \_\_init\_\_(self,\*args,\*\*kwargs):

Tk.\_\_init\_\_(self,\*args,\*\*kwargs)

Tk.wm\_title(self,"bOOk!nG.CoM")

Tk.iconbitmap(self,"logo.ico")

container = Frame(self)

container.pack(side="top",fill="both",expand = True)

container.grid\_rowconfigure(0,weight=1)

container.grid\_columnconfigure(0,weight=1)

menubar = Menu(container)

filemenu = Menu(menubar,tearoff=0)

filemenu.add\_command(label="Home", command = lambda: self.show\_frame(StartPage))

filemenu.add\_separator()

filemenu.add\_command(label="Exit", command=quit)

menubar.add\_cascade(label="File", menu=filemenu)

Tk.config(self, menu=menubar)

self.frames={}

for F in (StartPage,RegisterPage,LoginPage,WelcomePage):

frame = F(container,self)

self.frames[F] = frame

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

self.show\_frame(StartPage)

def show\_frame(self,cont):

frame = self.frames[cont]

frame.tkraise()

class StartPage(tk.Frame):

def \_\_init\_\_(self,parent,controller):

tk.Frame.\_\_init\_\_(self,parent)

image = Image.open("bg2.jpg")

background\_image=ImageTk.PhotoImage(image)

background\_label = Label(self, image=background\_image)

background\_label.place(x=0, y=0, relwidth=1, relheight=1)

background\_label.image = background\_image

label1 = Label(self,text="bOOk!nG.CoM",font=("Ravie",26),bg="black",fg="white")

label1.pack(padx=10,pady=50)

text1 = "New user?\nRegister here"

button1 = tk.Button(self, text=text1,font=("Calisto MT",12),command = lambda: controller.show\_frame(RegisterPage),activebackground="white",

relief=tk.RAISED,width = 15,bg="darkorange3")

button1.pack()

text2 = "Existing user?\nLog In"

button2 = tk.Button(self, text=text2,font=("Calisto MT",12), command=lambda: controller.show\_frame(LoginPage), activebackground="white",

relief=tk.RAISED,width = 15,bg="darkorange3")

button2.pack()

class RegisterPage(tk.Frame):

def \_\_init\_\_(self,parent,controller):

Frame.\_\_init\_\_(self,parent)

rows=0

while rows<50:

# self.grid\_rowconfigure(rows,weight=6)

self.grid\_columnconfigure(rows,weight=1)

rows+=1

label = Label(self,text="SIGN UP",font=("Calisto MT",28,"bold","underline"),bg="gray94",fg="black")

label.grid(row=0,column=23,columnspan=3,rowspan=3)

Label(self,text="\n\n\n\n").grid(row=3)

label1 = Label(self, text="First Name",font=("Calisto MT",18))

label1.grid(row=6,column=23)

label2 = Label(self, text="Last Name",font=("Calisto MT",18))

label2.grid(row=7,column=23)

Label(self, text="E-mail Id",font=("Calisto MT",18)).grid(row=8,column=23)

Label(self, text="Password",font=("Calisto MT",18)).grid(row=9,column=23)

Label(self, text="Confirm Password",font=("Calisto MT",18)).grid(row=10,column=23)

e1 = Entry(self,font=("Calisto MT",16))

e2 = Entry(self,font=("Calisto MT",16))

e1.grid(row=6, column=24)

e2.grid(row=7, column=24)

e3 = Entry(self,font=("Calisto MT",16))

e3.grid(row=8, column=24)

ep1= Entry(self,font=("Calisto MT",16),show="\*")

ep1.grid(row=9, column=24)

ep2=Entry(self,font=("Calisto MT",16),show="\*")

ep2.grid(row=10, column=24)

dict\_ent = {"FirstName":e1,"LastName":e2,"E-mail":e3,"Password":ep1,"ConfPassword":ep2}

var1 = IntVar()

cb1 = Checkbutton(self, text="Show Password",variable=var1,command=lambda:cb\_change(var1,ep1))

cb1.grid(row=9, column=25)

Label(self,text="\n\n").grid(row=11)

button1 = tk.Button(self, text="Submit",font=("Calisto MT",12),command = lambda: self.reg\_status(controller,dict\_ent),

activebackground="light cyan",relief=tk.RAISED,width = 15)

button1.grid(row=13,column=23)

button2 = tk.Button(self, text="Reset",font=("Calisto MT",12),command = lambda: reset\_field([e1,e2,e3,ep1,ep2]),activebackground="light cyan",

relief=tk.RAISED,width = 15)

button2.grid(row=13,column=24)

button2 = tk.Button(self, text="Already registered? Log in Here",font=("Calisto MT",12),command = lambda: [controller.show\_frame(LoginPage),reset\_field([e1,e2,e3,ep1,ep2])],activebackground="light cyan",

relief=tk.RAISED,width = 25)

button2.grid(row=14,column=23,columnspan=2)

def reg\_status(self,controller,dict\_ent):

dict\_out = {}

jsonObj = []

dict\_out = collect\_ent(dict\_ent)

for key in dict\_out:

if dict\_out[key] =="":

popupmsg("Fill all the fields")

return

varE = self.exist\_email(dict\_out)

if(varE==1):

popupmsg("E-mail Id already exists!!")

return

if(len(dict\_out["Password"])<8):

popupmsg("Password too short!")

return

if(dict\_out["Password"]!=dict\_out["ConfPassword"]):

popupmsg("Passwords don't match!!")

return

else:

jsonObj.append(dict\_out)

fw = open("database.txt","a")

fw.write(json.dumps(jsonObj))

fw.write("\n")

fw.close()

controller.show\_frame(LoginPage)

ent\_lst=[]

for key in dict\_ent:

ent\_lst.append(dict\_ent[key])

reset\_field(ent\_lst)

# fr.close()

def exist\_email(self,dict\_):

lst = read\_filejson()

for dict1 in lst:

if dict1['E-mail']==dict\_['E-mail']:

return 1

return 0

class LoginPage(tk.Frame):

def \_\_init\_\_(self,parent,controller):

Frame.\_\_init\_\_(self,parent)

rows=0

while rows<21:

# self.grid\_rowconfigure(rows,weight=6)

self.grid\_columnconfigure(rows,weight=1)

rows+=1

label = Label(self,text="SIGN IN",font=("Calisto MT",28,"bold","underline"),bg="gray94",fg="black")

label.grid(row=0,column=10,columnspan=2,rowspan=3)

Label(self,text="\n\n\n\n").grid(row=3)

label1 = Label(self, text="E-mail",font=("Calisto MT",18))

label1.grid(row=6,column=10)

label2 = Label(self, text="Password",font=("Calisto MT",18))

label2.grid(row=7,column=10)

e1 = Entry(self,font=("Calisto MT",16))

e2 = Entry(self,font=("Calisto MT",16),show="\*")

e1.grid(row=6, column=11)

e2.grid(row=7, column=11)

ldict\_ent = {"E-mail":e1,"Password":e2}

var1 = IntVar()

cb1 = Checkbutton(self, text="Show Password",variable=var1,command=lambda:cb\_change(var1,e2))

cb1.grid(row=7, column=12)

Label(self,text="\n\n").grid(row=8)

button1 = tk.Button(self, text="Login",font=("Calisto MT",12),command =lambda: self.log\_status(controller,ldict\_ent),activebackground="light cyan",

relief=tk.RAISED,width = 15)

button1.grid(row=10,column=11)

button2 = tk.Button(self, text="New User? Register",font=("Calisto MT",12),command = lambda: controller.show\_frame(RegisterPage),activebackground="light cyan",

relief=tk.RAISED,width = 15)

button2.grid(row=10,column=10)

def log\_status(self,controller,ent):

dict\_out = {}

dict\_out = collect\_ent(ent)

# print(dict\_out)

for key in dict\_out:

if dict\_out[key] =="":

popupmsg("Incomplete Fields!!")

return

varE = self.match\_det(dict\_out)

if(varE==0):

popupmsg("Invalid Email or Password!!")

return

else:

controller.show\_frame(WelcomePage)

ent\_lst=[]

for key in ent:

ent\_lst.append(ent[key])

reset\_field(ent\_lst)

def match\_det(self,dict\_):

lst = read\_filejson()

# print(dict\_)

for dict1 in lst:

# print(dict1)

if dict1['E-mail']==dict\_['E-mail']:

if dict1['Password']==dict\_['Password']:

return 1

return 0

class WelcomePage(tk.Frame):

def \_\_init\_\_(self,parent,controller):

Frame.\_\_init\_\_(self,parent)

rows=0

image = Image.open("bg3.gif")

background\_image=ImageTk.PhotoImage(image)

background\_label = Label(self, image=background\_image)

background\_label.place(x=0, y=0, relwidth=1, relheight=1)

background\_label.image = background\_image

rows=0

while rows<10:

# self.grid\_rowconfigure(rows,weight=6)

self.grid\_columnconfigure(rows,weight=1)

rows+=1

button1 = ttk.Button(self, text="Logout",command = lambda: controller.show\_frame(StartPage))

button1.grid(row=0,column=9)

def quit():

app.destroy()

def reset\_field(lst):

for ent in lst:

ent.delete(0,tk.END)

def popupmsg(msg):

popup = Tk()

def leavemini():

popup.destroy()

popup.wm\_title("!")

popup.iconbitmap("logo.ico")

label = Label(popup,text=msg,font=("Verdana",12,"bold"))

label.pack(side="top",fill="x",pady=10)

button1 = ttk.Button(popup,text="Okay",command=leavemini)

button1.pack()

popup.mainloop()

def cb\_change(var1,ep1):

if(var1.get()):

ep1.config(show="")

else:

ep1.config(show="\*")

def collect\_ent(dict):

dict\_temp = {}

for key in dict:

dict\_temp[key] = dict[key].get()

return dict\_temp

def read\_filejson():

lst = []

fr = open("database.txt","r")

for line in fr:

data = json.loads(line)

# dict1 = data[0]

# print(data[0])

lst.append(data[0])

fr.close()

return lst

app=bookingApp()

app.geometry("1280x720")

app.mainloop()