**HABIT TRACKER**

*A*

*Mini Project Report*

*Submitted in partial fulfilment of the*

*Requirements for the award of the Degree of*

**BACHELOR OF ENGINEERING**

IN

**INFORMATION TECHNOLOGY**

By

**BHARGAV JALAGAM - 1602-20-737-068**

**AKASH AAKI - 1602-20-737-105**

**NANDANI V - 1602-20-737-118**

****

**Department of Information Technology**

**Vasavi College of Engineering (Autonomous)**

**(Affiliated to Osmania University)**

**Ibrahimbagh, Hyderabad-31**

**2022**

**Vasavi College of Engineering (Autonomous)**

**(Affiliated to Osmania University)**

**Hyderabad-500 031**

**Department of Information Technology**

****

**DECLARATION BY THE CANDIDATE**

We, **NANDANI V, JALAGAM BHARGAV** and **AKASH A,** bearing hall ticket numbers, **1602-20-737-118, 1602-20-737-068** and **1602-20-737-105**, hereby declare that the project report entitled **“HABIT TRACKER“** is submitted in partial fulfilment of the requirement for the award of the degree of **Bachelor of Engineering** in **Information Technology**

This is a record of bonafide work carried out by us and the results embodied in this project report have not been submitted to any other university or institute for the award of any other degree or diploma.

**JALAGAM BHARGAV**

**1602-20-737-068**

**AKASH AAKI**

**1602-20-737-105**

**NANDANI V**

**1602-20-737-118**

(Faculty In-Charge) (Head, Dept of IT)

**ACKNOWLEDGEMENT**

The enduring pages of the work are the cumulative sequence of extensive guidance and arduous work. We wish to acknowledge and express our personal gratitude to all those without whom this work could not have been reality.

We feel very delighted to get this rare opportunity to show our profound senses of reverences and indebtedness to our esteemed teacher **DRL PRASANNA** ma’am and **RADHA GADIGE** ma’am, for their keen and sustained interest, valuable advice, throughout the course of which led our mini project, to a successful completion. For this kind act of consideration, we are beholder to them in special manner and no one can fully convey our feelings of respect and regard for them.

**ABSTRACT**

This is a Habit Tracker with Graphical User Interface. This can be used by anyone who would like to improve their life. The user must register himself and verify himself with his email account. The user should login with the registered account. Multiple users can be registered. The user will be provided with many habits, he should tick the habits once performed. The user’s streak on the habit will be incremented by 1 if the habit is done, which will motivate the user and is a reward for completing the habit. The steaks can be viewed by clicking a button present on the screen, which will reveal the steaks for all the habits.

This project is developed to provide fun while doing habits. We came across this project looking at various advertisements during the Indian premier league. For doing this project, we gained some basic knowledge about GUI programming in python. The package tkinter is used in this project, it will provide the required GUI.

**TABLE OF CONTENTS**

| S.No | Topic | Page No. |
| --- | --- | --- |
| 1. | Introduction | 6 |
| 2. | Technology | 7 |
| 3. | Design | 8 |
| 4. | Implementation | 16 |
| 5. | Testing | 38 |
| 6. | Additional Knowledge Acquired | 47 |
| 7. | Conclusion and Future work | 48 |
| 8. | References | 48 |

**INTRODUCTION**

If you want to stick with a habit for good, one simple and effective thing you can do is keep a habit tracker. Habit Tracker is as it says, a Habit Tracker. A habit tracker reminds you to act, it motivates you to continue, and provides immediate satisfaction. Habit tracker can help kickstart a habit or keep you on track with behaviors that you tend to forget or let slide when things get busy. You can keep track of your habits. It is a very simple code and can be used to keep track of your habits.

First of all, the user must register/ sign up first. This data is stored in the mysql database. Then, he can login with the id and password set by him earlier, if wrong id or password is entered, error will be shown. After log-in, he has got several habits visible to him, he can tick each one off whenever the habit is performed. He can check the streaks for all the habits with a click of the button available on the screen.

Even if the user logs-out of the Habit Tracker, the streaks will be stored safely in the mysql database. Once the user logs back in, the habits will be visible again and can be ticked off and increase the streak.

The user’s login id and password will be stored inside the mysql database, which can be accessed by the admin and can remove users.

**TECHNOLOGY**

**Hardware Requirements**

* 512 MB RAM
* 2GB HDD
* CORE i5

**Software Requirements**

* Windows XP/ Windows 2000
* PYTHON INTERPRETER
* VSCODE

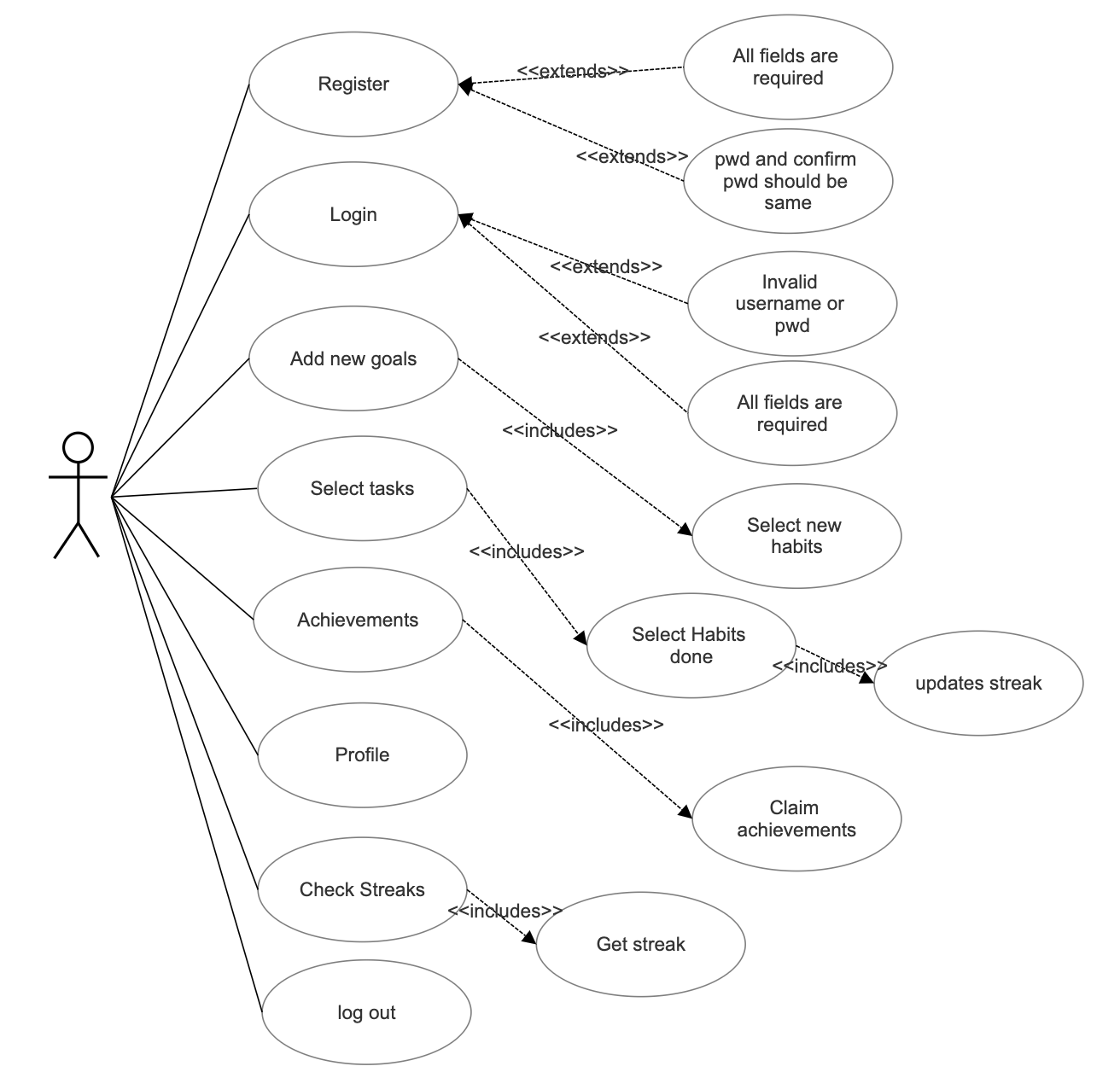
**Packages used**

* tkinter
* PIL
* msilib.schema

**PROPOSED WORK**

**DESIGN –**

USE CASE DIAGRAM –



**Use-Case Descriptions:**

**Use Case ID: UC01**

Name: Register

Actors: User

Description: Allows new user to register for an account

Pre-conditions: None

Post-conditions: An account is created for the user

Main flow:

| **user** | **system** |
| --- | --- |
| 1.Chooses the register option |  |
|  | 2.Prompts the data required for registration |
| 3.Enters the data prompted by the system |  |
|  | 4.Validates user information and creates account if its correct |

**Use Case ID: UC02**

Name: Login

Actors: User

Description: Allows registered users to Login

Pre-conditions: user should be registered with the system

Post-conditions: User logs in and the home screen is displayed on the screen

Main Flow:

| **User** | **System** |
| --- | --- |
| Enters username and pw and chooses login option |  |
|  | Validates credentials:  Prompts error if incorrect  Displays the home page if correct |

**Use Case ID: UC03**

Name: Add new goals

Actors: User

Description: Allows registered users to select and add habits they like.

Pre-conditions: User should be successfully signed-in to the system.

Post-conditions: Habit is added

Main Flow:

| **user** | **system** |
| --- | --- |
| 1.Chooses Add new goals |  |
|  | 2.Displays habits |
| 3.Selects a habit he likes |  |
|  | 4. Habit gets added to list of Habits |

**Use Case ID: UC04**

Name: Select tasks

Actors: User

Description: The user is displayed with the habits and he has to choose the habit he has performed.

Pre-conditions: User should be successfully signed-in to the system.

Post-conditions: Streaks updated

Main Flow:

| **user** | **system** |
| --- | --- |
| 1.Chooses Select Tasks |  |
|  | 2.Displays habits |
| 3.Selects which habit performed |  |
|  | 4. Streaks updated for that Habit |

**Use Case ID: UC05**

Name: Achievements

Actors: User

Description: An achievement is made if the user has more than 15 streaks on any habit, once the user clicks the claim achievements, he will be given a crown for each habit that has more than 15.

Pre-conditions: User should be successfully signed-in to the system.

Post-conditions: Once the user clicks claim achievements, his profile will be updated with crowns he has received.

Main Flow:

| **User** | **System** |
| --- | --- |
| 1.Chooses See Achievements and chooses claim achievements |  |
|  | 2. The system will award the user with crowns  (one crown for each habit with 15+ streaks) |

**Use Case ID: UC06**

Name: Profile

Actors: User

Description: Shows the achievements of the user.

Pre-conditions: User should be successfully signed-in to the system.

Post-conditions: Achievements displayed.

Main Flow:

| **User** | **System** |
| --- | --- |
| 1.Chooses Profile |  |
|  | 2. The system will display the crowns |

**Use Case ID: UC07**

Name: Check Streaks

Actors: User

Description: When clicked on this, it displays the Streaks of all the habits.

Pre-conditions: User should be successfully signed-in to the system.

Post-conditions: Displays the streaks of all habits.

Main Flow:

| **User** | **System** |
| --- | --- |
| 1.Chooses Check Streaks |  |
|  | 2. The system will display the streaks of all habits of user |

**Use Case ID: UC08**

Name: Log out

Actors: User

Description: logs-out

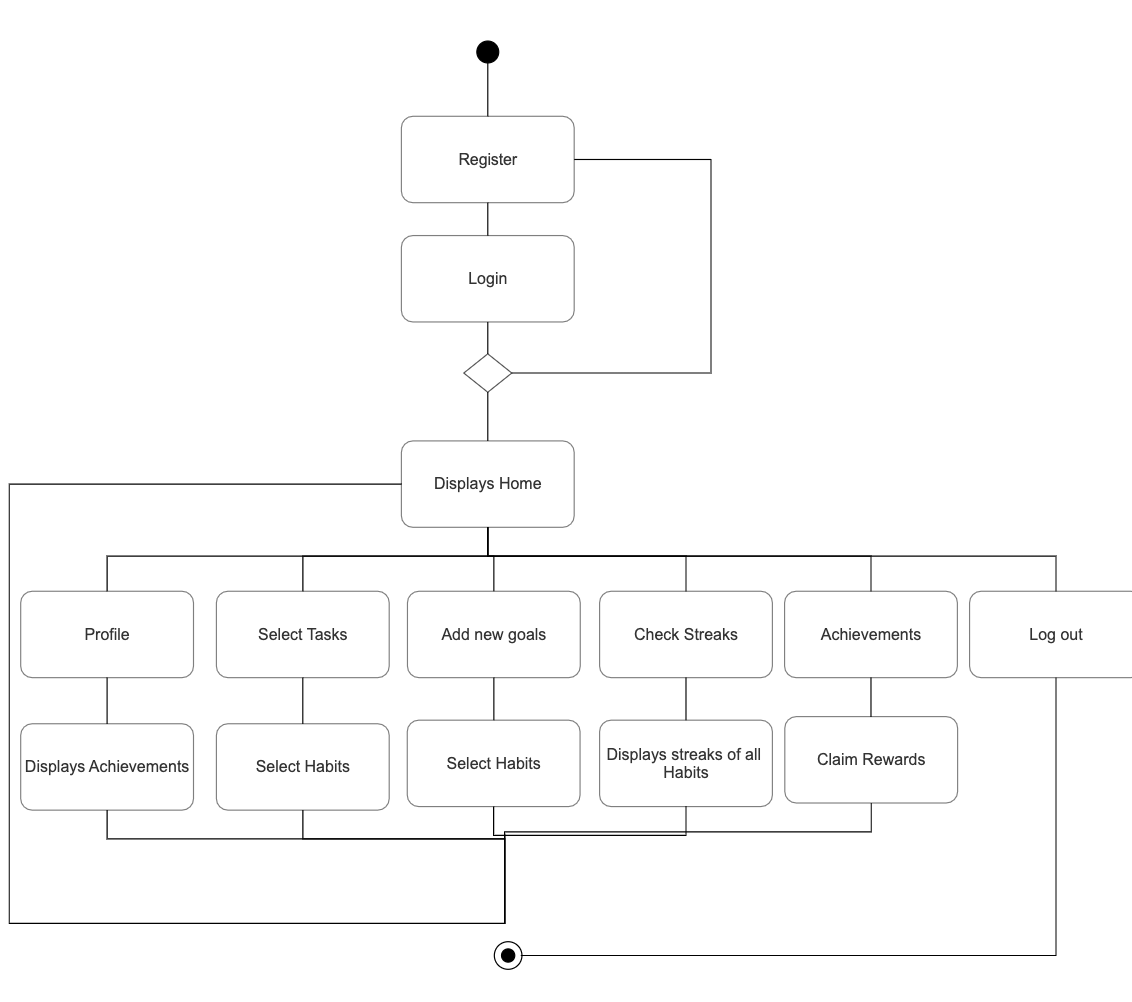
Pre-conditions: User should be successfully signed-in to the system.

Post-conditions: logs out.

Main Flow:

| **User** | **System** |
| --- | --- |
| 1.Chooses log out |  |
|  | 2. logs-out and end of program execution. |

**ACTIVITY DIAGRAM**

****

**IMPLEMENTATION**

**Source code –**

from tkinter import \*

from tkinter import ttk

from PIL import ImageTk,Image

from tkinter import messagebox,filedialog

import mysql.connector

import tkinter as tk

class Login:

def \_\_init\_\_(self,root) :

self.root=root

self.root.title("Habit Tracker")

self.root.geometry("1366x700+0+0")

self.root.resizable(False,False)

self.loginform()

def loginform(self):

Frame\_login= Frame(self.root,bg="white")

Frame\_login.place(x=0,y=0,height=700,width=1366)

self.img= ImageTk.PhotoImage(file="habitt.jpeg")

img=Label(Frame\_login,image=self.img).place(x=0,y=0,width=1366,height=700)

frame\_input=Frame(self.root,bg="white")

frame\_input.place(x=220,y=130,height=450,width=350)

label1=Label(frame\_input,text="Login here",font=('impact',32,'bold'),fg="black",bg="white")

label1.place(x=75,y=20)

label2=Label(frame\_input,text="Username",font=('Goudy old style',32,'bold'),fg="orangered",bg="white")

label2.place(x=30,y=95)

self.email\_txt=Entry(frame\_input,font=('times new roman',15,'bold'),bg="lightgray")

self.email\_txt.place(x=30,y=145,height=35,width=270)

label3=Label(frame\_input,text="Password",font=('Goudy old style',20,'bold'),fg="orangered",bg="white")

label3.place(x=30,y=195)

self.password=Entry(frame\_input,font=('times new roman',15,'bold'),bg="lightgray")

self.password.place(x=30,y=245,height=35,width=270)

#btn1= Button(frame\_input,text="forgot password?",cursor='hand2',font=('calibri',10),bg="white",fg="black",bd=0)

# btn1.place(x=125,y=305)

btn2= Button(frame\_input,text="Login",command=self.login,cursor='hand2',font=('times new roman',15),bg="white",fg="orangered",bd=0,width=15,height=1)

btn2.place(x=90,y=340)

btn3= Button(frame\_input,text="Not Registered? Register",command=self.Register,cursor='hand2',font=('calibri',10),bg="white",fg="black",bd=0)

btn3.place(x=110,y=390)

def login(self):

if self.email\_txt.get()=="" or self.password.get()=="":

messagebox.showerror("Error","All feilds are required",parent=self.root)

else:

try:

con= mysql.connector.connect(host="localhost",user="Nandani",passwd="MiniProjecthehe1.",database="pythongui")

cur=con.cursor(buffered=True)

cur.execute('select \* from register where username=%s and password=%s',(self.email\_txt.get(),self.password.get()))

row=cur.fetchone()

if row==None:

messagebox.showerror('Error','Invalid username and password',parent=self.root)

self.loginclear()

self.email\_txt.focus()

else:

self.appscreen()

con.close()

except Exception as es:

messagebox.showerror('Error',f'Error Due to: {str(es)}',parent=self.root)

def Register(self):

Frame\_login1=Frame(self.root,bg="white")

Frame\_login1.place(x=0,y=0,height=700,width=1366)

frame\_input2=Frame(self.root,bg="white")

frame\_input2.place(x=320,y=130,height=450,width=630)

label1=Label(frame\_input2,text="Register here",font=('impact',32,'bold'),fg="black",bg="white")

label1.place(x=45,y=20)

label2=Label(frame\_input2,text="Username",font=('Goudy old style',20,'bold'),fg="orangered",bg="white")

label2.place(x=30,y=95)

self.entry=Entry(frame\_input2,font=('times new roman',15,'bold'),bg="lightgray")

self.entry.place(x=30,y=145,height=35,width=270)

label3=Label(frame\_input2,text="Password",font=('Goudy old style',20,'bold'),fg="orangered",bg="white")

label3.place(x=30,y=195)

self.entry2=Entry(frame\_input2,font=('times new roman',15,'bold'),bg="lightgray")

self.entry2.place(x=30,y=245,height=35,width=270)

label4=Label(frame\_input2,text="Email id",font=('Goudy old style',20,'bold'),fg="orangered",bg="white")

label4.place(x=330,y=95)

self.entry3=Entry(frame\_input2,font=('times new roman',15,'bold'),bg="lightgray")

self.entry3.place(x=330,y=145,height=35,width=270)

label5=Label(frame\_input2,text="confirm Password",font=('Gody old style',20,'bold'),fg="orangered",bg="white")

label5.place(x=330,y=195)

self.entry4=Entry(frame\_input2,font=('times new roman',15,'bold'),bg="lightgray")

self.entry4.place(x=330,y=245,height=35,width=270)

btn2= Button(frame\_input2,text="Register",command=self.register,cursor='hand2',font=('times new roman',15),bg="white",fg="orangered",bd=0,width=15,height=1)

btn2.place(x=90,y=340)

btn3= Button(frame\_input2,text="Already Registered?login",command=self.loginform,cursor='hand2',font=('calibri',10),bg="white",fg="black",bd=0)

btn3.place(x=110,y=390)

def register(self):

if self.entry.get()==""or self.entry2.get()==""or self.entry3.get()==""or self.entry4.get()=="":

messagebox.showerror("Error","All feilds are required",parent=self.root)

elif self.entry2.get()!=self.entry4.get():

messagebox.showerror("Error","Password and confirm password should be same",parent=self.root)

else:

#try:

con= mysql.connector.connect(host="localhost",user="Nandani",passwd="MiniProjecthehe1.",database="pythongui")

cur=con.cursor(buffered=True)

#cur.execute('select \* from register where emailid=%s ',(self.entry3.get()))

#row=cur.fetchall()

#if row!=None:

#messagebox.showerror("Error","user already exists,please try with another email id",parent=self.root)

#self.regclear()

#self.entry.focus()

#else:

#string= self.entry.get()

cur.execute('INSERT INTO register VALUES("'+str(self.entry.get())+'","'+str(self.entry3.get())+'","'+str(self.entry2.get())+'","'+str(self.entry4.get())+'","'+str(0)+'","'+str(0)+'");')

con.commit()

con.close()

# messagebox.showinfo("register successful",parent=self.root)

messagebox.showinfo("Yayy!","You are successfully registered",parent=self.root)

self.regclear()

#except Exception as es:

#messagebox.showerror("error",f"error due to:{str(es)}",parent=self.root)

def isAdded(self):

print(1)

username=self.email\_txt.get()

print(username)

habit1="Swimming"

print(self.CheckVar3.get())

if (self.CheckVar3.get() == 1):

print(1)

con= mysql.connector.connect(host="localhost",user="Nandani",passwd="MiniProjecthehe1.",database="pythongui")

cur=con.cursor(buffered=True)

cur.execute("CREATE TABLE IF NOT EXISTS "+str(username)+" (habits varchar(20) not null, streaks int(5) DEFAULT '0');")

cur.execute('select habits from nandani where habits="Swimming"')

#cur.execute("select habits from "+str(username)+" where habits= "+"'Swimming'")

row=cur.fetchone()

if row==None:

#cur.execute('INSERT INTO nandani VALUES(')

cur.execute('INSERT INTO nandani VALUES ("' +str(habit1)+ '",'+str('0')+ ',"'+str('I007')+'",'+str('NULL')+')')

con.commit()

print("inserted")

else:

print("no")

habit2="Quit smoking"

if self.CheckVar4.get() == 1:

print(1)

con= mysql.connector.connect(host="localhost",user="Nandani",passwd="MiniProjecthehe1.",database="pythongui")

cur=con.cursor(buffered=True)

cur.execute("CREATE TABLE IF NOT EXISTS "+str(username)+" (habits varchar(20) not null, streaks int(5) DEFAULT '0');")

#cur.execute('select habits from nandani where habits="Swimming"')

cur.execute("select habits from "+str(username)+" where habits= "+"'Quit smoking'")

row=cur.fetchone()

if row==None:

cur.execute('INSERT INTO '+str(username)+ ' VALUES ("' +str(habit2)+ '",'+str('0')+ ')')

con.commit()

print("inserted")

habit3="Quit drinking"

if self.CheckVar5.get() == 1:

print(1)

con= mysql.connector.connect(host="localhost",user="Nandani",passwd="MiniProjecthehe1.",database="pythongui")

cur=con.cursor(buffered=True)

cur.execute("CREATE TABLE IF NOT EXISTS "+str(username)+" (habits varchar(20) not null, streaks int(5) DEFAULT '0');")

#cur.execute('select habits from nandani where habits="Swimming"')

cur.execute("select habits from "+str(username)+" where habits= "+"'Quit drinking'")

row=cur.fetchone()

if row==None:

cur.execute('INSERT INTO '+str(username)+ ' VALUES ("' +str(habit3)+ '",'+str('0')+ ')')

con.commit()

print("inserted")

habit4="No fast food"

if self.CheckVar3.get() == 1:

print(1)

con= mysql.connector.connect(host="localhost",user="Nandani",passwd="MiniProjecthehe1.",database="pythongui")

cur=con.cursor(buffered=True)

cur.execute("CREATE TABLE IF NOT EXISTS "+str(username)+" (habits varchar(20) not null, streaks int(5) DEFAULT '0');")

#cur.execute('select habits from nandani where habits="Swimming"')

cur.execute("select habits from "+str(username)+" where habits= "+"'No fast food'")

row=cur.fetchone()

if row==None:

cur.execute('INSERT INTO '+str(username)+ ' VALUES ("' +str(habit4)+ '",'+str('0')+ ')')

con.commit()

print("inserted")

habit5="Breakfast"

if self.CheckVar4.get() == 1:

print(1)

con= mysql.connector.connect(host="localhost",user="Nandani",passwd="MiniProjecthehe1.",database="pythongui")

cur=con.cursor(buffered=True)

cur.execute("CREATE TABLE IF NOT EXISTS "+str(username)+" (habits varchar(20) not null, streaks int(5) DEFAULT '0');")

#cur.execute('select habits from nandani where habits="Swimming"')

cur.execute("select habits from "+str(username)+" where habits= "+"'Breakfast'")

row=cur.fetchone()

if row==None:

cur.execute('INSERT INTO '+str(username)+ ' VALUES ("' +str(habit5)+ '",'+str('0')+ ')')

con.commit()

print("inserted")

habit6="8 hours sleep"

if self.CheckVar5.get() == 1:

print(1)

con= mysql.connector.connect(host="localhost",user="Nandani",passwd="MiniProjecthehe1.",database="pythongui")

cur=con.cursor(buffered=True)

cur.execute("CREATE TABLE IF NOT EXISTS "+str(username)+" (habits varchar(20) not null, streaks int(5) DEFAULT '0');")

#cur.execute('select habits from nandani where habits="Swimming"')

cur.execute("select habits from "+str(username)+" where habits= "+"'8 hours sleep'")

row=cur.fetchone()

if row==None:

cur.execute('INSERT INTO '+str(username)+ ' VALUES ("' +str(habit6)+ '",'+str('0')+ ')')

con.commit()

print("inserted")

def removethis2(self):

self.frame\_inputtt.destroy()

def on\_close(self):

response=messagebox.askyesno('Exit','Are you sure you want to exit?')

if response:

self.New\_Window.destroy()

def newGoals(self):

self.New\_Window = Toplevel()

self.New\_Window.geometry("1366x700+0+0")

#self.New\_Window.protocol('WM\_DELETE\_WINDOW',self.on\_close)

frame\_inputtt=Frame(self.New\_Window,bg="white")

frame\_inputtt.place(x=0,y=0,height=1500,width=2000)

self.CheckVar3 = BooleanVar()

self.CheckVar4 = BooleanVar()

self.CheckVar5 = BooleanVar()

self.CheckVar6 = BooleanVar()

self.CheckVar7 = BooleanVar()

self.CheckVar8 = BooleanVar()

self.CheckVar9 = BooleanVar()

self.CheckVar10 = BooleanVar()

Checkbutton(frame\_inputtt, text="Swimming", variable=self.CheckVar3, onvalue=1, offvalue=0, command=self.isAdded).place(x=40,y=30)

Checkbutton(frame\_inputtt, text="Quit smoking", variable=self.CheckVar4, onvalue=1, offvalue=0, command=self.isAdded).place(x=40,y=50)

Checkbutton(frame\_inputtt, text="Quit drinking", variable=self.CheckVar5, onvalue=1, offvalue=0, command=self.isAdded).place(x=40,y=70)

Checkbutton(frame\_inputtt, text="No fast food", variable=self.CheckVar6, onvalue=1, offvalue=0, command=self.isAdded).place(x=40,y=90)

Checkbutton(frame\_inputtt, text="Breakfast", variable=self.CheckVar7, onvalue=1, offvalue=0, command=self.isAdded).place(x=40,y=110)

Checkbutton(frame\_inputtt, text="8 hours sleep", variable=self.CheckVar8, onvalue=1, offvalue=0, command=self.isAdded).place(x=40,y=130)

Checkbutton(frame\_inputtt, text="Screen time limited", variable=self.CheckVar9, onvalue=1, offvalue=0, command=self.isAdded).place(x=40,y=150)

Checkbutton(frame\_inputtt, text="Sports", variable=self.CheckVar10, onvalue=1, offvalue=0, command=self.isAdded).place(x=40,y=170)

print(self.CheckVar3.get())

#butt = Button(New\_Window, text="Push me2", command=self.removethis2)

#butt.place(x=800,y=450)

def removethis(self):

self.Frame\_loginn.destroy()

def displayStreaks(self):

con= mysql.connector.connect(host="localhost",user="Nandani",passwd="MiniProjecthehe1.",database="pythongui")

cur=con.cursor(buffered=True)

sql='SELECT \* FROM nandani'

#sql=('SELECT YogaStreaks,HomeworkStreaks FROM register where emailid=%s',self.email\_txt.get())

#cur.execute('SELECT YogaStreaks,HomeworkStreaks FROM register where emailid=%s',self.email\_txt.get())

cur.execute(sql)

rows=cur.fetchall()

self.Display\_Window = Toplevel()

self.Display\_Window.geometry("1366x700+0+0")

self.Frame\_loginn=Frame(self.Display\_Window,bg="white")

self.Frame\_loginn.place(x=0,y=0,height=2000,width=1500)

tv=ttk.Treeview(self.Frame\_loginn,columns=(1,2),show="headings",height=7)

tv.pack()

tv.heading(1,text="Habits")

tv.heading(2,text="Streaks")

for i in rows:

tv.insert('','end',values=i)

#butt = Button(self.root, text="Push me", command=self.removethis)

#butt.place(x=1000,y=450)

def togglecheck(self,event):

mydb=mysql.connector.connect(host="localhost",user="Nandani",passwd="MiniProjecthehe1.",database="pythongui")

cursor=mydb.cursor()

rowid=self.trv.identify\_row(event.y)

tag =self.trv.item(rowid,"tags")[0]

tags=list(self.trv.item(rowid,"tags"))

tags.remove(tag)

self.trv.item(rowid,tags=tags)

if tag=="checked":

self.trv.item(rowid,tags="unchecked")

else:

self.trv.item(rowid,tags="checked")

hey=rowid

print(rowid)

print(hey)

#cursor.execute("update nandani set streaks=streaks+1 where habits= "+"'Swimming'")

#if(rowid=='I005'):

#cursor.execute("update nandani set streaks=streaks+1 where rowid= "+"'I005'")

#cursor.execute("update nandani set streaks=streaks+1 where rowid= "+rowid)

#cursor.execute('update nandani set streaks=streaks+1 where rowid=%s ',(str(rowid)))

cursor.execute('update nandani set streaks=streaks+1 where rowid="'+str(rowid)+'"')

#cur.execute('INSERT INTO register VALUES("'+str(self.entry.get())+'","'+str(self.entry3.get())+'","'+str(self.entry2.get())+'","'+str(self.entry4.get())+'","'+str(0)+'","'+str(0)+'");')

mydb.commit()

def update(self):

mydb=mysql.connector.connect(host="localhost",user="Nandani",passwd="MiniProjecthehe1.",database="pythongui")

cursor=mydb.cursor()

self.Update\_Window = Toplevel()

self.Update\_Window.geometry("1366x700+0+0")

#wrapper1=LabelFrame(root,text="Select today's finished tasks")

wrapper1=LabelFrame(self.Update\_Window,text="Select today's finished tasks")

wrapper1.pack(fill="both",expand="yes",padx=20,pady=10)

im\_checked= ImageTk.PhotoImage(Image.open("checked1.png"))

im\_unchecked= ImageTk.PhotoImage(Image.open("unchecked1.png"))

#im\_checked= ImageTk.PhotoImage(file="checked.jpg")

#im\_unchecked= ImageTk.PhotoImage(file="unchecked.jpg")

self.trv=ttk.Treeview(wrapper1,columns=(1,2))

style=ttk.Style(self.trv)

style.configure('Treeview',rowheight=30)

self.trv.tag\_configure('checked',image=im\_checked)

self.trv.tag\_configure('unchecked',image=im\_unchecked)

self.trv.pack()

self.trv.heading('#0',text="")

self.trv.heading('#1',text="habits")

self.trv.heading('#2',text="streaks")

#trv.heading('#3',text="last name")

#trv.heading('#4',text="age")

#trv.bind('<Double 1>',getrow)

self.trv.bind('<Button 1>',self.togglecheck)

query="SELECT \* FROM nandani"

cursor.execute(query)

rows=cursor.fetchall()

for i in rows:

self.trv.insert('','end',values=i,tags="unchecked")

def profile(self):

self.Profile\_Window = Toplevel()

self.Profile\_Window.geometry("1366x700+0+0")

#self.Profile\_Window.protocol('WM\_DELETE\_WINDOW',self.on\_close2)

mydb=mysql.connector.connect(host="localhost",user="Nandani",passwd="MiniProjecthehe1.",database="pythongui")

cursor=mydb.cursor()

Frame\_Profile= Frame(self.Profile\_Window,bg="white")

Frame\_Profile.place(x=0,y=0,height=1500,width=2000)

usernameDisplay=Label(Frame\_Profile,text="Nandani",font=('times new roman','22','bold'),fg="orangered",bg="white")

usernameDisplay.place(x=200,y=40)

achievementsDisplay=Label(Frame\_Profile,text="Achievements",font=('times new roman','22','bold'),fg="black",bg="white")

achievementsDisplay.place(x=200,y=200)

cursor.execute('select crown from nandani where crown is not null')

rows=cursor.fetchall()

j=0

for i in rows:

j=j+1

print(j)

if(j==1):

self.img= ImageTk.PhotoImage(file="crown1.jpg")

img=Label(Frame\_Profile,image=self.img).place(x=300,y=300,width=120,height=120)

if(j==2):

self.img= ImageTk.PhotoImage(file="2crowns.png")

img=Label(Frame\_Profile,image=self.img).place(x=300,y=300,width=200,height=120)

if(j==3):

self.img= ImageTk.PhotoImage(file="3crowns.png")

img=Label(Frame\_Profile,image=self.img).place(x=300,y=300,width=300,height=120)

def claimcrown(self):

mydb=mysql.connector.connect(host="localhost",user="Nandani",passwd="MiniProjecthehe1.",database="pythongui")

cursor=mydb.cursor()

cursor.execute('UPDATE register SET Crowns="'+str(self.k)+'" WHERE username="Nandani"')

#cur.execute("select habits from "+str(username)+" where habits= "+"'8 hours sleep'")

mydb.commit()

#cursor.execute('update register set Crowns= where rowid="'+str(rowid)+'"')

def achievements(self):

self.New\_Windoww = Toplevel()

self.New\_Windoww.geometry("1366x700+0+0")

#self.New\_Windoww.protocol('WM\_DELETE\_WINDOW',self.on\_close2)

mydb=mysql.connector.connect(host="localhost",user="Nandani",passwd="MiniProjecthehe1.",database="pythongui")

cursor=mydb.cursor()

Frame\_login= Frame(self.New\_Windoww,bg="white")

Frame\_login.place(x=0,y=0,height=1500,width=2000)

self.img= ImageTk.PhotoImage(file="crown1.jpg")

img=Label(Frame\_login,image=self.img).place(x=0,y=0,width=120,height=120)

crownlabel=Label(Frame\_login,text="REGAL\n Earn 1 crown per habit if streaks greater than 15",font=('times new roman','22','bold'),fg="black",bg="white")

crownlabel.place(x=200,y=40)

cursor.execute('SELECT streaks FROM nandani WHERE streaks>7 AND crown=1')

rows=cursor.fetchall()

j=0

for i in rows:

j=j+1

print(j)

cursor.execute('UPDATE nandani SET crown=1 WHERE streaks>7 AND crown is null')

print("yes")

mydb.commit()

cursor.execute('SELECT streaks FROM nandani WHERE streaks>7 AND crown=1')

rows=cursor.fetchall()

self.k=0

for i in rows:

print(i)

self.k=self.k+1

print(self.k)

if(self.k>j):

print(str(self.k-j)+"new crowns")

claimcrown= Button(Frame\_login,text="claim crown",command=self.claimcrown,cursor='hand2',font=('times new roman',15),bg="white",fg="orangered",bd=0,width=15,height=1)

claimcrown.place(x=800,y=40)

self.img2= ImageTk.PhotoImage(file="bronze1.jpg")

img2=Label(Frame\_login,image=self.img2).place(x=0,y=120,width=120,height=120)

bronzelabel=Label(Frame\_login,text="BRONZE LEAGUE\n Enter the Bronze league if 1/4 habits streaks greater than 15",font=('times new roman','22','bold'),fg="black",bg="white")

bronzelabel.place(x=200,y=150)

self.img3= ImageTk.PhotoImage(file="silver1.jpg")

img3=Label(Frame\_login,image=self.img3).place(x=0,y=240,width=120,height=120)

silverlabel=Label(Frame\_login,text="SILVER LEAGUE\n Enter the Silver league if 1/2 habits streaks greater than 15",font=('times new roman','22','bold'),fg="black",bg="white")

silverlabel.place(x=200,y=260)

self.img4= ImageTk.PhotoImage(file="gold1.jpg")

img4=Label(Frame\_login,image=self.img4).place(x=0,y=360,width=120,height=120)

goldlabel=Label(Frame\_login,text="GOLD LEAGUE\n Enter the Gold league if 3/4 habits streaks greater than 15",font=('times new roman','22','bold'),fg="black",bg="white")

goldlabel.place(x=200,y=370)

self.img5= ImageTk.PhotoImage(file="diamond1.jpg")

img5=Label(Frame\_login,image=self.img5).place(x=0,y=480,width=120,height=120)

diamondlabel=Label(Frame\_login,text="DIAMOND LEAGUE\n Enter the Diamond league if all habits streaks greater than 15",font=('times new roman','22','bold'),fg="black",bg="white")

diamondlabel.place(x=200,y=480)

self.img6= ImageTk.PhotoImage(file="fire1.jpg")

img6=Label(Frame\_login,image=self.img6).place(x=0,y=600,width=120,height=120)

firelabel=Label(Frame\_login,text="WILDFIRE\n 1 wildfire per 7 streaks",font=('times new roman','22','bold'),fg="black",bg="white")

firelabel.place(x=400,y=590)

def appscreen(self):

#self.appscreen\_window = Toplevel()

#self.appscreen\_window.geometry("1366x700+0+0")

#Frame\_login=Frame(self.appscreen\_window,bg="white")

Frame\_login=Frame(self.root,bg="white")

Frame\_login.place(x=0,y=0,height=700,width=1366)

self.img= ImageTk.PhotoImage(file="appscreen.jpg")

img=Label(Frame\_login,image=self.img).place(x=0,y=0,width=1366,height=700)

label1=Label(Frame\_login,text="Welcome to Habit Tracker",font=('times new roman',32,'bold'),fg="black",bg="white")

label1.place(x=375,y=100)

label2=Label(Frame\_login,text="Consistency is the key",font=('times new roman',12,'bold'),fg="black",bg="white")

label2.place(x=700,y=160)

#label3=Label(Frame\_login,text="Select the tasks performed",font=('times new roman',25,'bold'),fg="black",bg="white")

#label3.place(x=340,y=220)

btn4= Button(Frame\_login,text="Select tasks",command=self.update,cursor='hand2',font=('calibri',15),bg="white",fg="orangered",bd=0)

btn4.place(x=320,y=550)

btn3= Button(Frame\_login,text="Add new goals",command=self.newGoals,cursor='hand2',font=('times new roman',15),bg="white",fg="orangered",bd=0,width=15,height=1)

btn3.place(x=450,y=550)

btn3= Button(Frame\_login,text="Check Streaks",command=self.displayStreaks,cursor='hand2',font=('times new roman',15),bg="white",fg="orangered",bd=0,width=15,height=1)

btn3.place(x=650,y=550)

btn5= Button(Frame\_login,text="Achievements",command=self.achievements,cursor='hand2',font=('times new roman',15),bg="white",fg="orangered",bd=0,width=15,height=1)

btn5.place(x=850,y=550)

btn2= Button(Frame\_login,text="log out",command=self.loginform,cursor='hand2',font=('times new roman',15),bg="white",fg="orangered",bd=0,width=15,height=1)

btn2.place(x=1050,y=550)

btn2= Button(Frame\_login,text="Profile",command=self.profile,cursor='hand2',font=('times new roman',15),bg="white",fg="orangered",bd=0,width=15,height=1)

btn2.place(x=140,y=550)

def regclear(self):

self.entry.delete(0,END)

self.entry2.delete(0,END)

self.entry3.delete(0,END)

self.entry4.delete(0,END)

def loginclear(self):

self.email\_txt.delete(0,END)

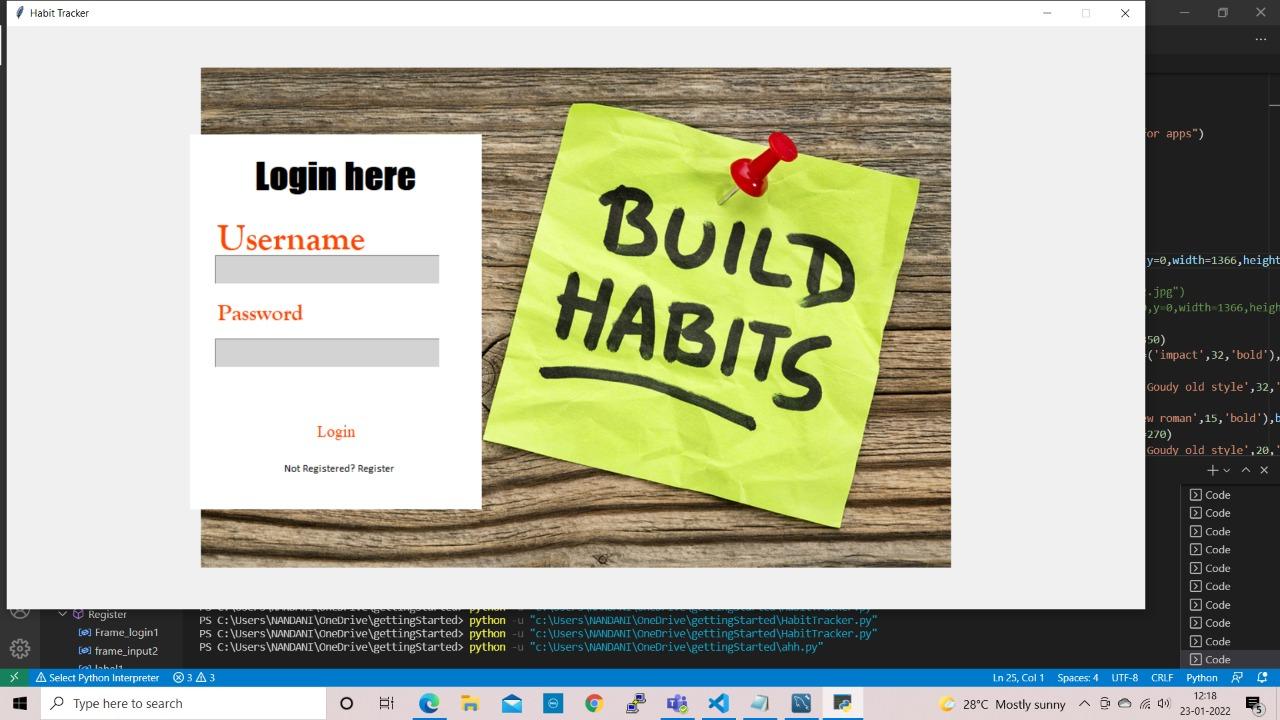
self.password.delete(0,END)

root=Tk()

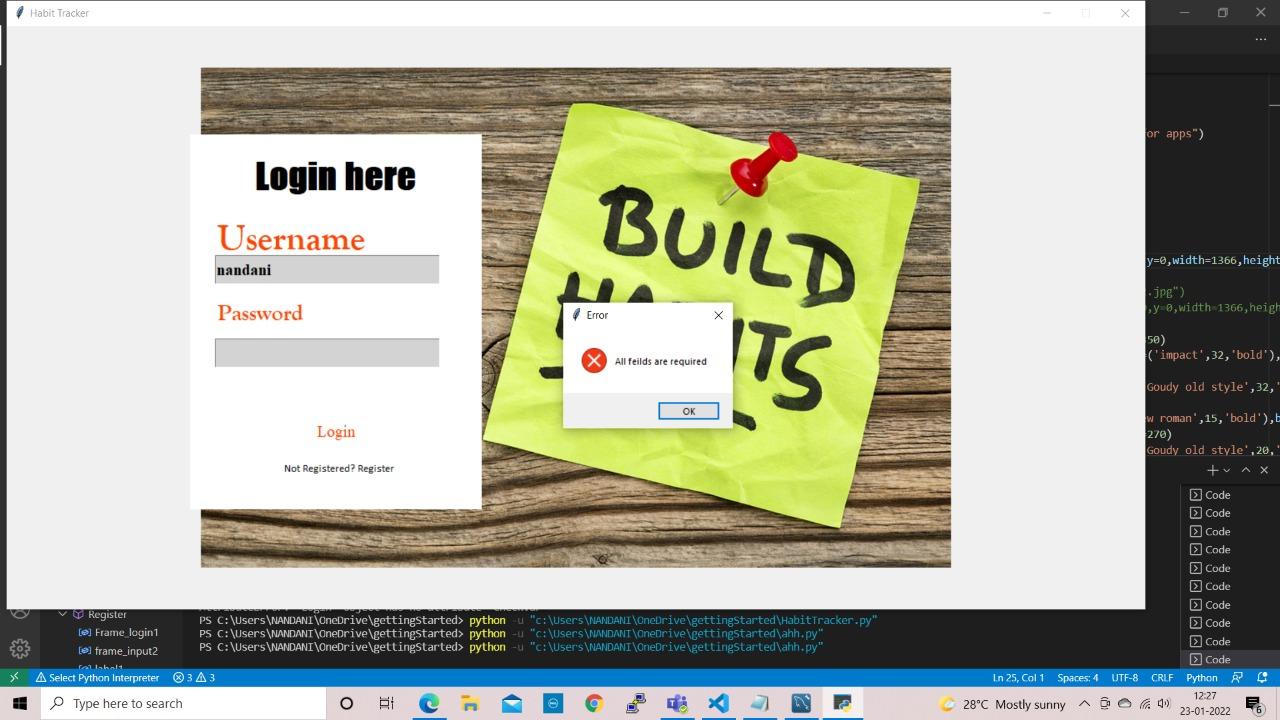
ob=Login(root)

root.mainloop()

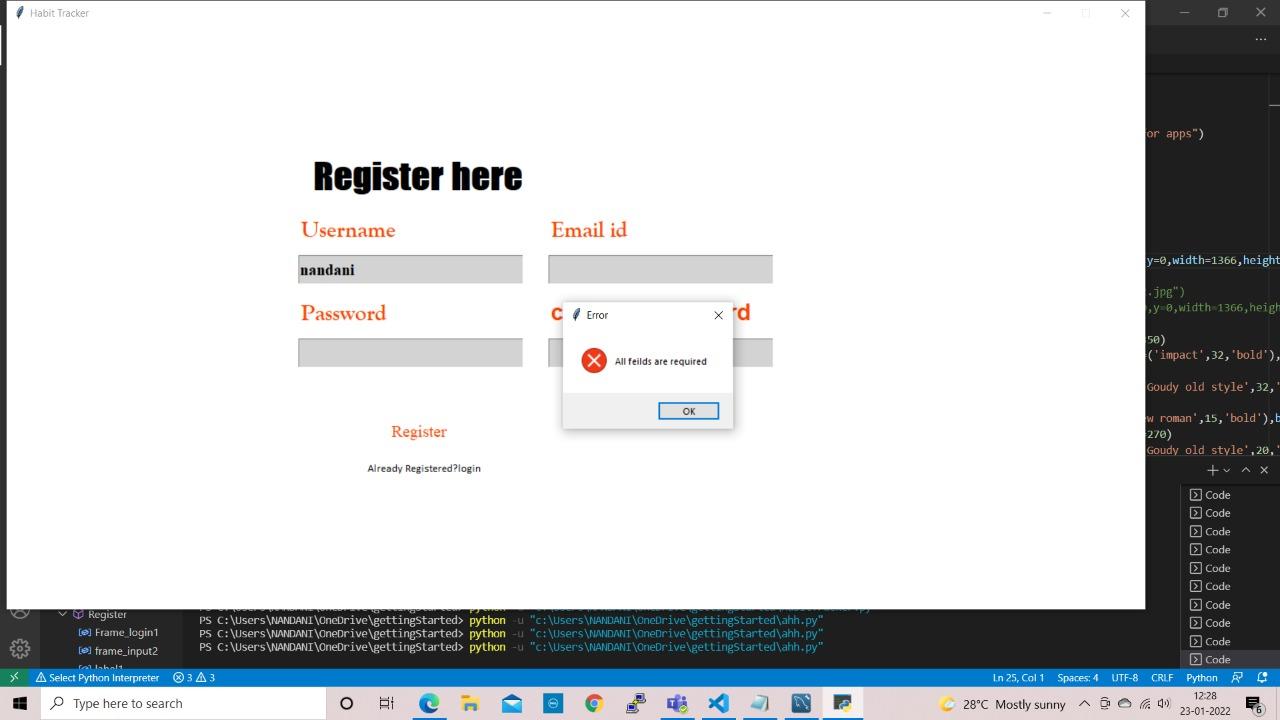
**SCREENSHOTS OF APPLICATION TEST CASES –**

****

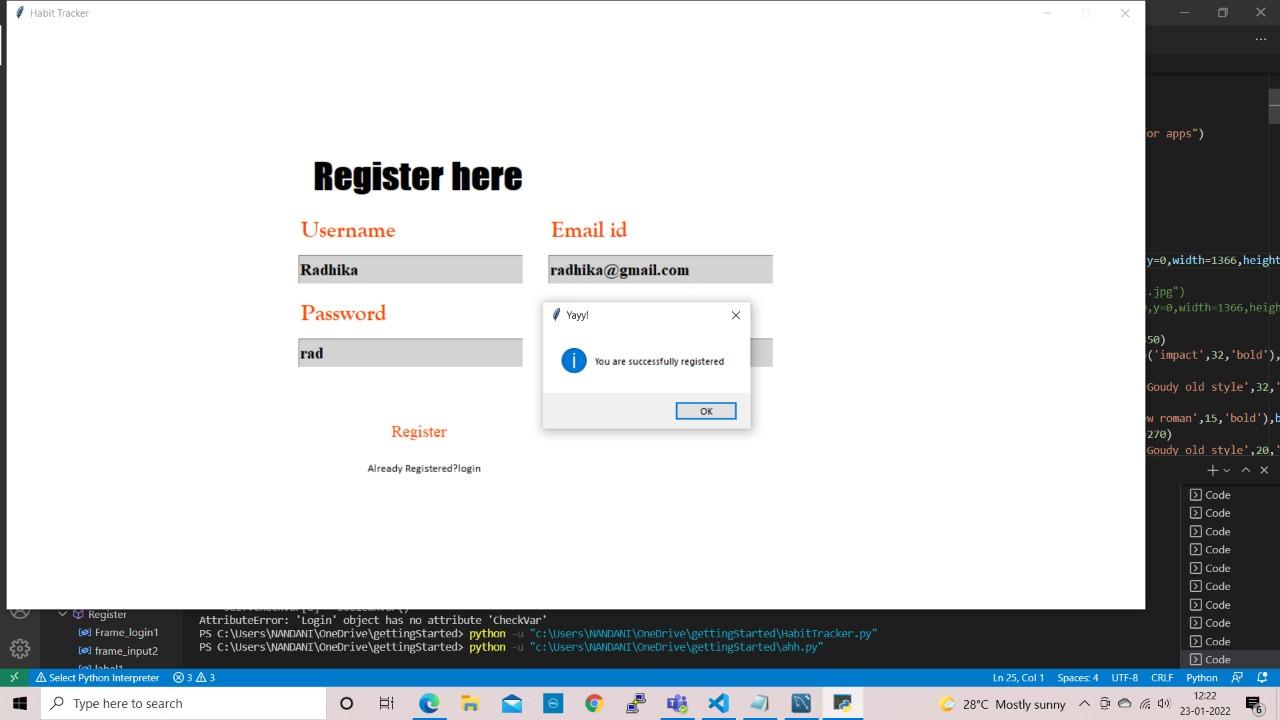
**This is the login page, if the user is registered, he can login else he has to register.**

****

**In case either password or username is not entered and “Login” is clicked, an error will be shown saying “All fields required”.**

****

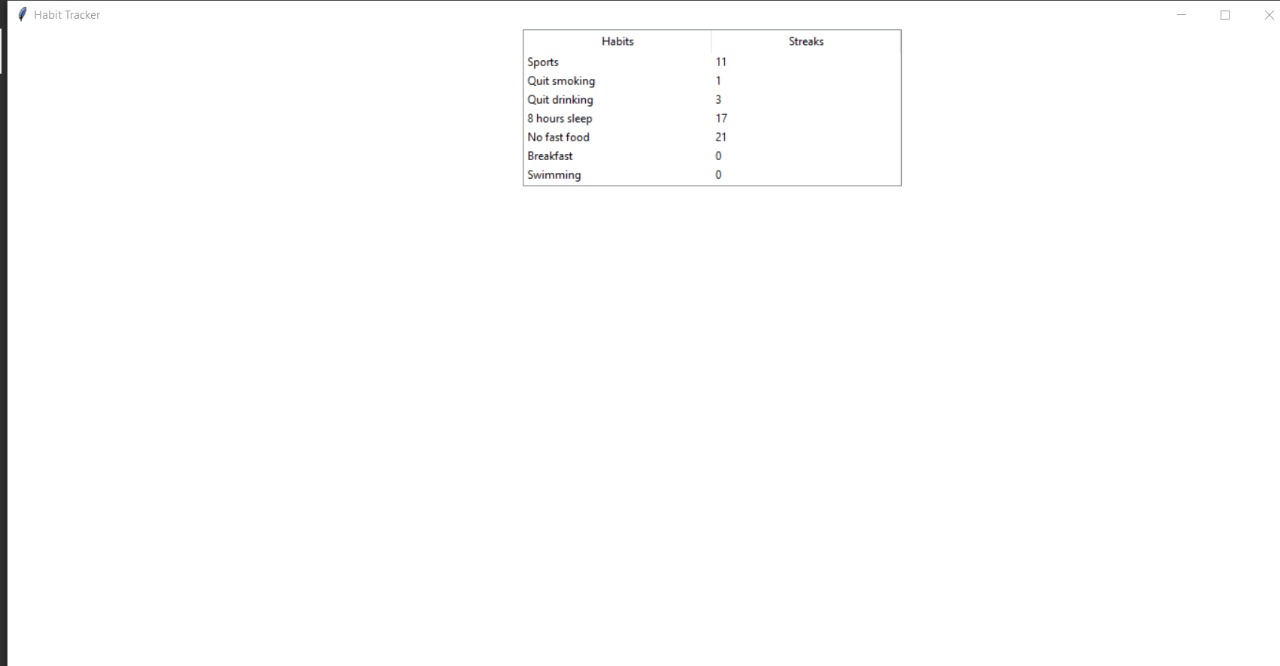
**If the “Register” is clicked, this page will be shown and the user has to enter data in the given fields, if the user clicks “Register” without filling all the fields, an error will be shown saying “All fields required”.**

****

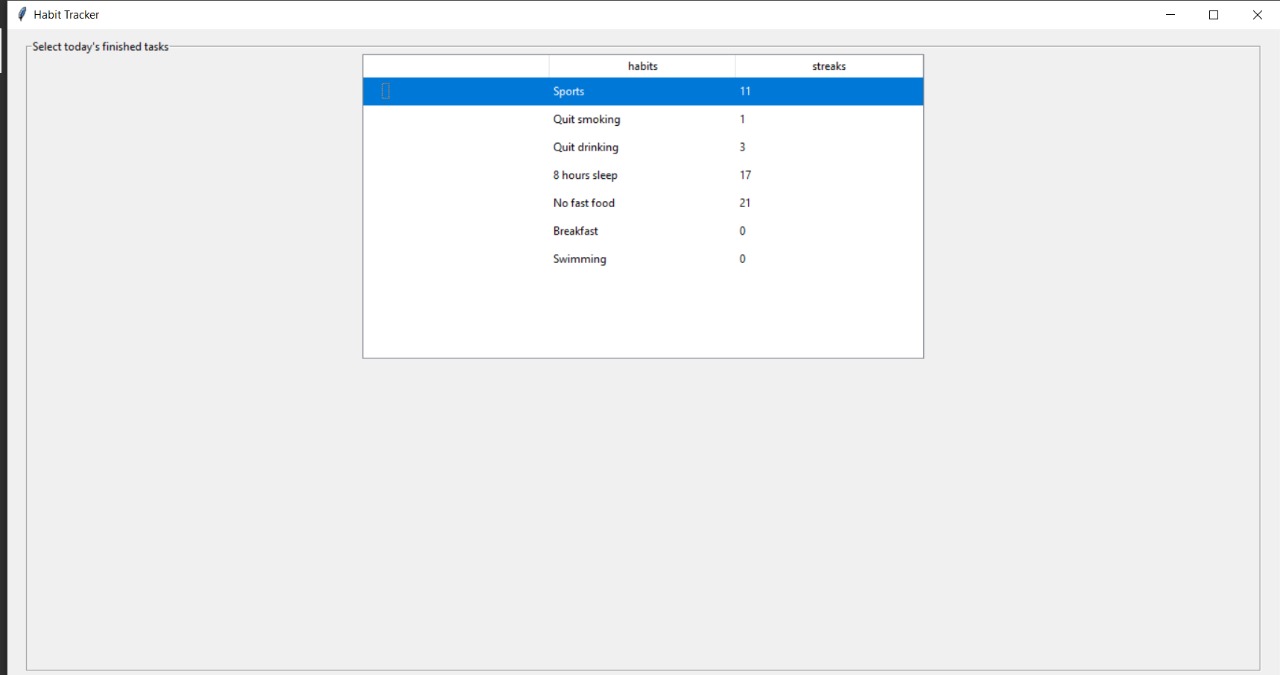
**If the user enters username, email id, password and confirm password correctly, the user is registered.**

****

**When the user logs-in, the Home page is displayed with all the options. The user may choose anything to continue.**

****

**This is the streaks before selecting(doing a goal/habit)**

****

**This page is displayed when the user clicks “Select tasks”, the user can select a task as completed and the streaks will be updated.**

****

**Previously the streaks for Sports was 11 and after selecting the Sports option in “Select tasks”, the streak got incremented.**

****

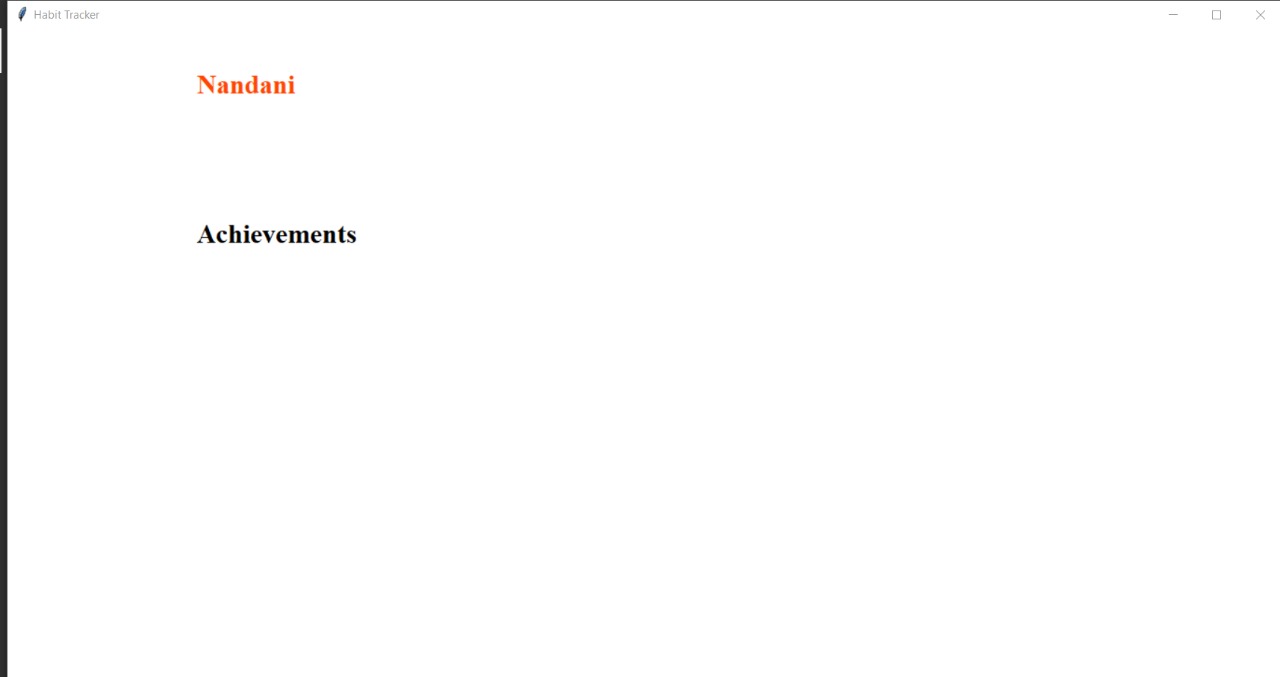
**This is the list of habits and their streaks of the user before.**

****

**This window is displayed when the user clicks “Add new goals”, the user is presented with many options to choose from and when the user selects one, it gets added into the list of habits the user has. For example if the user selects “Swimming”,**

****

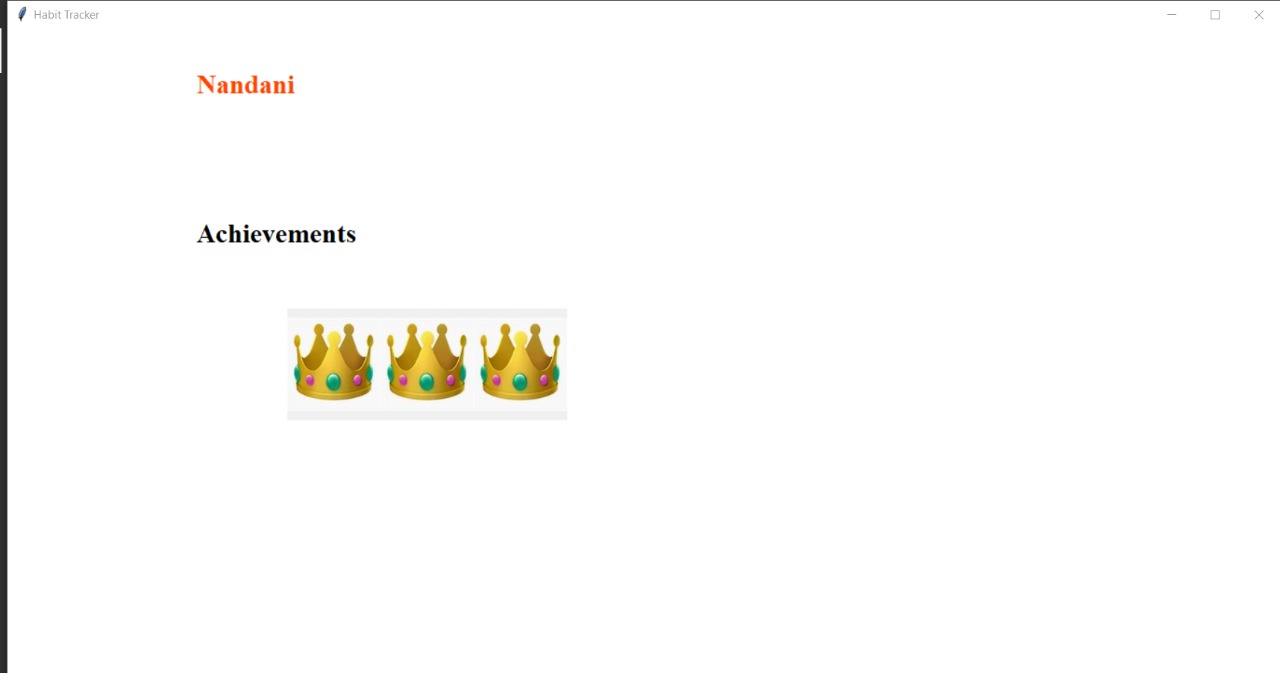
**As you can see, a new row is added under habits named “Swimming”.**

****

**Before(Inside Profile)**

****

**Achievements with claim button if any achievements available to claim, when the claim button is clicked,**

****

**The Profile page is updated with crowns in it after claiming crowns.**

****

**Achievements without claim button because awards already claimed**

**Github link**

**https://github.com/nandaniv**

**ADDITIONAL KNOWLEDGE ACQUIRED**

* Python GUI programming using the module tkinter.
* mysql to store and retrieve data.
* Use case diagrams.
* Activity diagram.

**CONCLUSION –**

Habit Tracker is a simple but useful application that is used to keep track of habits. It is a very basic code that provides a loads of value and is helpful.

**FUTURE WORK –**

* The user can connect through his Facebook account and make habits with his friends.
* Make it a web application rather than a console one.
* Improve the GUI and include the feature to send photos/snaps.
* To include the data on to a database and make it a kind of a mobile app.
* To include a calendar and show when the habits a done and when they are skipped.

**REFERENCES –**

Basic Python

1. Course covered during 2nd semester by Dr. Ramesh Vassapanavara Sir.
2. Ppts and handouts provided by the sir.
3. Python Programming - Using Problem Solving Approach First Edition by Reema Thareja.

Tkinter tutorials

1.<https://www.youtube.com/watch?v=YXPyB4XeYLA>

2. <https://www.youtube.com/watch?v=VMP1oQOxfM0&t=271s>

3. <https://www.youtube.com/watch?v=NkAwxoQkdOA>

Use cases and activity diagram

1. Hand-outs provided by DRL PRASANNA ma’am.
2. <https://www.youtube.com/watch?v=zid-MVo7M-E>
3. <https://www.youtube.com/watch?v=knM8BGY9yVI&t=161s>