

MINI PROJECT TITLE:

FITNESS DISCOVERER

NAME:MOHAN KALYAN

USN:1NH18CS051

REVIEWER:Mrs. SHANMUKAPRIYA .S

DESIGNATION : SR.ASSISTANT PROFESSOR



NEW HORIZON
COLLEGE OF ENGINEERING

Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC
Accredited by NAAC with 'A' Grade, Accredited by NBA

Title

FITNESS DISCOVERER

REPORT

Submitted by

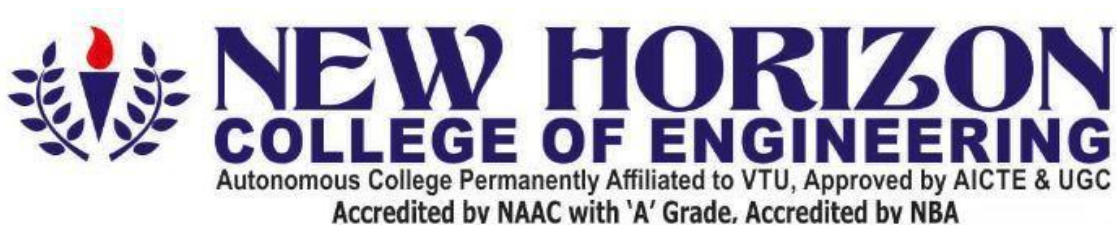
C. MOHAN KALYAN

*In partial fulfillment for the award of
the degree of*

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING



Certificate

This is to certify that the mini project work titled

FITNESS DISCOVERER

*Submitted in partial fulfillment for the award of the degree of Bachelor of
Engineering in Computer Science and Engineering*

Submitted by

C. MOHAN KALYAN

1NH18CS051

During

SEMESTER 2020-2021

For

20CSE59

Signature of reviewer

Signature of HOD

SEMESTER END EXAMINATION

*Name of Examiner
date*

Signature with

1. _____

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be impossible without the mention of the people who made it possible, whose constant guidance and encouragement crowned our efforts with success.

I have great pleasure in expressing gratitude to **Dr. Mohan Manghnani**, Chairman of New Horizon Educational Institutions for providing necessary infrastructure and creating good environment.

I take this opportunity to express my profound gratitude to **Dr. Manjunatha**, Principal NHCE, for his constant support and encouragement.

I would also like to thank **Dr. B. Rajalakshmi**, Professor and Head, Department of Computer Science and Engineering, for her constant support.

I express my gratitude to **Mr./Ms./Dr. Faculty Name, Designation**, my project guide, for constantly monitoring the development of the project and setting up precise deadlines. Her valuable suggestions were the motivating factors in completing the work.

STUDENT NAME (USN)

NAME- C.MOHAN KALYAN

USN- 1NH18CS051

CONTENTS

ABSTRACT	I
ACKNOWLEDGEMENT	II
LIST OF FIGURES	V
1.	
INTRODUCTION	
1.1. INTRODUCTION	1
1.2. COURSE OBJECTIVES	1
1.3. PROBLEM DEFINATION	2
1.4. EXPECTED OUTCOMES	2
2.	
REQUIREMENTS AND DESIGN	
2.1. HARDWARE REQUIREMENTS	3
2.2. SOFTWARE REQUIREMENTS	3
3.	
PYTHON FEATURES	4-5
PYTHON FUNDAMENTALS	6
DBMS FUNDAMENTALS	6
DIAGRAM OF TYPES OF ATTRIBUTES	7
TKINTER WIDGETS	8
SQL	9
4.	
ALGORITHM	12
5.	
E-R DIAGRAM	13
MAPPING	14

6

IMPLEMENTATION

15-27

6.1 MODULE1 FUNCTIONALITY

15

6.2 MODULE2 FUNCTIONALITY

15

6.3 MODULE3 FUNCTIONALITY

16

6.4 MODULE4 FUNCTIONALITY

17

6.5 MODULE4 FUNCTIONALITY

18

6.6 MODULE4 FUNCTIONALITY

25

7.

RESULTS

29-35

8.

CONCLUSION

36

REFERENCES

37

LIST OF FIGURES

<u>Fig. No</u>	<u>Figure Description</u>	<u>Page No</u>
1.1	TYPES OF ATTRIBUTES	7
1.2	SQL CONNECTOR SERVER	10
1.3		11

Abstract:

FITNESS DISCOVERER helps the user to find the gyms .Here in my project the owner has to login as the owner and he has the rights to join the trainers whom ever he wants to,the owner can remove the trainer if he needs .and the customer has to login the here he\she has to give the username and password to gym and has to give the phone number and email address to login the gym . therefore the user can add the trainer if they want to and has the option to remove also so here mainly it helps the user to add whom ever they want .

While adding the trainer he has to give the username and password and name of the trainer and phone number of the trainer and email address of the trainer so here in my project the customer can get to know all the details of the trainer easily by clicking the trainer whom they want .and here in my project the owner has the right to remove the customer or can add the customer . every time the customer has to login the site to know about the details of the trainer she can easily remove or change the trainer or can add the trainer by one click.

And we know the situation we facing now this pandemic created alot of problems among us .in this situation its very difficult to step out of the house so my project helps the user to login through the online which makes easy and can know about the gyn by sitting in home .

CHAPTER 1

INTRODUCTION

FITNESS DISCOVERER helps the user to find the gyms .Here in my project the owner has to login as the owner and he has the rights to join the trainers whom ever he wants to,the owner can remove the trainer if he needs .and the customer has to login the here he\she has to give the username and password to gym and has to give the phone number and email address to login the gym . therefore the user can add the trainer if they want to and has the option to remove also so here mainly it helps the user to add whom ever they want .While adding the trainer he has to give the username and password and name of the trainer and phone number of the trainer and email address of the trainer so here in my project the customer can get to know all the details of the trainer easily by clicking the trainer whom they want .and here in my project the owner has the right to remove the customer or can add the customer . every time the customer has to login the site to know about the details of the trainer she can easily remove or change the trainer or can add the trainer by one click.

1.1 Course Objectives

- The fundamentals of python programming should be understood. All the features of python and the usage of python has to be understood.
- Using the features of python one should be able to create an own application.
Using all the latest features of python a real time application has to be developed.
- One should be able to design a database or an algorithm using the connectivity between database and python. Where the foreground is python and background would be database. An application has to be created in that manner.
- One should be able to implement fully functional python application connecting to a database or algorithm implemented. It should give an appropriate outputs or results.

1.2 Problem Definition

In my project I will help user to find the gym ,here we can add the trainers and can remove them by one click it helps the by making them easy way and the owner has to login first and can add the customers and trainers too it helps them to easy and the customers can easily know the details of the trainers.While adding the trainer he has to give the username and password and name of the trainer and phone number of the trainer and email address of the trainer so here in my project the customer can get to know all the details of the trainer easily by clicking the trainer whom they want .and here in my project the owner has the right to remove the customer or can add the customer . every time the customer has to login the site to know about the details of the trainer she can easily remove or change the trainer or can add the trainer by one click.

1.3 Outcomes of Project Work

The project gives the login page first which the owner has to login by giving the details and address username and password . here after the owner has to add the trainers or remove the trainers if he want ,then later customer has to login the site by giving id, username , password and phonenumber and email address ,after that he\she can add the trainers whom ever they want too and can also remove accordingly to their fitness . the customer can alsochange the trainer if they want, it shows the date ,time ,day it, takes from the computer. Owner has to login only once multiple login shows error ,and the customer has login with the same username and password different username and password give the invalid syntax.

CHAPTER 2

REQUIREMENTS AND DESIGN

2.1 Hardware Requirements

- Processor: X64-based processor CPU with 1.80GHz clock speed.
- System: Intel Core
- RAM: 16.0 GB
- Hard Disk: 1TB

2.2 Software Requirements

- IDLE: Eclipse Python 3.7.4
- Compiler: Python 3.7.4
- Data base software: MySQL
- OS: Windows 10

CHAPTER-3

PYTHON FEATURES

1. Python Language is simple to learn and easy to implement bigger application programs

Python is a simple and user friendly language. It has easy syntax to create any application easily. Compared to other languages Python is simpler and if other languages takes 15 lines of code for a program it can be simplified into 3-4 line in python. Python's feature is very useful. Nowadays python is used exclusively for real time applications. It is very easy to learn language. For beginners this language becomes easier compared to other languages because the syntax and steps are easy. All complicated features are made simple in python.

2. Python language is Platform non dependent

Platform independent means once a program is written in one IDLE or platform then it has to be compatible in any other platform. Similar to java, Python is also platform independent language. Python once typed in one application can be run in any other applications easily. This is one of the best feature in python. This also supports re usability since once the code is written we need not write it again.

3. Python language is a Dynamically typed language

Usually in other programming language it is mandatory to declare a variables before using them but in python it dynamically allocates the data types for the variables. In python we need not declare the data types of the variables. Whenever a value is initialized to a variable based on the type of values the data type is assigned to a variable. This makes it easy for the user because the user can directly enter a value to a variable. He need not declare the variables initially.

4. Python language has extensive libraries.

Python provides various built-in libraries. The programmers can easily use the built in libraries to develop their application. It makes it very easy for the user to develop complex real time applications using these libraries. By using these libraries the

development becomes easier. One can easily add and use the third party libraries in python. This is one of the best libraries. Graphs and any other development can be performed.

5. Python helps in developing GUI and web based programs.

Using Python's libraries we can develop Graphical user interface. For example we can design a GUI using tkinter. Widgets can be created easily using this application of python. Hence, GUI programming becomes easier. Using python language we can also develop web based applications using python. It becomes easier for the developer to create web based application.

These are few of the features of Python Which makes it easier for the user to develop applications. Compared to other programming languages python makes the task easier for the user. It uses Object oriented programming concepts. Thereby, Python is one of the easy programming language for a developer.

6. Python is a good Interpret language.

The languages which are familiar to c++ or java which has to be compiled and then we have to run the program. But in Python we don't have to compile it rather we have to directly run it. Python has a internal source code which converts into a immediate form called bytecode. Therefor, all we need is to run the python code without worrying about linking to libraries and a few other things in C++ or java.

By interpret we can execute the source code line by line where in C++ and java we have to run full code at once because this might be easier to debug your code much faster and much easier. Also interpreting makes Java and C++ little slower, but that because Java has many features which it offers.

PYTHON FUNDAMENTALS

Python is a scripting programming language known for both its simplicity and wide breadth of applications. For this reason, it is considered one of the best languages for beginners. Used for everything from Web Development to Scientific Computing (and SO much more), Python is referred to as a “general purpose” language by the greater programming community.

STATEMENTS:

Python statements are nothing but logical instructions that interpreter can read and execute. It can be both single and multiline. There are two categories of statements in Python:
Expression Statements
Assignment Statements

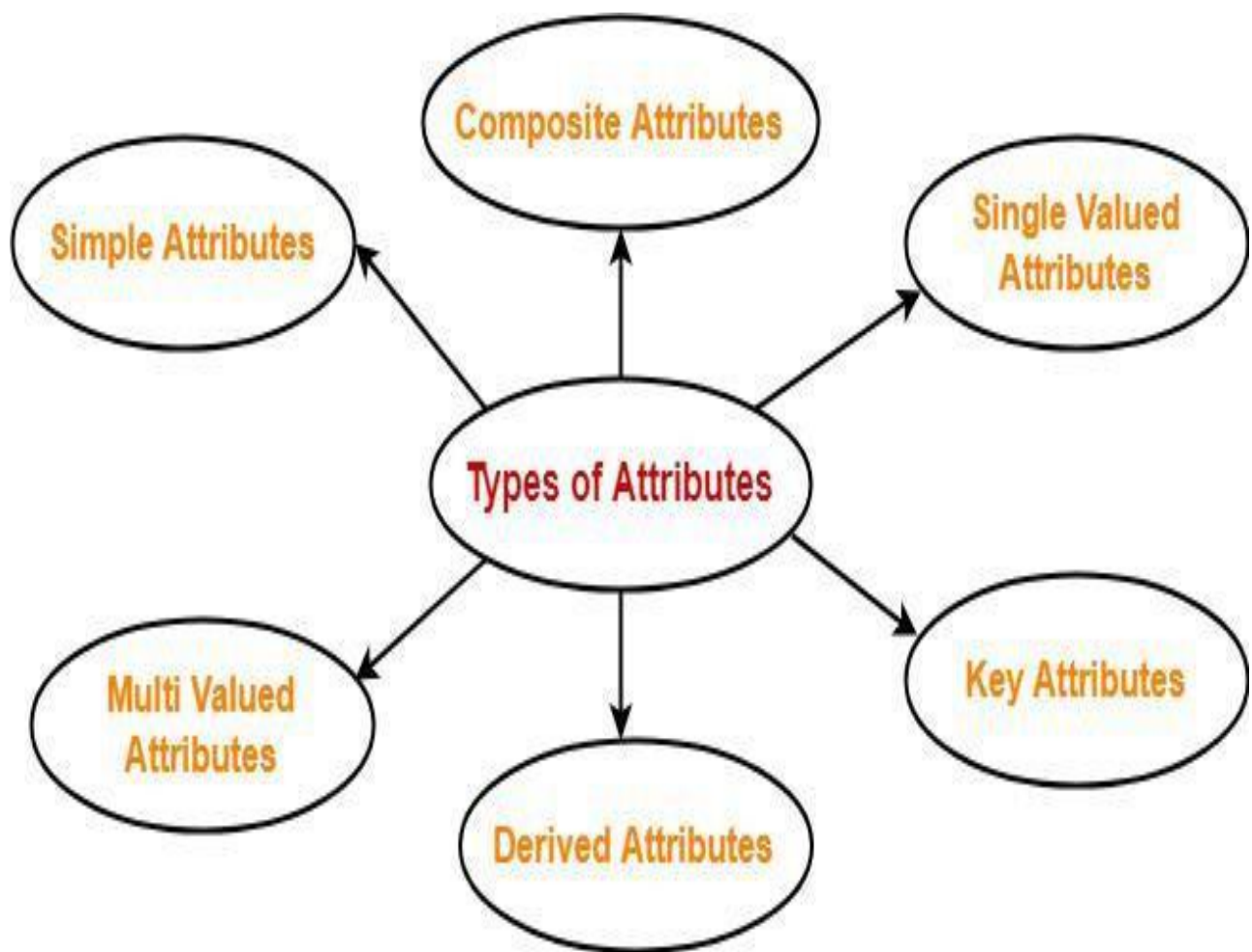
DBMS FUNDAMENTALS

ENTITY AND ATTRIBUTES

Entity type is collection of entities which consists of various attributes which may be of various data types and can have various constraints. While selecting an entity we must always care of two things, does that selected entity have enough members and also enough attributes. If the entity satisfies both the conditions then such entities are called

good entity but if the entity fails to satisfy one of these conditions then such entities are considered to be bad entities

. Attributes for each entity should have a proper data type assigned to it and may or may not have constraints. Intangible Entity: Intangible Entities are those entities which exist only logically and have no physical existence. Example: Bank Account, etc. Strong Entity Type: Strong entity are those entity types which has a key attribute. Weak Entity Type: Weak entity type doesn't have a key attribute. Types of attributes: In ER diagram, attributes associated with an entity set may be of the following types

**FIGURE 1**

TKINTER WIDGETS

Python provides multiple options for GUI(graphical user interface). Tkinter GUI tool kit is used for creating GUI applications. There are many widgets available in python tkinter.

First step is to import tkinter

Import tkinter

Second is to create a window:

Root=Tk() -> this statement creates a window

Third you can create widgets. The widgets can be placed onto the window in 3 ways:

1. Pack() : It organizes widgets into blocks and then places into the parent window.
2. Grid() : It divides the parent window into rows and columns and places the widgets in the specified row and column.
3. Place() : it organizes the widgets by placing them in particular position mentioned by the programmer.

Syntax: Widget.placing_Method()

There are many widgets in Tkinter. Let us discuss about 5 important widgets.

1. **Button**: It is used to create buttons in to Python language.

Syntax : w = Button (master_window, option = value, ...)

Ex: W = Button (root , text = 'Hello' , font=('Cambria',20)).pack()

2. **Label** : This widget implements a display box where you can place text or images.

Syntax: w = Label (master_window, option, ...)

Ex: W = Label(root, text = 'User name ').pack()

3. **Message box**: This widget is used to prompt any message while clicking on any button.

Ex: messagebox.warning('warning')

There are many message boxes available like askyesorno, error and more

4. **Entry widget**: The Entry widget is used to accept single-line text strings from a user.

Syntax: w = Entry(master_window, option, ...)

Ex: `w = Entry (root, textvariable=var1).grid(row=1,column=1)`

5. **Check Button** : The Checkbutton widget is used to display a number of options to a user as toggle buttons. It helps the user to select one or more options by the clicking button corresponding to each option.

Syntax: `w = Checkbutton (master_window, option, ...)`

Ex: `w = Checkbutton (root, variable=var1, onvalue=1).pack()`

SQL

What is SQL?

- SQL stands for Structured Query Language
- SQL lets you access and manipulate databases
- SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987

What Can SQL do?

- SQL can execute queries against a database
- SQL can retrieve data from a database
- SQL can insert records in a database
- SQL can update records in a database
- SQL can delete records from a database
- SQL can create new databases
- SQL can create new tables in a database
- SQL can create stored procedures in a database
- SQL can create views in a database
- SQL can set permissions on tables, procedures, and views

SQL is a Standard - BUT....

Although SQL is an ANSI/ISO standard, there are different versions of the SQL language.

However, to be compliant with the ANSI standard, they all support at least the major commands (such as SELECT, UPDATE, DELETE, INSERT, WHERE) in a similar manner.

SQL CONNECTION SERVER

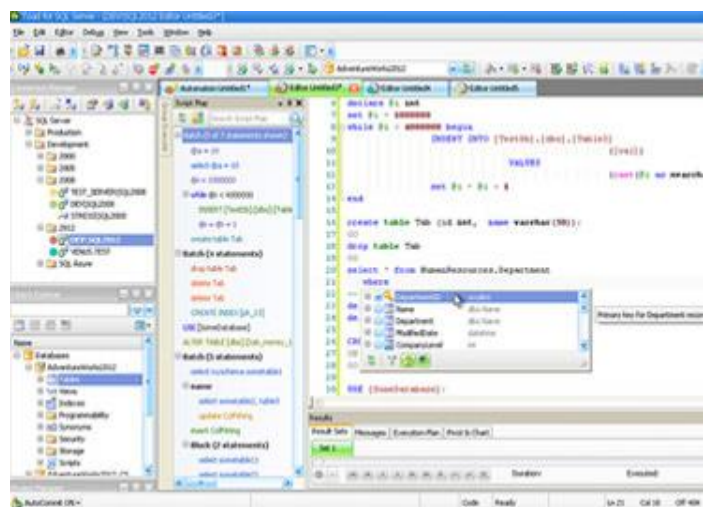
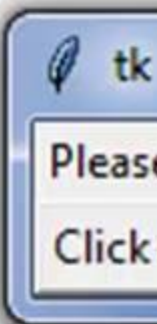


FIGURE 2

```
from tkinter import *  
  
my_window=Tk()  
  
label_1=Label(my_window,text="Please  
entry_1 = Entry(my_window)  
button_1=Button(my_window,text="Click")  
  
label_1.grid(row=0,column=0)  
entry_1.grid(row=0,column=1)  
button_1.grid(row=1,column=0)  
  
my_window.mainloop()
```



It is like having a personal tutor

Figure 3

CHAPTER 4

ALGORITHM

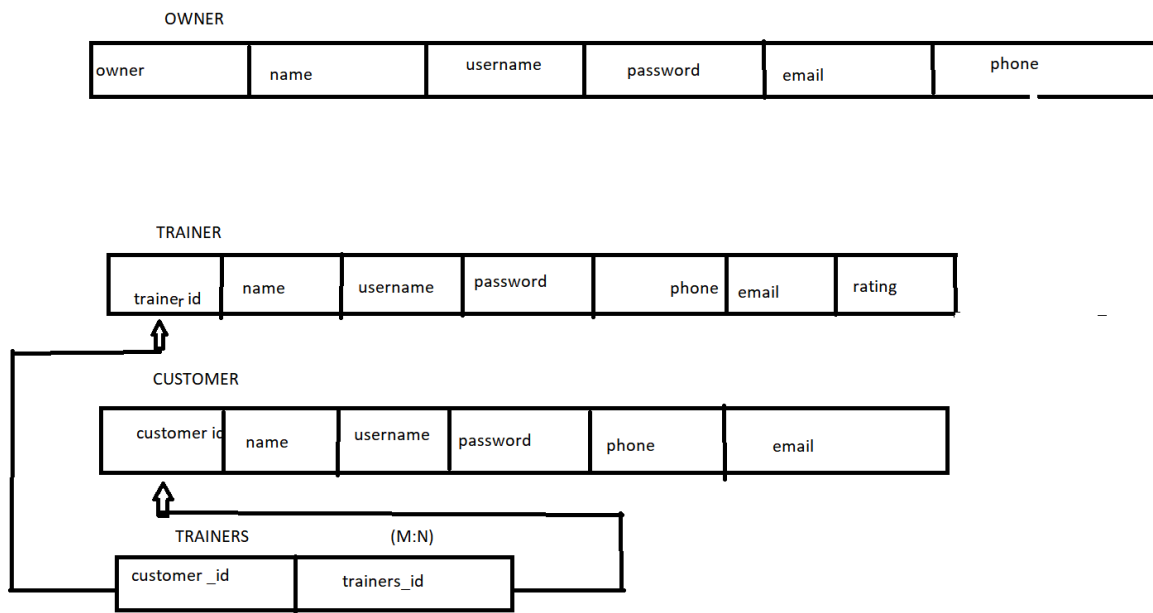
1. Here the login page opens , the owner has to login through the name and user name and password
2. And the owner has give the phone number and email address.
3. Here the owner can add the trainers by entering the details of the trainers.
4. The id of the trainers and username and password ,the owner has to enter the phone number and email address of the trainers
5. after the details of the trainers ,here it comes the customer should enter the details of the customer
6. customer should give id of the customer and should create the user name and password and give the phone number and email address of the customer
7. After the customer loggedin the customer can choose the trainer of their instrest.
8. He\she can choose the trainer whom ever they want too and can remove the trainer if they are not instrested at anytime .
9. The customer can also change the trainer if they want.
10. FITNESS DISCOVERER helps the user to choose online without stepping out of the house.

CAPTER -5

ER -DIAGRAM



MAPPING



Chapter 6

IMPLEMENTATION

6.1.

first a connector between python and mysql has to be set up and we should have mysql connector installed for that and the connection is in following manner:

```
● import mysql.connector as mysql
from tkinter import *
import datetime as dt
import functools
from tkinter import messagebox
```

```
db = mysql.connect(
    host="localhost",
    user="root",
    passwd="kalyan"
)
```

6.2.

Tables are created in python and tables are actually created in the background in mysql. In the foreground we run the commands and they are executed in the background hence, tables are created using the following commands:

```
cursor = db.cursor(buffered=True)

#cursor.execute("create database kalyan")
cursor.execute("use kalyan")
#cursor.execute("CREATE TABLE
    owner(owner_id INT(5) NOT NULL
    AUTO_INCREMENT PRIMARY
```

```
KEY,Username varchar(30)
UNIQUE>Password varchar(30),Name
varchar(30),Phone varchar(30),Email
varchar(30))")
```

6.3.

Here we should enter the details of the owner like id ,username and password and owner has to give the phone number and email address of the owner

```
#cursor.execute("CREATE TABLE
customer(cus_id INT(5) NOT NULL
AUTO_INCREMENT PRIMARY
KEY,Username varchar(30)
UNIQUE>Password varchar(30),Name
varchar(30),Phone varchar(30),Email
varchar(30))")

#cursor.execute("CREATE TABLE
trainer(trainer_id INT(5) NOT NULL
AUTO_INCREMENT PRIMARY
KEY,Username varchar(30)
UNIQUE>Password varchar(30),Name
varchar(30),Phone varchar(30),Email
varchar(30),rating INT(5))")

#cursor.execute("CREATE TABLE trains(cus_id
INT(5),trainer_id INT(5), constraint p1
primary key(cus_id,trainer_id),constraint
f1 foreign key(cus_id) references
customer(cus_id) on delete cascade on
update cascade, constraint f2 foreign
key(trainer_id) references
```



```

trainer(trainer_id) on delete cascade on
update cascade)")

```

```

def login_owner(uid,Password):
    try:
        q = "select * from owner where
            Username='"+str(uid)+"'"
        cursor.execute(q)
        a = cursor.fetchall()
        if Password == a[0][2]:
            dashboard_owner(a)
    except IndexError:
        messagebox.showinfo("Error","No user:
            "+str(uid))

```

6.4.

here we will create the details of the trainer like id of the trainer And username and password and phone number and email address of the trainer and we can remove the trainer if we want .

```

def
    add(Username,Password,Name,Phone,E
        mail,root):
    try:
        query = "INSERT INTO
            trainer(Username,Password,Name,Phon
                e,Email,rating)
            VALUES(%s,%s,%s,%s,%s,1)"

```

```

        values =
            (Username>Password>Name>Phone>Email
            )
        cursor.execute(query,values)
        db.commit()
        root.destroy()
    except:
        messagebox.showinfo("Error","User
        already Exist")

def remove_trainer(uid,root):
    q = "delete from trainer where
        trainer_id="+str(uid)
    cursor.execute(q)
    db.commit()
    root.destroy()
    alltrainer()
just been created )
c

root.title("Dashboard (Owner)")
root.geometry("500x500")

```

6.5.

The labeling part comes here we should label date and name ,username and password and give the phone number and email address.

```

date = dt.datetime.now()
format_date = f"{date:%a, %b %d %Y}"
w = Label(root, text=format_date,
           fg="white", bg="black",

```

```
font=("helvetica", 10))
w.place(x=380,y=10)

root.config(bg="black")

Label3 = Label(root,text="Profile
    ",font=("Helvetica",15),bg="black",fg="
    white")
Label3.place(x=10,y=30)

Label6 = Label(root,text="Name
    :"+a[0][3],font=("Helvetica",12),bg="bla
    ck",fg="white")
Label6.place(x=50,y=80)

Label7 = Label(root,text="Phone
    :"+a[0][4],font=("Helvetica",12),bg="bla
    ck",fg="white")
Label7.place(x=50,y=110)

Label8 = Label(root,text="Email
    :"+a[0][5],font=("Helvetica",12),bg="bla
    ck",fg="white")
Label8.place(x=50,y=140)

Button1 = Button(root,text="Add
    Trainer",width=50,command=signup_tr
    ainer)
Button1.place(x=60,y=250)
```

```
Button2 = Button(root,text="All  
Trainers",width=50,command=alltraine  
r)  
Button2.place(x=60,y=300)
```

```
Button3 = Button(root,text="All  
Customers",width=50,command=allcus  
t)  
Button3.place(x=60,y=350)
```

```
Button4 = Button(root,text="Log  
out",bg="red",command=root.destroy)  
Button4.place(x=380,y=450)
```

```
root.mainloop()
```

```
" +str(name[3])+"\n\nPhone:" +str(name[4])+"  
 \n\nEmail :"+str(name[5])+"\n\nRating  
 :"+str(name[6])  
Details.config(text=t)
```

```
Button2 =  
Button(Details,text="Join",bg="lightgre  
en",width=8,command=lambda:join(ui  
d,name[0]))  
Button2.place(x=625,y=425)
```

```
mylist = dbtrainer()
for item in mylist:

    button =
        Button(frame2,text=item[1],command=
            functools.partial(func,item),height=5,wi
            dth=25,bd=2,bg="white",fg="black",fo
            nt=("verdana", 12))
    button.pack()
```

```
frame2.update() # update frame2 height so
    it's no longer 0 ( height is 0 when it has
    just been created )
```

```
canvas_container.configure(yscrollcom
mand=myscrollbar.set, scrollregion="0
0 0 %s" % frame2.winfo_height()) # the
scrollregion mustbe the size of the
frame inside it,
```

```
#in this case "x=0 y=0 width=0
height=frame2height"
```

```
#width 0 because we only scroll
vertically so don't mind about the
width.
```

```
canvas_container.pack(side=LEFT)
myscrollbar.pack(side=RIGHT, fill = Y)

frame_container.grid(row=0,column=0)

root.mainloop()
```

```
def dbcustomer():
    q = "select * from customer"
    cursor.execute(q)
    return (cursor.fetchall())

def delete_cust(uid,root):
    q = "delete from customer where
        cus_id="+str(uid)
    cursor.execute(q)
    db.commit()
    root.destroy()
    allcust()
```

```
def allcust():
    def _on_mousewheel(event):
        #print(-1*(event.delta/120))
        canvas_container.yview_scroll(int(-
            1*(event.delta/120)), "units")
```

```
root = Tk()

root.geometry("1000x500")

root.title("All Customers")

root.configure(background='black')


frame_container=Frame(root,width=1000,height=1000,bg="black")

canvas_container=Canvas(frame_container,
    height=1000,width=250,bg="black")

frame2=Frame(canvas_container,bg="black")

myscrollbar=Scrollbar(frame_container,
    orient="vertical",command=canvas_container.yview,width=5) # will be visible
if the frame2 is too big for the canvas

canvas_container.bind_all('<MouseWheel>','_on_mousewheel')

canvas_container.create_window((0,0),window=frame2,anchor='w')

Details =

Label(root,width=80,height=27,font=("Helvetica",
    12),anchor='nw',justify="left")
```

```
Details.place(x=265,y=0)
```

```
def func(name):
```

```
    t = "Customer ID :
```

```
        "+str(name[0])+"\n\nUsername:
```

```
        "+str(name[1])+"\n\nName :
```

```
        "+str(name[3])+"\n\nPhone:" +str(name
```

```
        [4])+"\n\nEmail :"+str(name[5])
```

```
Details.config(text=t)
```

```
Button2 = Button(Details,text="Delete
```

```
Customer",bg="orange",command=la
```

```
mbda: delete_cust(name[0],root))
```

```
Button2.place(x=625,y=425)
```

```
mylist = dbcustomer()
```

```
for item in mylist:
```

```
    button =
```

```
        Button(frame2,text=item[3],command=
```

```
        functools.partial(func,item),height=5,wi
```

```
        dth=25,bd=2,bg="lightgreen",fg="blac
```

```
        k",font=("verdana", 12))
```

```
    button.pack()
```

```
frame2.update() # update frame2 height so
```

```
    it's no longer 0 ( height is 0 when it has
```

```
    just been created )
```

```
    canvas_container.configure(yscrollcom
```

```
    mand=myscrollbar.set, scrollregion="0
```

```
    0 0 %s" % frame2.winfo_height()) # the
```



```
scrollregion mustbe the size of the  
frame inside it,
```

```
#in this case "x=0 y=0 width=0  
height=frame2height"
```

```
#width 0 because we only scroll  
verticaly so don't mind about the  
width.
```

```
canvas_container.pack(side=LEFT)  
myscrollbar.pack(side=RIGHT, fill = Y)
```

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

```
root.mainloop()
```

```
def signup():  
    root = Tk()  
    root.geometry("1000x1000")  
    root.title("Kalyan")
```

6.6.

Here the customer after creating the account customer can add the trainer or else if he\she wants remove the trainer also they can do it

```
Label1 = Label(root,text="FITNESS  
DISCOVERER",font=("Helvetica",20))  
Label1.place(x=430,y=50)
```

```
Label3 = Label(root,text="Create
```

```
Account",font=("Helvetica",16))
Label3.place(x=450,y=100)

Label4 = Label(root,text="Username
:",font=("Helvetica",12))
Label4.place(x=350,y=150)

Label5 = Label(root,text="Password
:",font=("Helvetica",12))
Label5.place(x=350,y=180)

Label6 = Label(root,text="Name
:",font=("Helvetica",12))
Label6.place(x=350,y=210)

Label7 = Label(root,text="Phone
:",font=("Helvetica",12))
Label7.place(x=350,y=240)

Label8 = Label(root,text="Email
:",font=("Helvetica",12))
Label8.place(x=350,y=270)

E1 = Entry(root)
E1.place(x=440,y=155,height=20)

E2 = Entry(root,show="*")
E2.place(x=440,y=185)

E3 = Entry(root)
```

```
E3.place(x=440,y=215,height=20)
```

```
E4 = Entry(root)
```

```
E4.place(x=440,y=245,height=20)
```

```
E5 = Entry(root)
```

```
E5.place(x=440,y=275,height=20)
```

```
Button1 = Button(root,text="Sign  
up",width=10,font=("helvetica",15),com  
mand=lambda:add_cust(E1.get(),E2.get(  
,E3.get(),E4.get(),E5.get(),root))
```

```
Button1.place(x=440,y=310)
```

```
Button3 = Button(root,text="Log  
in",command = lambda:  
login(root),width=10,font=("helvetica",  
15))
```

```
Button3.place(x=630,y=110)
```

```
root.mainloop()
```

```
signup()
```

CHAPTER 7

OUTPUT SNAPSHOTS

7.1 Output 1



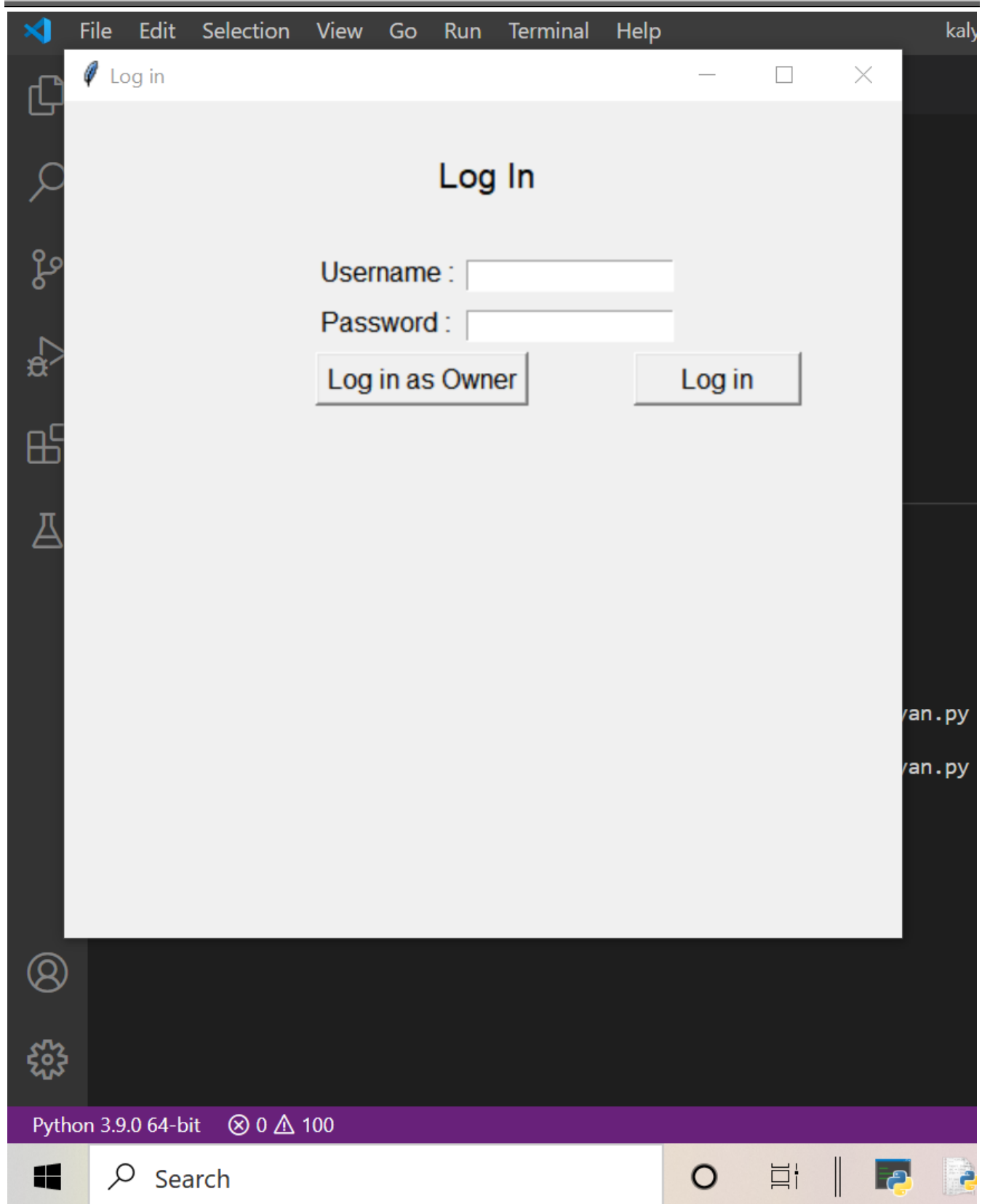
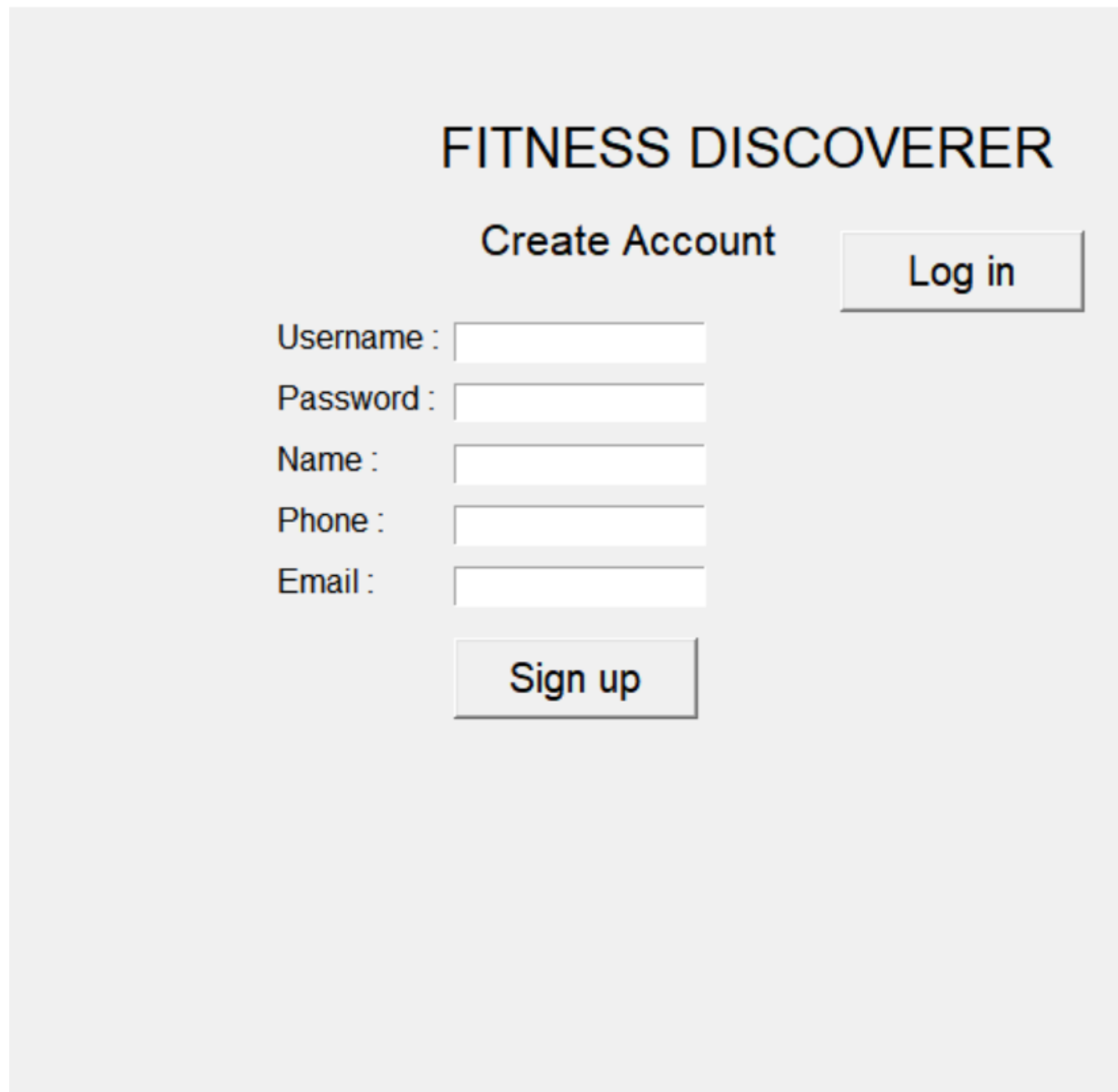


Figure 1

7.2 Output 2



The screenshot displays a web interface for 'FITNESS DISCOVERER'. At the top center, the title 'FITNESS DISCOVERER' is shown in a large, bold, black font. Below the title, there are two main options: 'Create Account' and 'Log in'. The 'Log in' option is enclosed in a rectangular button with a thin border. To the left of these options, there are five input fields for registration, each preceded by a label: 'Username :', 'Password :', 'Name :', 'Phone :', and 'Email :'. Below the 'Email :' field, there is a 'Sign up' button, also enclosed in a rectangular box with a thin border. The entire form is set against a light gray background.

FITNESS DISCOVERER

Create Account [Log in](#)

Username :

Password :

Name :

Phone :

Email :

[Sign up](#)

Figure 2

7.3 Output 3

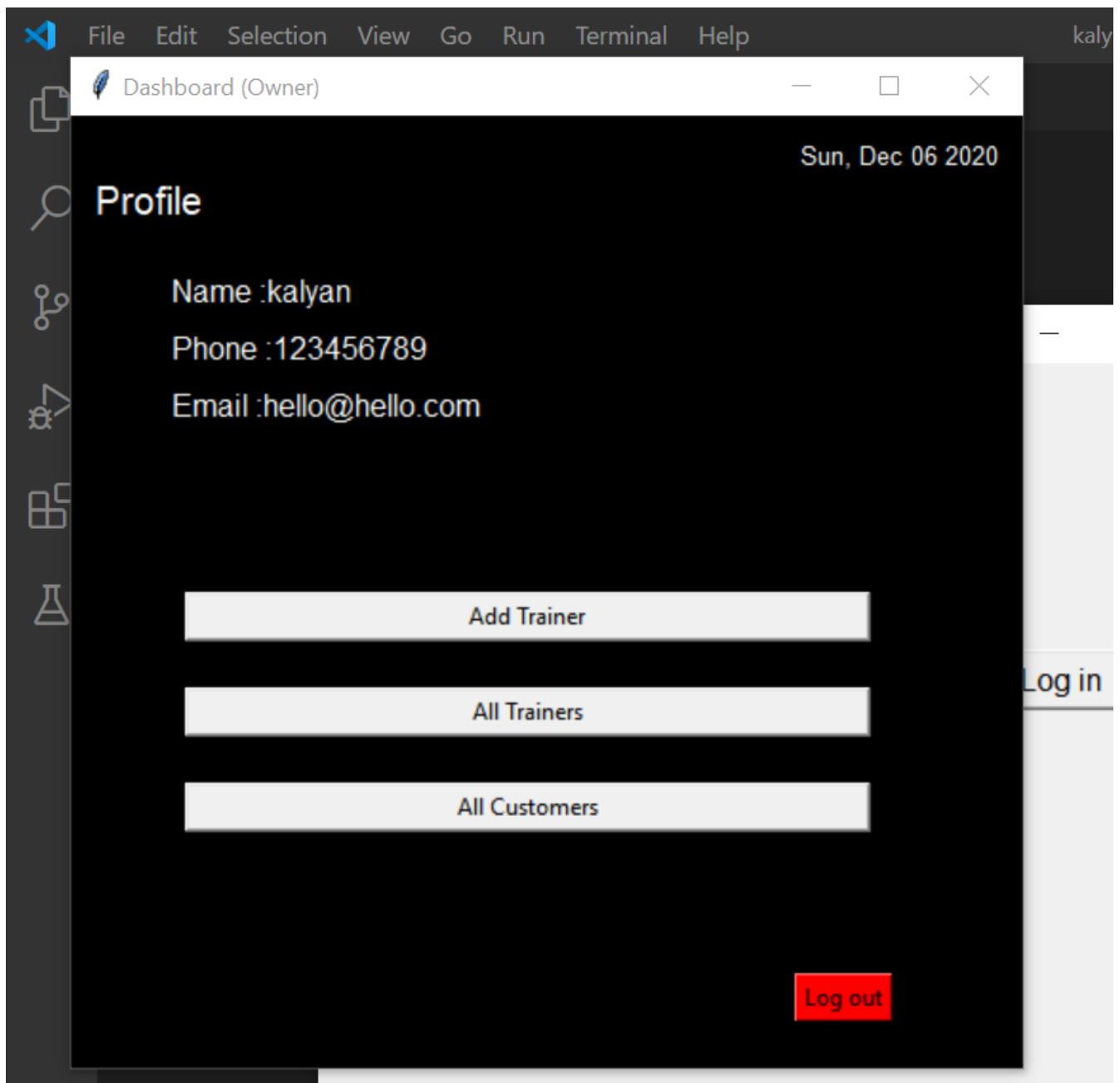


Figure 3

7.4 Output 4

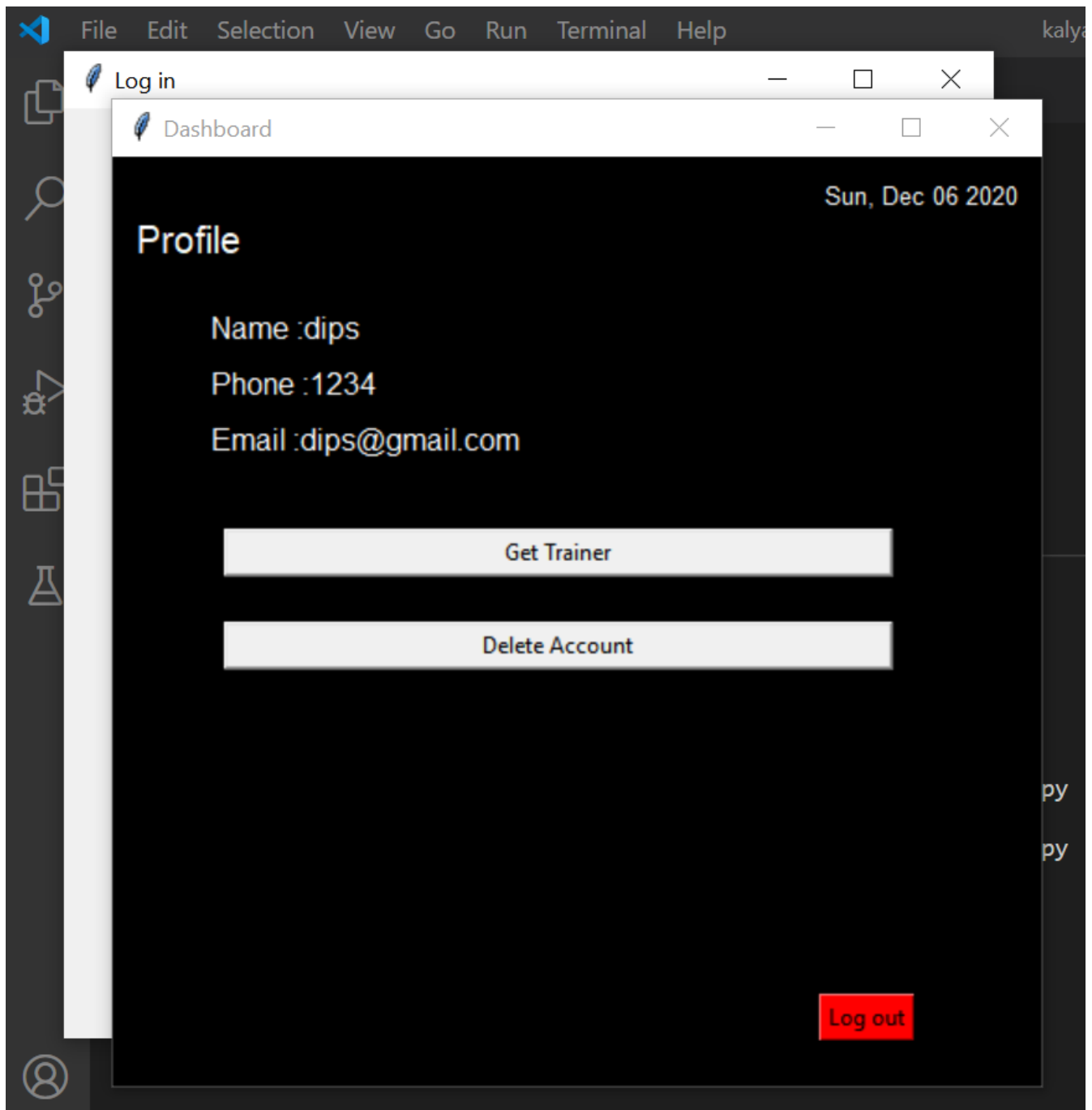


Figure 4

7.5 Output 5

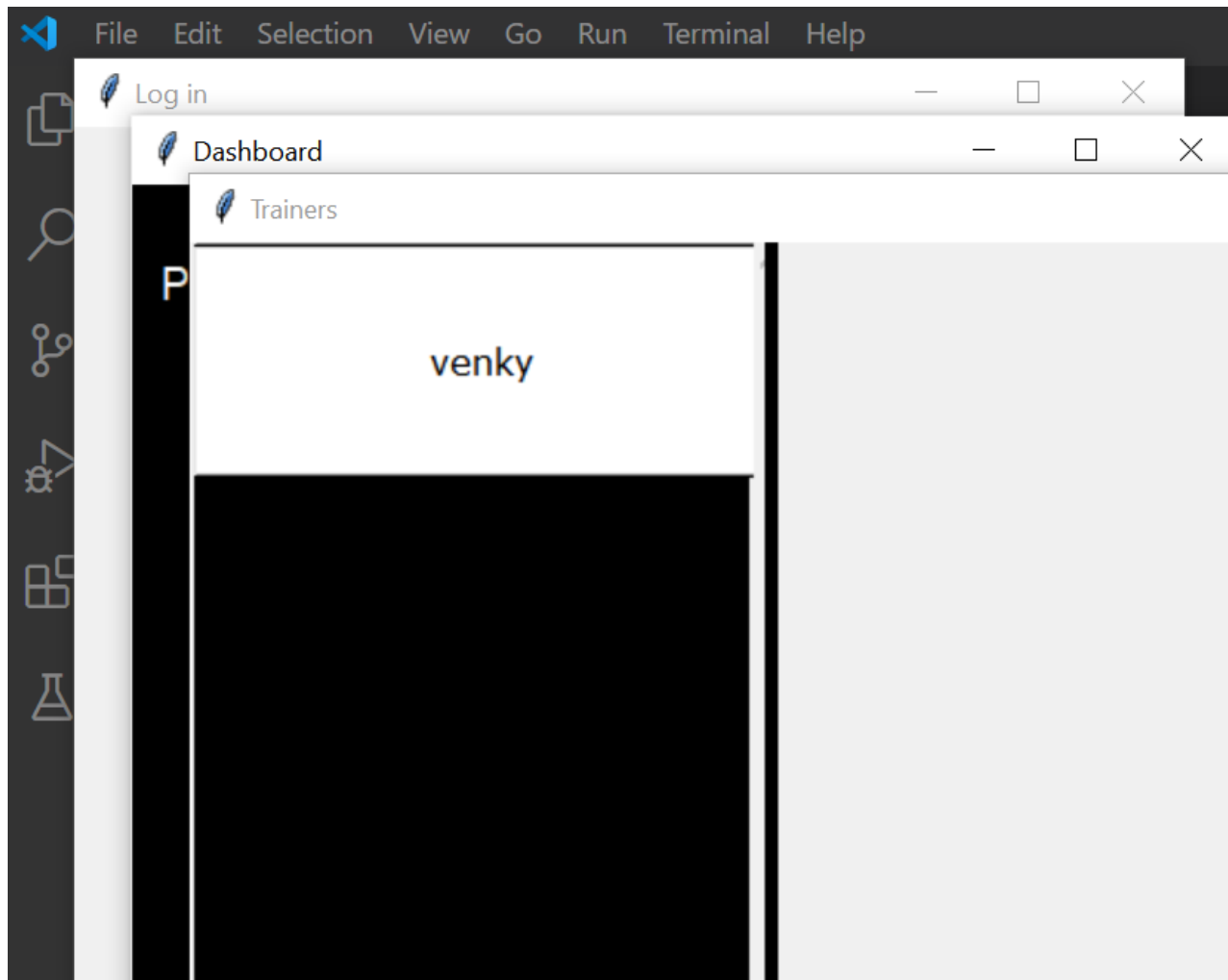


Figure 5

7.6 Output 6

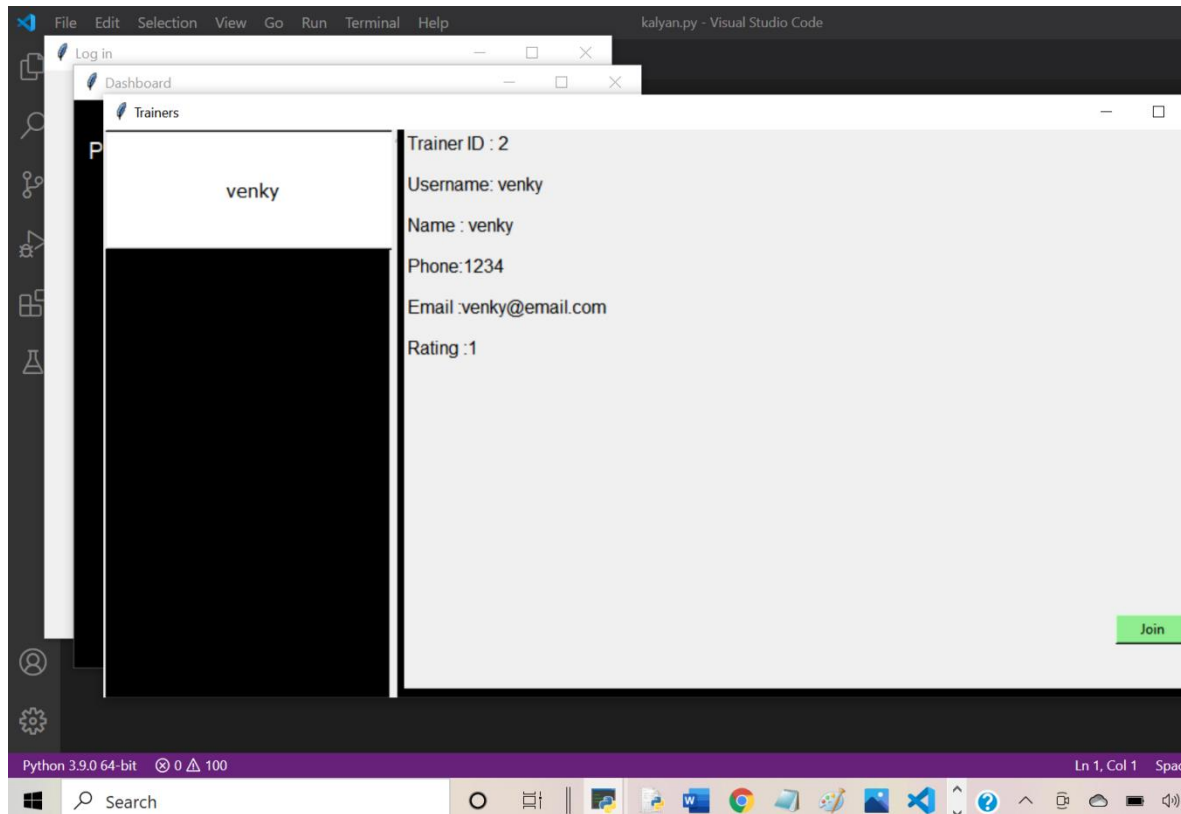


Figure 6

7.7 Output 7

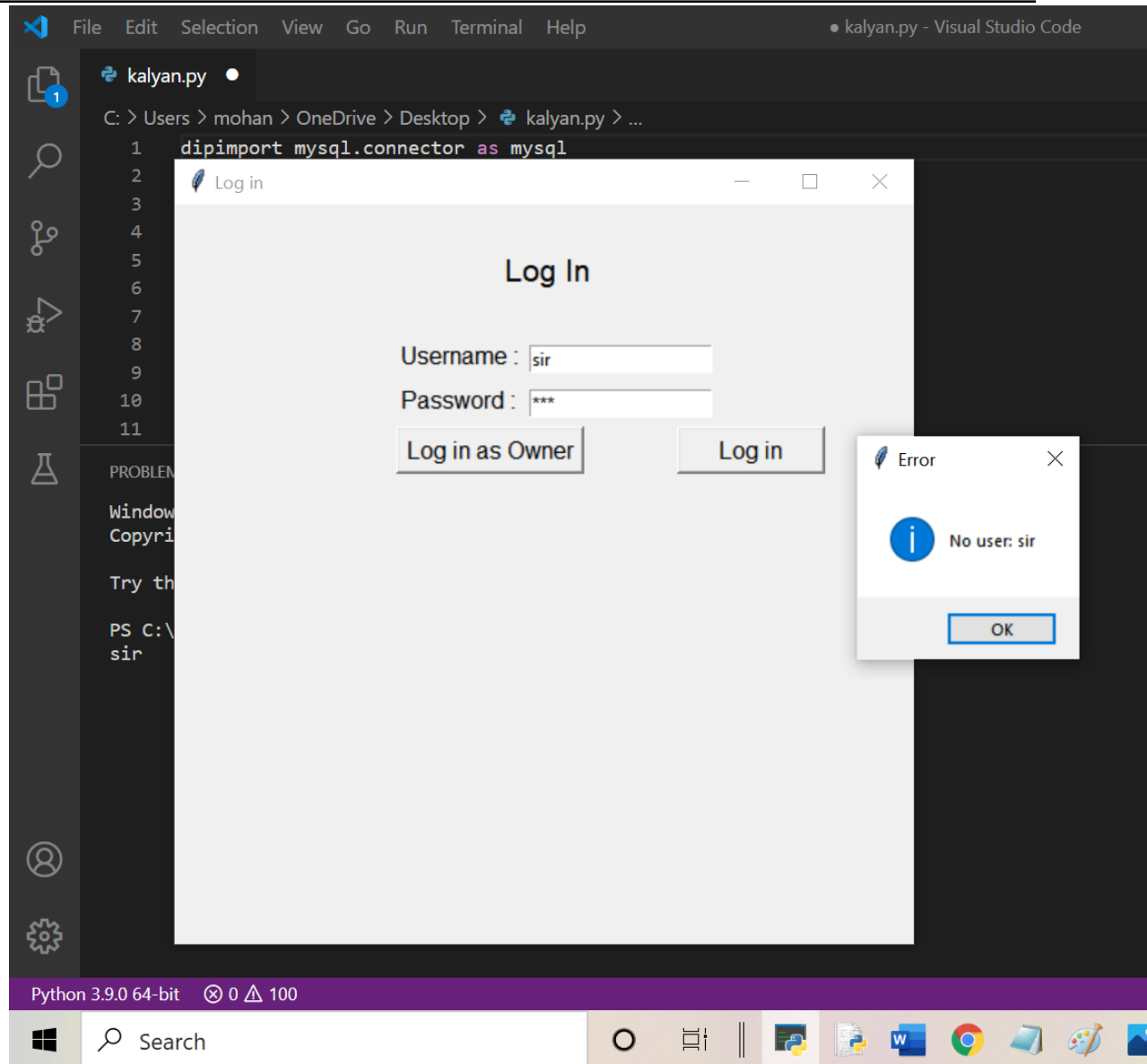


Figure 7

CHAPTER 8

CONCLUSION

Motivation of my project is to find the gym and it helps to get the good trainers of the gym without searching here and there. It makes the customer to save the time efficiency and money, without going anywhere we can join the gym by sitting in house and clicking. And we all know the pandemic is really terrible in this situation it's really tough to go out and search for the gyms. So my project helps the user to find the gym and trainers in the gym easily by one click. The customer can add the trainers whom ever they want and can remove whom ever they want according to the workout.

REFERENCE

<https://www.oreilly.com/library/view/python-in-a/0596001886/ch04s03.html#:~:text=A%20Python%20program%20accesses%20data,reference%20has%20no%20intrinsic%20type.>

<https://www.guru99.com/python-mysql-example.html>