

HEALTH CARE SERVICES

ANNAM MANIKANTA
1NH18CS024
5 A

CHAPTER 1

INTRODUCTION

Python is one of the programming software's that has picked up a huge number of users because of its explicit characteristics and features which are really helpful, compact, easy to understand and simple to an extent where the beginners can easily understand it. Python is one of many languages which is an interpreted language i.e. the interpreter executes the code line by line and debugging is very easy. It is a language which is freely available in the official page. Python is a platform independent language, therefore the code for any program need not be changed if the platform is changed. Python has a huge set of inbuilt libraries which help a programmer by reducing the time taken to create the code for certain cases and just import the libraries into the code.

We can easily connect our code to a database with the help of a connect method and data which we enter in the program or GUI will be directly stored in the database. Many commands related to the creation, insertion, updating, deletion can be implemented in the program which help reduce the work of the programmer and saves time.

One of the unique and amusing features of python is that we can inculcate a GUI in our program instead of running it in a typical command line to make the program a friendly interface between the user and the program. Many features like widgets, frames, grids, message boxes, buttons, labels, drop down lists and many more are available to design our GUI which will make the interface userfriendly.

COURSE OBJECTIVES.

- To learn how to design and program Python applications
- To learn how to build and package Python modules for reusability
- To learn how to design Object oriented programs
- To learn how to use class inheritance in Python for reusability
- To learn how to use exception handling in Python applications for error handling
- To make better understanding of lists, tuples, dictionaries in Python programs
- To understand why Python is a useful scripting language for developers

PROBLEM DEFINITION

The main theme of the project is to help the patients who are facing so difficult to choose an right hospital for right disease this project or this software helps for those people which give approximately right suggestions to them and it can help for them to find when doctor is free and to fix an appointment with them in this we will retrieve every patients data which helps for them to check previous problems and the given solutions are worth or not . It manages the smooth healthcare performance along with administrative, medical, legal and financial control. That is a cornerstone for the successful operation of the healthcare facility

OUTCOMES OF THE PROJECTWORK

The main theme of the project is to help the patients who are facing so difficult to choose an right hospital for right disease this project or this software helps for those people which give approximately right suggestions to them and it can help for them to find when doctor is free and to fix an appointment with them in this we will retrieve every patients data which helps for them to check previous problems and the given solutions are worth or not . It manages the smooth healthcare performance along with administrative, medical, legal and financial control. That is a cornerstone for the successful operation of the healthcare.

Enter the patients name:

Age:

Gender:

Appointment time:

Location:

CHAPTER 2

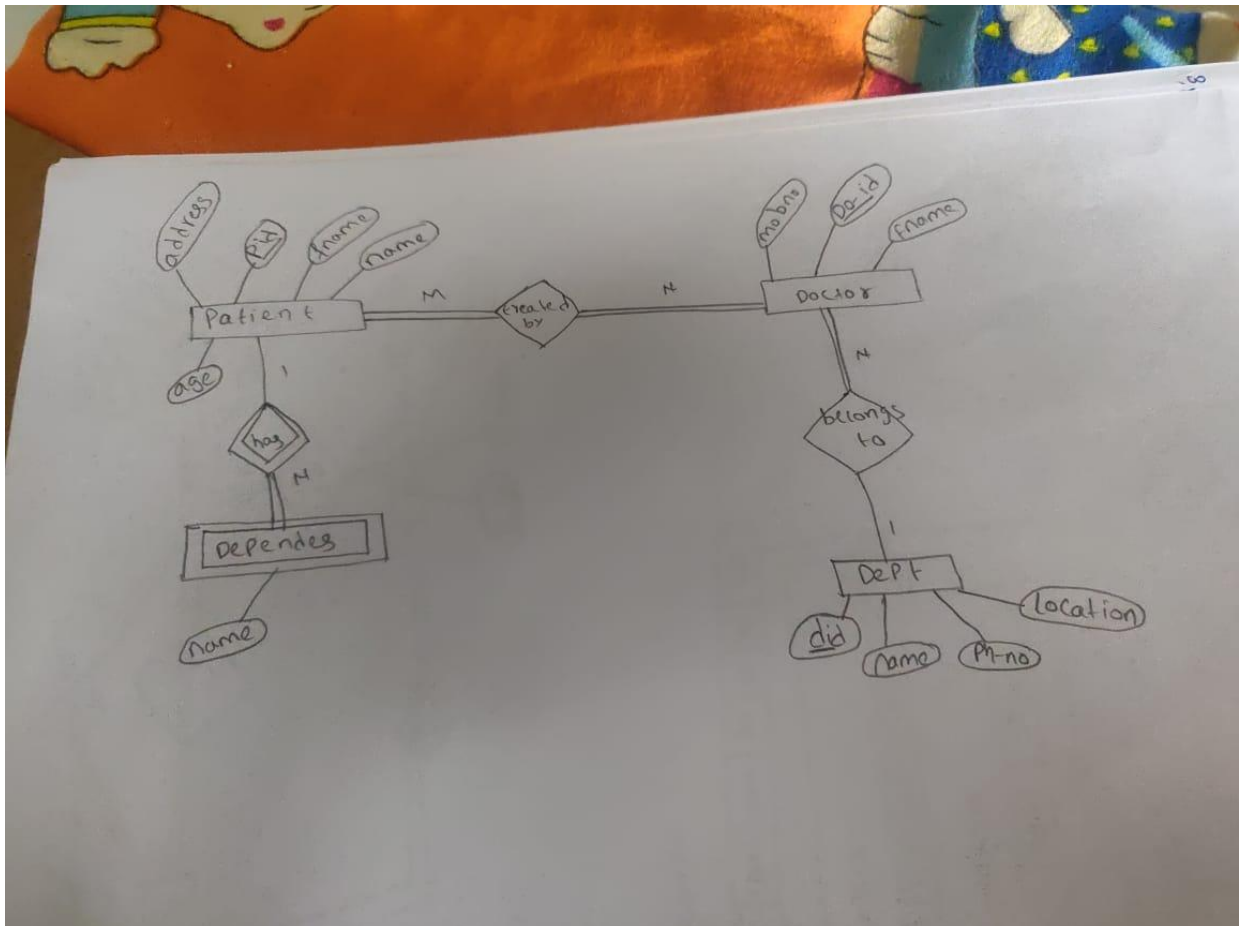
REQUIREMENTS AND DESIGN

SOFTWARE REQUIREMENTS.

- Operating system: Windows 7 or higher.
- Programming language: Python IDLE or Jupyter notebook or Pycharm
- Database manager: SQLite3 or any compatible manager.

HARDWARE REQUIREMENTS.

- Processor: Any one of the modern cores.
- RAM: 1GB or higher.
- SSD: 0.5TB or higher.

CHAPTER 3**ER MODEL****3.1 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.

3.2 KEY

Keys is a constraint used while defining attributes in a table. Keys is used to identify a row in a table. Keys plays a vital role in finding the relation between two tables. It helps you uniquely identify a row in a table by combination of one or more columns in that table. There are various keys which has various properties, one of them which is widely used is Primary key. Primary key is a unique identification of a table that is used while combining tables and it can never be NULL. In a single table more than one column can be primary key. When this primary key column is used in other table then that becomes Foreign key and its values can be NULL. Both primary and foreign key plays important role in determining relation between two tables. Unique key is also one of them which ensures that the particular column has unique values.

3.3 RELATIONSHIP AND PARTICIPATION

If one table has a foreign key that references the primary key of another table then there exists relation between two relational databases. This means one or the other way an entity is related with the other like an entity type is related with another entity with the relation "pays". Participation may be 1: 1, 1: N or M: N. One entity can relate with another table and can have partial or full participation. Here partial participation means the members of one entity may or may not be associated with the other entity

CHAPTER 4

4.1 Design Goals

SQL stands for Structured Query Language. It's a standard computer language for managing the relational database and manipulation of data. It is used to do all the operation in database like creation of schema, table, inserting, updating, deleting and retrieving rows. SQL is used by various database management system like: MySQL, oracle, vertica, Sybase, etc.

There are various SQL commands which are listed below:

DDLcommands:

DDL stands for Data Definition Language. These are the commands used for schema or the table definition manipulation but not for data. There are various DDL commands which are listed below:

CREATE:

This command is used to create a table and objects like view, triggers, assertions, etc. in database.

Syntax: CREATE TABLE <table_name> VALUES (column1, column2....);

e.g.: CREATE TABLE IF NOT EXISTS 'TEAM'(T_CODE INTEGER PRIMARY KEY,
T_NAME VARCHAR (10), CAP_NAME VARCHAR (20) NOT NULL,
USERNAME CHAR (10) NOT NULL UNIQUE, PWD VARCHAR (20) NOT
NULL, TINBOX VARCHAR(100));

DROP:

This command is used to delete table or the objects from the database.

Syntax: DROP TABLE <table_name>;

e.g.: DROP TABLE TEAM;

ALTER:

This command is used to alter the table structure like table definitions, constraints, delete columns, add columns, etc. It has various options like ADD, MODIFY, CHANGE, DELETE etc.

Syntax: ALTER TABLE <table_name><option><commands>;

e.g.: ALTER TABLE TEAM DROP COLUMN PWD;

TRUNCATE:

This command is used to delete the contents of the table including all the spaces.

Syntax: TRUNCATE TABLE <table_name>;

e.g.: TRUNCATE TABLE TEAM;

DQL commands:

DQL stands for Data Query Language for performing database queries.

SELECT:

This command is used to retrieve the data from the database. It is one of the widely used command and is complex.

Syntax: SELECT <column_name> FROM <table_name> WHERE <condition>;

e.g.: SELECT * FROM TEAM WHERE T_code=900;

DML commands:

DML stands for Data Manipulation Language and is used to manipulate the rows in the table. The rows in the table can be updated, deleted, inserted, etc.

INSERT:

This command is used to insert the tuples into the table.

Syntax: INSERT INTO <table_name><column_name> VALUES <column_values>;

e.g.: INSERT INTO TEAM (T_CODE, T_NAME, CAP_NAME, USERNAME, PWD) VALUES (900,'Ranger', 'Dipesh', 'iamdipesh', 'iamdipesh');

UPDATE:

This command is used to update the existing values of the tuple.

Variable values can be updated using this command.

Syntax: UPDATE <table_name> SET <column_name>=<values> WHERE <condition>;

e.g.: UPDATE TEAM SET PWD= 'iamdipeshstha' WHERE T_CODE=900;

DELETE:

This command is used to delete the tuples from the tables.

Syntax: DELETE FROM <table_name> WHERE <conditions>;

e.g.: DELETE FROM TEAM WHERE T_CODE=900;

TCLcommand:

TCL stands for Transaction Control Language which deals with the transaction within the database.

COMMIT:

This command is used to commit the transaction so that the previous transaction will be successfully saved into the database. Once commit is done it is not possible to rollback.

Syntax: COMMIT;

ROLLBACK:

This command is used to rollback/ undo the transaction if any error occurs.

Syntax: ROLLBACK;

Broad Standard Library:

Python has huge collection of defined libraries which makes very easy to code in python. Its library is portable and compatible with all platforms like Macintosh, UNIX, and Windows. You don't have to write your own code for each and every thing as it provides rich sets of modules and functions. It has various libraries for web browsing, regular expressions, etc.

Interpreted Language:

Python is one of the Interpreted Languages its code is executed line by line at a time. It is not required to compile our code like in other languages like java, c++, etc. which makes it easier to debug our code. The source code of python is converted into an immediate form called bytecode.

Support for GUI Programming:

Python provides various modules like PyQt, Tkinter, wxPython through which user can create Graphical User Interface for mobile applications. The most popular for creating graphical apps using python is PyQt5. Tkinter also provides all of the required options to create a beautiful user interface even Gaffer is made importing Tkinter module. This user interface can be connected to the backend using any one of the DBMS also supported by python, makes it more beautiful.

Object Oriented Programming Language:

Python is an object-oriented programming language which includes the concept of class and object. It supports all OOPs concepts like inheritance, data abstraction, polymorphism, encapsulation etc.

Frame

A frame is a container which associates other widgets. It is mainly used for grouping and organizing widgets. A bunch of labels, entry, buttons, etc. can be added into the frame and moving frame alone moves other widgets too. Various options like: bg, bd, cursor, relief, width, etc. can be used to configure frame.

Syntax: `Frame (window_name, options)`

Label:

A label is a widget which is used to display non-editable text. Label in fact is also used to display images using `PhotoImage` module. The most commonly used label is with 'text' configuration option and can change this at any time. Label makes use of many options like fg, bg, font, width, height, etc.

Syntax: `Label (window_name, options)`

We can add an image into label as below:

```
project_img=PhotoImage (file="path with file name")
```

```
project_img_label=ttk.Label(captainDashboard, image=project_img)
```

Entry:

An entry is a single line text field user can use to type anything. It's mostly used in log in form for retrieving username and password. It has a special property to hide/ encrypt the text typed by user by using "`show='*'`" option which replaces each and every letter with the specified symbol/letter (in this case every letter typed by user is encrypted with '*').

Syntax: `Entry (window_name, options)`

Button:

Button is one of the widely used widget among all in GUI with Tkinter. It is a functional widget that is clickable and on click it performs some action defined in the command option. It is used for linking two functions. They can display text or images same as labels, but also have a whole range of new options used to control their behavior.

Syntax: Button (window_name, options)

Listbox:

Listbox displays a list of contents which a user interacts with and user can accept any number of times. It looks like a column of a table that displays values in various rows. It provides option to browse, select multiple, select single through selectmode option. It also offers other variety of options like: bg, fg, font, height, width, highlightcolor, etc.

Syntax: Listbox(window_name, options)

ScrolledText:

This widget provides the feature of multiple line input field with scrollbar wherein user can type multiple lines of text. This is very much useful for typing paragraphs, letter, essays, etc. It also supports various options like: height, width, etc.

Syntax: scrolledtext.ScrolledText(window_name, options).

Chapter 5

IMPLEMENTATION

5.1 Creating a appointment

In this project im going to develop an health care services in which im creating four tables which stores the patient details and pharmacy lisit and importing the mysql connector which is used to connect the python and sql lets go into the project and lets see the code I created appointment table also which is used to store the patient details and I created appointment page by using GUI by importing tkinter and I used radio buttons to choose the gender in the appointment I had created the link to database which is located in mysql with the help of mycursor and I inserted the values into the database with the help of mycursor insert into command based on that im able to enter the values into the database

```
def add_appointment(self):
    # getting the user inputs
    self.val1 = self.name_ent.get()
    self.val2 = self.age_ent.get()
    self.val3 = self.gender_ent.get()
    self.val4 = self.location_ent.get()
    self.val5 = self.time_ent.get()
    self.val6 = self.phone_ent.get()
    self.val7 = self.time1_ent.get()

    # checking if the user input is empty
    if self.val1 == "" or self.val2 == "" or self.val3 == "" or self.val4 == "" or self.val5 == "":
        messagebox.showinfo("Warning", "Please Fill Up All Boxes")
    else:
        # now we add to the database
        mydb=mysql.connector.connect(host="localhost",
                                     user="root",
                                     password="manikanta",
                                     database= 'appointment'
                                    )
        mycursor=mydb.cursor()
        mycursor1 = mydb.cursor()
        mycursor.execute("insert into book values(%s,%s,%s,%s,%s,%s)",(self.val1, self.val2,
self.val3, self.val4, self.val5, self.val6))
        mycursor1.execute("insert into appoint values(%s,%s)", (self.val5, self.val7))
        #mycursor.execute("INSERT INTO 'details' (name,
age, gender, location, scheduled_time, phone) VALUES(?, ?, ?, ?, ?, ?)")
```

Health care services

```
#a=(sql, (self.val1, self.val2, self.val3, self.val4, self.val5, self.val6))
#mydb.commit()

#db="CREATE TABLE 'details'(name varchar(10), age int PRIMARY KEY, gender
varchar(10), location varchar (10), scheduled_time date, phone int"

#mycursor.execute(db,val)

mydb.commit()

messagebox.showinfo("Success", "Appointment for " + str(self.val1) + " has been
created")
```

In the above I created the add appointment function which is used to create an appointment in health care services project in which we need to fill the patient details like patient name ,age,appointment date,time,location these are the things which are the need to fill by the Patient.

Creating a table for appointment:

```
create table appoint(app_date date,app_time varchar(10));
```

Appointment table is used to store the patient data in sql server and which is helpful to the patients to view their old records and it will be easy to doctor for diagnose the patient and know the problem easily. mydb=mysql.connector.connect(host="localhost", user="root", password="manikanta", database= 'appointment' by the help of these lines of code im able to connect the data base to sql and then I created tables to store the values

5.2 Creating a Display

```
def display(self):
    self.val1 = self.name_ent.get()
    self.val2 = self.age_ent.get()
    self.val3 = self.gender_ent.get()
    self.val4 = self.location_ent.get()
    self.val5 = self.time_ent.get()
    self.val6 = self.phone_ent.get()

    mydb = mysql.connector.connect(host="localhost",
                                   user="root",
                                   password="manikanta",
                                   database='appointment'
                                   )

    mycursor = mydb.cursor()
    mycursor.execute("select * from book")
```

```
master=Tk()
master.geometry('500x500')
master.title('DETAILS')

Label1 = Label(master, text="NAME", width=10,font='green')
Label1.grid(row=0, column=0)
Label2 = Label(master, text="AGE", width=10,font='green')
Label2.grid(row=0, column=1)
Label3 = Label(master, text="GENDER", width=10,font='green')
Label3.grid(row=0, column=2)
Label1 = Label(master, text="LOCATION", width=10,font='green')
Label1.grid(row=0, column=3)
Label1 = Label(master, text="SCH DATE", width=10,font='green')
Label1.grid(row=0, column=4)
Label1 = Label(master, text="PHONE", width=15,font='green')
Label1.grid(row=0, column=5)

for index, dat in enumerate(mycursor):
    Label(master, text=dat[0]).grid(row=index + 1, column=0)
    Label(master, text=dat[1]).grid(row=index + 1, column=1)
    Label(master, text=dat[2]).grid(row=index + 1, column=2)
    Label(master, text=dat[3]).grid(row=index + 1, column=3)
    Label(master, text=dat[4]).grid(row=index + 1, column=4)
    Label(master, text=dat[5]).grid(row=index + 1, column=5)
```

The above function is display which is used to display all patient details and help for the admin to view how many patients are filling the application form to book the slot and how to improve our facilities to improve to attract the customers

In this project we use Graphical user interface which will be easy for customers to book their slots for graphically user interface Im importing tkinter package from the python inbuilt libraies and which is used to create GUI in this we use lables frames and buttons which are used to look the site good and better and attracting for the patient and it will be easy to book a slot by using these tings in the project let us see below what and all are used in the creation of GUI In this projected I created buttons for add appointment ,patient name,age,location, appointment time, date Frames are used to create the clours to GUI which is known as Graphical user interface and lables arealso usedto lable windows are used to display the content present in the page.

```
# creating the frames in the master
self.left = Frame(master, width=1200, height=720, bg='lightblue')
self.left.pack(side=LEFT)

self.right = Frame(master, width=400, height=720, bg='steelblue')
self.right.pack(side=RIGHT)
```

```
# labels for the window
self.heading = Label(self.left, text="Helath Care Services", font=('arial 40 bold'), fg='black',
                    bg='lightblue')
self.heading.place(x=0, y=0)
# patients name
self.name = Label(self.left, text="Patient's Name", font=('arial 18 bold'), fg='black',
bg='lightblue')
self.name.place(x=0, y=100)

# age
self.age = Label(self.left, text="Age", font=('arial 18 bold'), fg='black', bg='lightblue')
self.age.place(x=0, y=140)

# gender
self.gender = Label(self.left, text="Gender", font=('arial 18 bold'), fg='black', bg='lightblue')
self.gender.place(x=0, y=180)

# location
self.location = Label(self.left, text="Location", font=('arial 18 bold'), fg='black',
bg='lightblue')
self.location.place(x=0, y=220)

# appointment time
self.time = Label(self.left, text="Appointment Date", font=('arial 18 bold'), fg='black',
bg='lightblue')
self.time.place(x=0, y=260)

#time
self.time1 = Label(self.left, text="Time", font=('arial 18 bold'), fg='black', bg='lightblue')
self.time1.place(x=0, y=300)

# phone
self.phone = Label(self.left, text="Phone Number", font=('arial 18 bold'), fg='black',
bg='lightblue')
self.phone.place(x=0, y=340)
```


5.3 Creating a Delete

I have created a delete button by the help of buttons and the frames which will be helpful for the admin and I created a delete function which will be helpful for admin to delete the patients data when they are cured and the space will be cleared and searching will be easy for the admin to search the patients data it will be very helpful for the admin lets the coding part below

```
def delete(self):
    master = Tk()
    master.geometry('1200x720+0+0')
    master.title('Del..')
    self.left = Frame(master, width=1200, height=720, bg='lightgreen')
    self.left.pack(side=LEFT)
    self.heading = Label(self.left, text=".....Deleting.....", font=('arial 40 bold'), fg='black',
                        bg='lightgreen')
    self.heading.place(x=400, y=100)

    self.s2 = Label(self.left, text="Patient's Name", font=('arial 18 bold'), fg='black',
                    bg='lightblue')
    self.s2.place(x=50, y=300)
    self.s2_ent = Entry(self.left, width=30)
    self.s2_ent.place(x=300, y=305)

    self.submit = Button(self.left, text="Submit", width=20, height=2, bg='steelblue',
                        command=self.submit3)
    self.submit.place(x=450, y=350)
```

5.4 Creating admin and tables

I created the tables for admin and medicines and display which are helpful to store the data in the sql server lets see the tables below:

```
create table book(name varchar(10) primary key,age int,gender varchar(10),location
varchar(10),scheduled_time date,phone varchar(30));
create table login(id varchar(10),password varchar(10));
create table medicine(mid varchar(10),mname varchar(10),manf varchar(10),exp
varchar(10),pname varchar(10)); and I created admin login page in which only admin can login
with his password it is designed by general basic if else conditions and the admin button is
created by the frames and the tkinter now lets see the coding part
```

```
def admin(self):
    master = Tk()
    master.geometry('500x500')
    master.title('log')
```

```
self.master = master
self.master.title("Login System")
self.master.geometry('900x500')
self.master.config(bg='crimson')
self.frame = Frame(self.master, bg='crimson')
self.frame.pack()

self.Username = StringVar()
self.Password = StringVar()

self.lblTitle = Label(self.frame, text="ADMIN LOGIN PAGE", font=("arial", 50, 'bold'),
                      bg='crimson', fg='cornsilk')
self.lblTitle.grid(row=0, column=0, columnspan=2, pady=20)

#
=====

self.LoginFrame1 = LabelFrame(self.frame, width=1250, height=300, fg='cornsilk',
text="Login",
                                font=('arial', 20, 'bold'), relief='ridge', bg='crimson', bd=15)
self.LoginFrame1.grid(row=1, column=0)

self.LoginFrame2 = LabelFrame(self.frame, width=900, height=130, fg='cornsilk',
text="Log",
                                font=('arial', 20, 'bold'), relief='ridge', bg='crimson', bd=15)
self.LoginFrame2.grid(row=2, column=0)

#
=====

self.lblUsername = Label(self.LoginFrame1, text="Username", font=("arial", 15, 'bold'),
bd=10, bg='crimson',
                        fg='cornsilk')
self.lblUsername.grid(row=0, column=0)

self.txtUsername = Entry(self.LoginFrame1, font=("arial", 15, 'bold'), bd=7,
textvariable=self.Username,
                        width=23)
self.txtUsername.grid(row=0, column=1, padx=120)

self.lblPassword = Label(self.LoginFrame1, text="Password", font=("arial", 15, 'bold'),
bd=10, bg='crimson',
                        fg='cornsilk')
self.lblPassword.grid(row=1, column=0)
```

```
self.txtPassword = Entry(self.LoginFrame1, font=("arial", 15, 'bold'), show='*', bd=7,
                          textvariable=self.Password, width=23)
self.txtPassword.grid(row=1, column=1, columnspan=2, pady=30)

#
=====

self.btnLogin = Button(self.LoginFrame2, text='Login', width=15, font=('arial', 20, 'bold'),
                        bg='crimson',
                        fg='cornsilk', command=self.Login_System)
self.btnLogin.grid(row=3, column=0, pady=10, padx=8)

#self.btnReset = Button(self.LoginFrame2, text='Reset', width=15, font=('arial', 20, 'bold'),
                        bg='crimson',
                        # fg='cornsilk', command=self.iReset)
#self.btnReset.grid(row=3, column=1, pady=10, padx=8)

self.btnExit = Button(self.LoginFrame2, text='Exit', width=15, font=('arial', 20, 'bold'),
                       bg='crimson',
                       fg='cornsilk', command=self.iExit)
self.btnExit.grid(row=3, column=2, pady=10, padx=8)

#
=====

def Login_System(self):
    self.user = self.txtUsername.get()
    self.pas = self.txtPassword.get()
    if (self.user == str('mani') and self.pas == str(1234)):
        messagebox.showinfo("yaaa", "entered")

        mydb = mysql.connector.connect(host="localhost",
                                       user="root",
                                       password="manikanta",
                                       database='appointment'
                                       )
        mycursor = mydb.cursor()
        mycursor.execute("insert into login values(%s,%s)", (self.user, self.pas))
        mydb.commit()
        b.abt()

    elif (self.user == str(12345678) and self.pas == str(123456)):
```

```
tkinter.messagebox.askyesno("Admin", "login")
#self.Login_details()
print("yes yaaa")
else:
    import tkinter
    tkinter.messagebox.askyesno("Admin", "Invalid Login details")
    self.Username.set("")
    self.Password.set("")
```

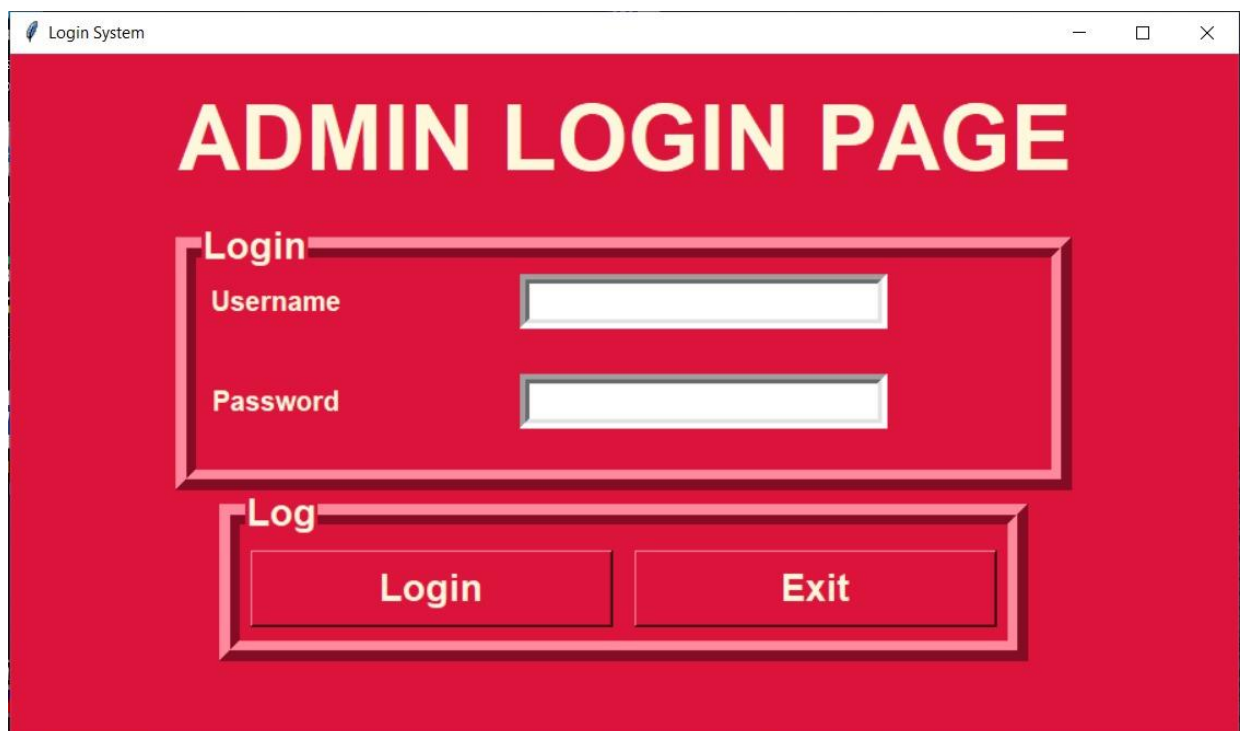
CHAPTER 9

Outputs

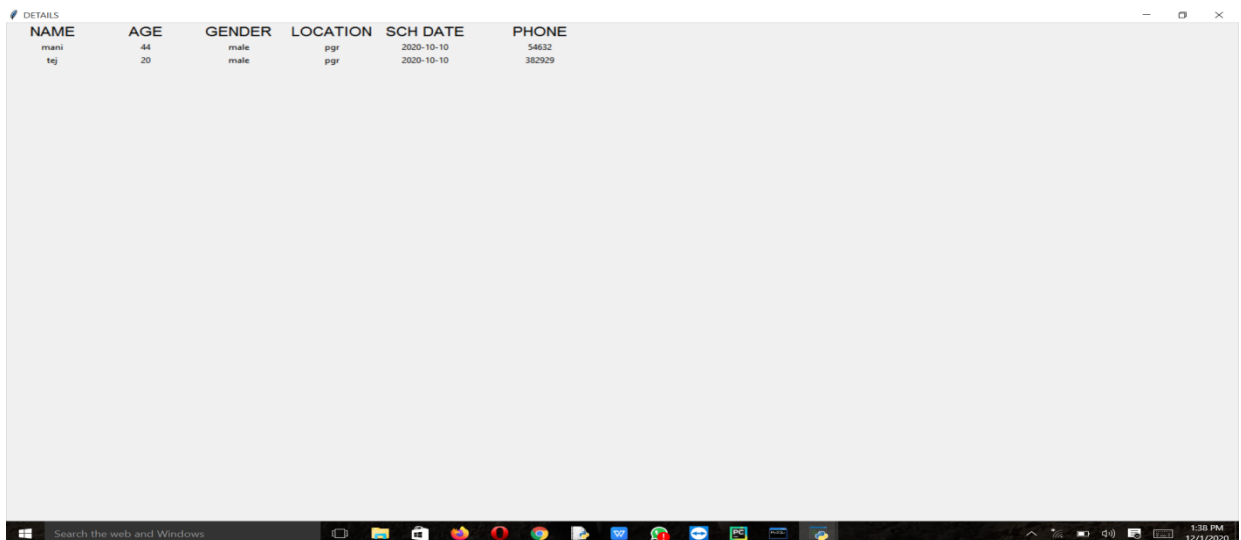
Apooointment slot page:



Admin page



Display page:

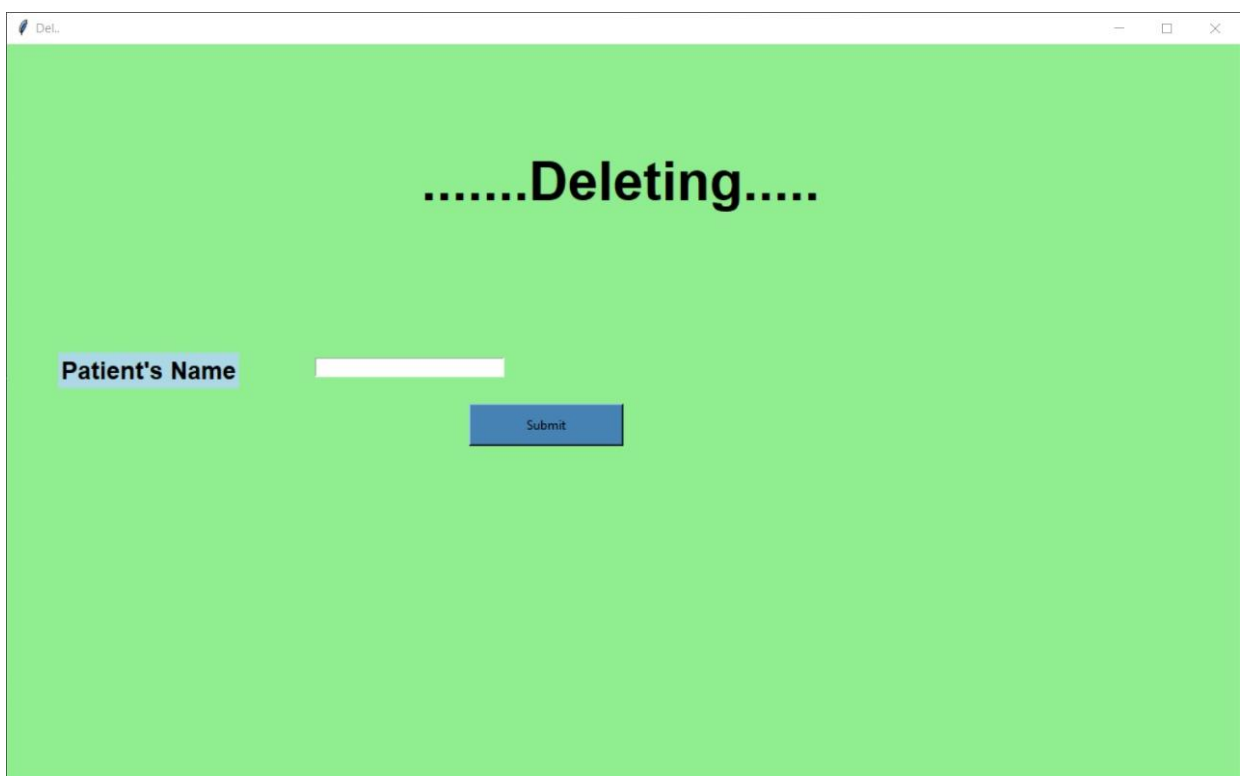


The screenshot shows a web browser window with the title 'DETAILS'. Inside the window is a table with the following data:

NAME	AGE	GENDER	LOCATION	SCH DATE	PHONE
mani	44	male	pgr	2020-10-10	54632
tej	20	male	pgr	2020-10-10	382929

The browser's taskbar is visible at the bottom, showing various application icons and the system clock indicating 1:38 PM on 12/1/2020.

Delete page



The screenshot shows a web browser window with the title 'DeL.'. The background is a solid light green color. In the center, the text '.....Deleting.....' is displayed in a large, bold, black font. Below this text, there is a form with a label 'Patient's Name' in a blue box, followed by a white text input field. To the right of the input field is a blue button with the text 'Submit' in white.

Updating patient details:



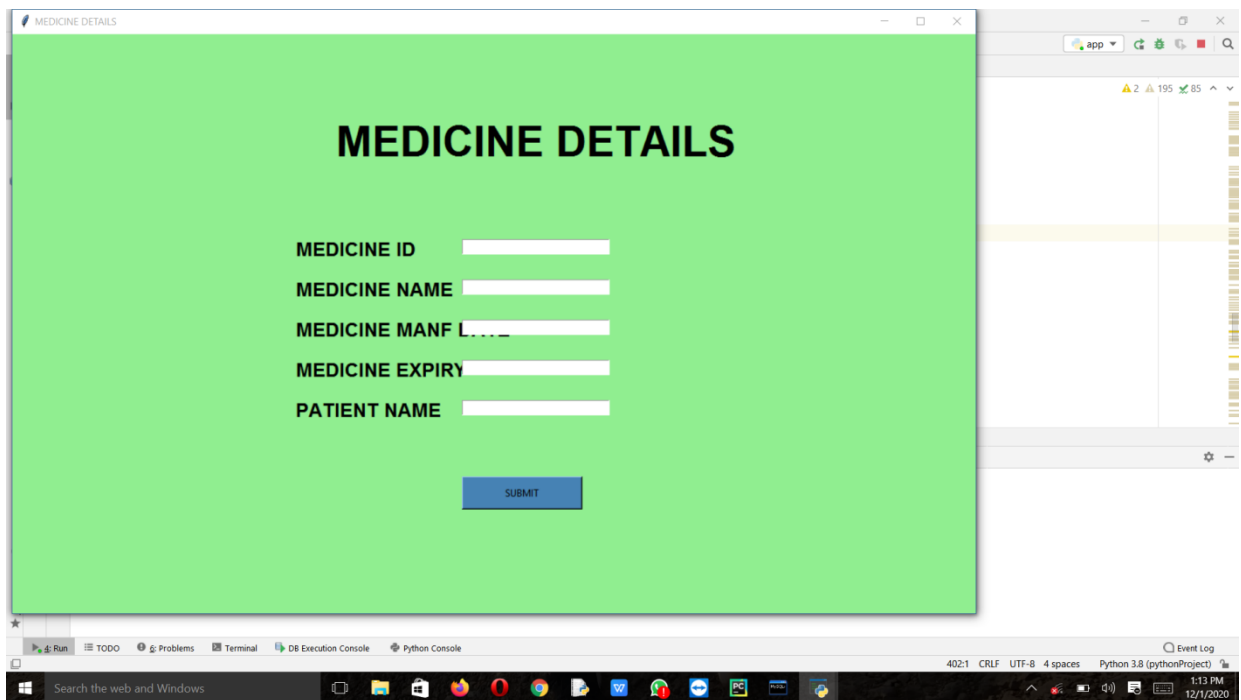
UPDATE

.....Updating.....

Patient's Name

Submit

Medical store



MEDICINE DETAILS

MEDICINE ID

MEDICINE NAME

MEDICINE MANF L...

MEDICINE EXPIRY

PATIENT NAME

SUBMIT

Mysql tables:

```
mysql> use appointment;
Database changed
mysql> select * from book;
+-----+-----+-----+-----+-----+-----+
| name | age | gender | location | scheduled_time | phone |
+-----+-----+-----+-----+-----+-----+
| abt | 22 | male | miuyg | 2020-09-09 | 1234567897 |
| abtt | 21 | male | miuyg | 2020-09-09 | 1234567897 |
| arjun | 17 | male | madndz | 2020-09-08 | 75567 |
| chandana | 21 | male | sdfv | 2020-02-21 | 2345 |
| chandu | 23 | 2 | national | 2020-11-21 | 8765 |
| mahesh | 22 | 1 | ngos | 2020-10-21 | 654321 |
| sita | 29 | 2 | mig-123 | 2020-04-13 | 98765432 |
| teja | 22 | 1 | ngos | 2020-10-21 | 654321 |
+-----+-----+-----+-----+-----+-----+
8 rows in set (0.01 sec)

mysql> select * from appoint;
+-----+-----+
| app_date | app_time |
+-----+-----+
| 2020-10-21 | 2 |
| 2020-10-21 | 4am |
| 2020-11-21 | 3pm |
| 2020-09-12 | 12 |
| 2020-09-08 | 12pm |
| 2020-02-21 | 2 |
+-----+-----+
6 rows in set (0.01 sec)

ERROR:
No query specified

mysql> select * from medicine;
+-----+-----+-----+-----+-----+
| mid | mname | manf | exp | pname |
+-----+-----+-----+-----+-----+
| 321 | dm | ghj | thjj | tejaa |
| m12 | nammmw | ghj | ghju | tejaa |
| m1234 | asdf | tesdfdg | sfzdxbgm | teja |
+-----+-----+-----+-----+-----+
3 rows in set (0.01 sec)

mysql> select * from login;
+-----+-----+
| id | password |
+-----+-----+
| teja | 9999 |
| teja | 9999 |
| teja | 9999 |
| teja | 9999 |
| teja | 9999 |
| mahesh | 1234 |
| mahesh | 1234 |
+-----+-----+
```


CHAPTER 6

CONCLUSION

Health care services Database is a Python application which will be very helpful for the Common people who are to willing to vist the hospital they need not to wait in lines they can fill their appointment through this project this so helpful hospital and medical stores to check their patinets how they are feeling how their health is going on and they can easily treat them and they can give can select the of booking the appointment and they can retrieve the data and ifthey want they can cancel the appointment easily by clicking on dlt button and it will help all medical and pharmcy sector to track the patients health condition by this software or by this programe.

REFERENCES

1. <https://www.geeksforgeeks.org/python-gui-tkinter/>
2. https://www.python-course.eu/python_tkinter.php
3. <https://www.youtube.com/watch?v=QXeEoD0pB3E&list=PLsyebzWxl7poL9JTVyndKe62ieoN-MZ3>