CERTIFICATE

This is to certify that the project titled **Patient Information** is an authentic work of Kendriya Vidyalaya Ambarnath School (CBSE).

This work has been carried out under my supervision and guidance.

Mast……from class: XII CBSE Roll NO: **……** was able to prepare the project in a systematic manner.

Ms. Aparna Dhirde
Dept. of computer

External Examiner

Principal
KV Ambarnath

ACKNOWLEGEMENT

With profound sense of graduate and regard, I express my sincere thanks to my guide mentor Ms. Aparna Dhirde for her valuable guidance and the confidence she instilled in us, which helped me in successful completion of this project report. Without her help, this project would have been a distant affair. Her through understanding of the subject and the professional guidance is needed of immense help to me.

I am also thankful to the faculty members of my school who cooperated with us and gave me their valuable time.

Name:- Mast. ……..

Date:-

SYSTEM COVERS FOLLOWING THINGS

Python:-

Python is a general-purpose, high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It has fewer syntactical constructions than other languages.

Characteristics of Python

Following are important characteristics of **Python Programming** −

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Applications of Python

As mentioned before, Python is one of the most widely used language over the web. I'm going to list few of them here:

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.

- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** − Python provides interfaces to all major commercial databases.
- **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** − Python provides a better structure and support for large programs than shell scripting.

It contains so many modules as well as libraries like

1. Python Satndard Library which contains modules like Math, Cmath, Random, URLlib ,
2. Numpy,
3. SciPyplot,
4. Tkinter

5. Matplotlib.

As well as various data structures like list, queue, stack.

Mysql:

Databse is defined as collection of interrelated data stored together to serve multiple applications. And MySQL is the most popular Open Source Relational SQL Database Management System. MySQL is one of the best RDBMS being used for developing various web-based software applications.

We use SQL in mysql in order to access data, manipulate data.
SQL stands for **Structured Query Language SQL** is a database computer language designed for the retrieval and management of data in a relational database. And has clearly established itself as a standard relational database language.

SQL is divided into Data Definition Language (DDL), Data Manipulation Language(DML), Transaction Control Language(TCL).

SOURCE CODE –PAITENT DATABASE

```
import tkinter
import tkinter.ttk
import tkinter.messagebox
import sqlite3
import mysql.connector

class Database:
def __init__(self):
self.dbConnection = sqlite3.connect("dbFile.db")
self.dbCursor = self.dbConnection.cursor()
self.dbCursor.execute("CREATE TABLE IF NOT EXISTS patient_info
(id PRIMARYKEY text, fName text, lName text, dob text, mob text,
yob text, gender text, address text, phone text, email text, bloodGroup
text, history text, doctor text)")

def __del__(self):
```

```
self.dbCursor.close()
self.dbConnection.close()

def Insert(self, id, fName, lName, dob, mob, yob, gender, address, phone, email, bloodGroup, history, doctor):
self.dbCursor.execute("INSERT INTO patient_info VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)", (id, fName, lName, dob, mob, yob, gender, address, phone, email, bloodGroup, history, doctor))
self.dbConnection.commit()

def Update(self, fName, lName, dob, mob, yob, gender, address, phone, email, bloodGroup, history, doctor, id):
self.dbCursor.execute("UPDATE patient_info SET fName = ?, lName = ?, dob = ?, mob = ?, yob = ?, gender = ?, address = ?, phone = ?, email = ?, bloodGroup = ?, history = ?, doctor = ? WHERE id = ?", (fName, lName, dob, mob, yob, gender, address, phone, email, bloodGroup, history, doctor, id))
self.dbConnection.commit()

def Search(self, id):
self.dbCursor.execute("SELECT * FROM patient_info WHERE id = ?", (id, ))
searchResults = self.dbCursor.fetchall()
return searchResults
```

```
def Delete(self, id):
self.dbCursor.execute("DELETE FROM patient_info WHERE id = ?", (id, ))
self.dbConnection.commit()
def Display(self):
self.dbCursor.execute("SELECT * FROM patient_info")
records = self.dbCursor.fetchall()
return records

class Values:
def Validate(self, id, fName, lName, phone, email, history, doctor):
if not (id.isdigit() and (len(id) == 3)):
return "id"
elif not (fName.isalpha()):
return "fName"
elif not (lName.isalpha()):
return "lName"
elif not (phone.isdigit() and (len(phone) == 10)):
return "phone"
elif not (email.count("@") == 1 and email.count(".") > 0):
return "email"
elif not (history.isalpha()):
return "history"
```

```
elif not (doctor.isalpha()):
return "doctor"
else:
return "SUCCESS"

class InsertWindow:
def __init__(self):
self.window = tkinter.Tk()
self.window.wm_title("Insert data")

# Initializing all the variables
self.id = tkinter.StringVar()
self.fName = tkinter.StringVar()
self.lName = tkinter.StringVar()
self.address = tkinter.StringVar()
self.phone = tkinter.StringVar()
self.email = tkinter.StringVar()
self.history = tkinter.StringVar()
self.doctor = tkinter.StringVar()

self.genderList = ["Male", "Female", "Transgender", "Other"]
self.dateList = list(range(1, 32))
```

```
self.monthList = ["January", "February", "March", "April", "May",
"June", "July", "August", "September", "October", "November",
"December"]

self.yearList = list(range(1900, 2020))

self.bloodGroupList = ["A+", "A-", "B+", "B-", "O+", "O-", "AB+",
"AB-"]


# Labels

tkinter.Label(self.window, text = "Patient ID",  width = 25).grid(pady =
5, column = 1, row = 1)

tkinter.Label(self.window, text = "First Name",  width = 25).grid(pady =
5, column = 1, row = 2)

tkinter.Label(self.window, text = "Last Name",  width = 25).grid(pady =
5, column = 1, row = 3)

tkinter.Label(self.window, text = "D.O.B",  width = 25).grid(pady = 5,
column = 1, row = 4)

tkinter.Label(self.window, text = "M.O.B",  width = 25).grid(pady = 5,
column = 1, row = 5)

tkinter.Label(self.window, text = "Y.O.B",  width = 25).grid(pady = 5,
column = 1, row = 6)

tkinter.Label(self.window, text = "Gender",  width = 25).grid(pady = 5,
column = 1, row = 7)

tkinter.Label(self.window, text = "Home Address",  width =
25).grid(pady = 5, column = 1, row = 8)
```

```
tkinter.Label(self.window, text = "Phone Number",  width = 25).grid(pady = 5, column = 1, row = 9)

tkinter.Label(self.window, text = "Email ID",  width = 25).grid(pady = 5, column = 1, row = 10)

tkinter.Label(self.window, text = "Blood Group",  width = 25).grid(pady = 5, column = 1, row = 11)

tkinter.Label(self.window, text = "Patient History",  width = 25).grid(pady = 5, column = 1, row = 12)

tkinter.Label(self.window, text = "Doctor",  width = 25).grid(pady = 5, column = 1, row = 13)


# Fields

# Entry widgets

self.idEntry = tkinter.Entry(self.window,  width = 25, textvariable = self.id)

self.fNameEntry = tkinter.Entry(self.window,  width = 25, textvariable = self.fName)

self.lNameEntry = tkinter.Entry(self.window,  width = 25, textvariable = self.lName)

self.addressEntry = tkinter.Entry(self.window,  width = 25, textvariable = self.address)

self.phoneEntry = tkinter.Entry(self.window,  width = 25, textvariable = self.phone)

self.emailEntry = tkinter.Entry(self.window,  width = 25, textvariable = self.email)
```

```
self.historyEntry = tkinter.Entry(self.window,  width = 25, textvariable = self.history)
self.doctorEntry = tkinter.Entry(self.window,  width = 25, textvariable = self.doctor)


self.idEntry.grid(pady = 5, column = 3, row = 1)
self.fNameEntry.grid(pady = 5, column = 3, row = 2)
self.lNameEntry.grid(pady = 5, column = 3, row = 3)
self.addressEntry.grid(pady = 5, column = 3, row = 8)
self.phoneEntry.grid(pady = 5, column = 3, row = 9)
self.emailEntry.grid(pady = 5, column = 3, row = 10)
self.historyEntry.grid(pady = 5, column = 3, row = 12)
self.doctorEntry.grid(pady = 5, column = 3, row = 13)


# Combobox widgets
self.dobBox = tkinter.ttk.Combobox(self.window, values = self.dateList, width = 20)
self.mobBox = tkinter.ttk.Combobox(self.window, values = self.monthList, width = 20)
self.yobBox = tkinter.ttk.Combobox(self.window, values = self.yearList, width = 20)
self.genderBox = tkinter.ttk.Combobox(self.window, values = self.genderList, width = 20)
```

```
self.bloodGroupBox = tkinter.ttk.Combobox(self.window, values =
self.bloodGroupList, width = 20)


self.dobBox.grid(pady = 5, column = 3, row = 4)
self.mobBox.grid(pady = 5, column = 3, row = 5)
self.yobBox.grid(pady = 5, column = 3, row = 6)
self.genderBox.grid(pady = 5, column = 3, row = 7)
self.bloodGroupBox.grid(pady = 5, column = 3, row = 11)


# Button widgets
tkinter.Button(self.window, width = 20, text = "Insert", command =
self.Insert).grid(pady = 15, padx = 5, column = 1, row = 14)
tkinter.Button(self.window, width = 20, text = "Reset", command =
self.Reset).grid(pady = 15, padx = 5, column = 2, row = 14)
tkinter.Button(self.window, width = 20, text = "Close", command =
self.window.destroy).grid(pady = 15, padx = 5, column = 3, row = 14)


self.window.mainloop()
def Insert(self):
self.values = Values()
self.database = Database()
self.test = self.values.Validate(self.idEntry.get(), self.fNameEntry.get(),
self.lNameEntry.get(), self.phoneEntry.get(), self.emailEntry.get(),
self.historyEntry.get(), self.doctorEntry.get())
```

```
if (self.test == "SUCCESS"):
self.database.Insert(self.idEntry.get(), self.fNameEntry.get(),
self.lNameEntry.get(), self.dobBox.get(), self.mobBox.get(),
self.yobBox.get(), self.genderBox.get(), self.addressEntry.get(),
self.phoneEntry.get(), self.emailEntry.get(), self.bloodGroupBox.get(),
self.historyEntry.get(), self.doctorEntry.get())
tkinter.messagebox.showinfo("Inserted data", "Successfully inserted the
above data in the database")
else:
self.valueErrorMessage = "Invalid input in field " + self.test
tkinter.messagebox.showerror("Value Error", self.valueErrorMessage)


def Reset(self):
self.idEntry.delete(0, tkinter.END)
self.fNameEntry.delete(0, tkinter.END)
self.lNameEntry.delete(0, tkinter.END)
self.dobBox.set("")
self.mobBox.set("")
self.yobBox.set("")
self.genderBox.set("")
self.addressEntry.delete(0, tkinter.END)
self.phoneEntry.delete(0, tkinter.END)
self.emailEntry.delete(0, tkinter.END)
self.bloodGroupBox.set("")
```

```
self.historyEntry.delete(0, tkinter.END)
self.doctorEntry.delete(0, tkinter.END)

class UpdateWindow:
def __init__(self, id):
self.window = tkinter.Tk()
self.window.wm_title("Update data")

# Initializing all the variables
self.id = id

self.fName = tkinter.StringVar()
self.lName = tkinter.StringVar()
self.address = tkinter.StringVar()
self.phone = tkinter.StringVar()
self.email = tkinter.StringVar()
self.history = tkinter.StringVar()
self.doctor = tkinter.StringVar()

self.genderList = ["Male", "Female", "Transgender", "Other"]
self.dateList = list(range(1, 32))
```

```
self.monthList = ["January", "February", "March", "April", "May",
"June", "July", "August", "September", "October", "November",
"December"]

self.yearList = list(range(1900, 2020))

self.bloodGroupList = ["A+", "A-", "B+", "B-", "O+", "O-", "AB+",
"AB-"]


# Labels

tkinter.Label(self.window, text = "Patient ID",  width = 25).grid(pady =
5, column = 1, row = 1)

tkinter.Label(self.window, text = id,  width = 25).grid(pady = 5, column
= 3, row = 1)

tkinter.Label(self.window, text = "First Name",  width = 25).grid(pady =
5, column = 1, row = 2)

tkinter.Label(self.window, text = "Last Name",  width = 25).grid(pady =
5, column = 1, row = 3)

tkinter.Label(self.window, text = "D.O.B",  width = 25).grid(pady = 5,
column = 1, row = 4)

tkinter.Label(self.window, text = "M.O.B",  width = 25).grid(pady = 5,
column = 1, row = 5)

tkinter.Label(self.window, text = "Y.O.B",  width = 25).grid(pady = 5,
column = 1, row = 6)

tkinter.Label(self.window, text = "Gender",  width = 25).grid(pady = 5,
column = 1, row = 7)
```

```
tkinter.Label(self.window, text = "Home Address",  width =
25).grid(pady = 5, column = 1, row = 8)

tkinter.Label(self.window, text = "Phone Number",  width =
25).grid(pady = 5, column = 1, row = 9)

tkinter.Label(self.window, text = "Email ID",  width = 25).grid(pady = 5,
column = 1, row = 10)

tkinter.Label(self.window, text = "Blood Group",  width = 25).grid(pady
= 5, column = 1, row = 11)

tkinter.Label(self.window, text = "Patient History",  width =
25).grid(pady = 5, column = 1, row = 12)

tkinter.Label(self.window, text = "Doctor",  width = 25).grid(pady = 5,
column = 1, row = 13)


# Set previous values

self.database = Database()

self.searchResults = self.database.Search(id)


tkinter.Label(self.window, text = self.searchResults[0][1],  width =
25).grid(pady = 5, column = 2, row = 2)

tkinter.Label(self.window, text = self.searchResults[0][2],  width =
25).grid(pady = 5, column = 2, row = 3)

tkinter.Label(self.window, text = self.searchResults[0][3],  width =
25).grid(pady = 5, column = 2, row = 4)

tkinter.Label(self.window, text = self.searchResults[0][4],  width =
25).grid(pady = 5, column = 2, row = 5)
```

```
tkinter.Label(self.window, text = self.searchResults[0][5],  width = 25).grid(pady = 5, column = 2, row = 6)

tkinter.Label(self.window, text = self.searchResults[0][6],  width = 25).grid(pady = 5, column = 2, row = 7)

tkinter.Label(self.window, text = self.searchResults[0][7],  width = 25).grid(pady = 5, column = 2, row = 8)

tkinter.Label(self.window, text = self.searchResults[0][8],  width = 25).grid(pady = 5, column = 2, row = 9)

tkinter.Label(self.window, text = self.searchResults[0][9],  width = 25).grid(pady = 5, column = 2, row = 10)

tkinter.Label(self.window, text = self.searchResults[0][10],  width = 25).grid(pady = 5, column = 2, row = 11)

tkinter.Label(self.window, text = self.searchResults[0][11],  width = 25).grid(pady = 5, column = 2, row = 12)

tkinter.Label(self.window, text = self.searchResults[0][12],  width = 25).grid(pady = 5, column = 2, row = 13)


# Fields

# Entry widgets

self.fNameEntry = tkinter.Entry(self.window,  width = 25, textvariable = self.fName)

self.lNameEntry = tkinter.Entry(self.window,  width = 25, textvariable = self.lName)

self.addressEntry = tkinter.Entry(self.window,  width = 25, textvariable = self.address)
```

```
self.phoneEntry = tkinter.Entry(self.window,  width = 25, textvariable = self.phone)
self.emailEntry = tkinter.Entry(self.window,  width = 25, textvariable = self.email)
self.historyEntry = tkinter.Entry(self.window,  width = 25, textvariable = self.history)
self.doctorEntry = tkinter.Entry(self.window,  width = 25, textvariable = self.doctor)


self.fNameEntry.grid(pady = 5, column = 3, row = 2)
self.lNameEntry.grid(pady = 5, column = 3, row = 3)
self.addressEntry.grid(pady = 5, column = 3, row = 8)
self.phoneEntry.grid(pady = 5, column = 3, row = 9)
self.emailEntry.grid(pady = 5, column = 3, row = 10)
self.historyEntry.grid(pady = 5, column = 3, row = 12)
self.doctorEntry.grid(pady = 5, column = 3, row = 13)


# Combobox widgets
self.dobBox = tkinter.ttk.Combobox(self.window, values = self.dateList, width = 20)
self.mobBox = tkinter.ttk.Combobox(self.window, values = self.monthList, width = 20)
self.yobBox = tkinter.ttk.Combobox(self.window, values = self.yearList, width = 20)
```

```
self.genderBox = tkinter.ttk.Combobox(self.window, values =
self.genderList, width = 20)

self.bloodGroupBox = tkinter.ttk.Combobox(self.window, values =
self.bloodGroupList, width = 20)


self.dobBox.grid(pady = 5, column = 3, row = 4)

self.mobBox.grid(pady = 5, column = 3, row = 5)

self.yobBox.grid(pady = 5, column = 3, row = 6)

self.genderBox.grid(pady = 5, column = 3, row = 7)

self.bloodGroupBox.grid(pady = 5, column = 3, row = 11)


# Button widgets

tkinter.Button(self.window, width = 20, text = "Update", command =
self.Update).grid(pady = 15, padx = 5, column = 1, row = 14)

tkinter.Button(self.window, width = 20, text = "Reset", command =
self.Reset).grid(pady = 15, padx = 5, column = 2, row = 14)

tkinter.Button(self.window, width = 20, text = "Close", command =
self.window.destroy).grid(pady = 15, padx = 5, column = 3, row = 14)


self.window.mainloop()

def Update(self):

self.database = Database()

self.database.Update(self.fNameEntry.get(), self.lNameEntry.get(),
self.dobBox.get(), self.mobBox.get(), self.yobBox.get(),
```

```
self.genderBox.get(), self.addressEntry.get(), self.phoneEntry.get(),
self.emailEntry.get(), self.bloodGroupBox.get(), self.historyEntry.get(),
self.doctorEntry.get(), self.id)

tkinter.messagebox.showinfo("Updated data", "Successfully updated the
above data in the database")

def Reset(self):

self.fNameEntry.delete(0, tkinter.END)

self.lNameEntry.delete(0, tkinter.END)

self.dobBox.set("")

self.mobBox.set("")

self.yobBox.set("")

self.genderBox.set("")

self.addressEntry.delete(0, tkinter.END)

self.phoneEntry.delete(0, tkinter.END)

self.emailEntry.delete(0, tkinter.END)

self.bloodGroupBox.set("")

self.historyEntry.delete(0, tkinter.END)

self.doctorEntry.delete(0, tkinter.END)


class DatabaseView:

def __init__(self, data):

self.databaseViewWindow = tkinter.Tk()

self.databaseViewWindow.wm_title("Database View")
```

```
# Label widgets
tkinter.Label(self.databaseViewWindow, text = "Database View
Window",  width = 25).grid(pady = 5, column = 1, row = 1)


self.databaseView = tkinter.ttk.Treeview(self.databaseViewWindow)
self.databaseView.grid(pady = 5, column = 1, row = 2)
self.databaseView["show"] = "headings"
self.databaseView["columns"] = ("id", "fName", "lName", "dob",
"mob", "yob", "gender", "address", "phone", "email", "bloodGroup",
"history", "doctor")


# Treeview column headings
self.databaseView.heading("id", text = "ID")
self.databaseView.heading("fName", text = "First Name")
self.databaseView.heading("lName", text = "Last Name")
self.databaseView.heading("dob", text = "D.O.B")
self.databaseView.heading("mob", text = "M.O.B")
self.databaseView.heading("yob", text = "Y.O.B")
self.databaseView.heading("gender", text = "Gender")
self.databaseView.heading("address", text = "Home Address")
self.databaseView.heading("phone", text = "Phone Number")
self.databaseView.heading("email", text = "Email ID")
```

```
self.databaseView.heading("bloodGroup", text = "Blood Group")
self.databaseView.heading("history", text = "History")
self.databaseView.heading("doctor", text = "Doctor")

# Treeview columns
self.databaseView.column("id", width = 40)
self.databaseView.column("fName", width = 100)
self.databaseView.column("lName", width = 100)
self.databaseView.column("dob", width = 60)
self.databaseView.column("mob", width = 60)
self.databaseView.column("yob", width = 60)
self.databaseView.column("gender", width = 60)
self.databaseView.column("address", width = 200)
self.databaseView.column("phone", width = 100)
self.databaseView.column("email", width = 200)
self.databaseView.column("bloodGroup", width = 100)
self.databaseView.column("history", width = 100)
self.databaseView.column("doctor", width = 100)

for record in data:
self.databaseView.insert(", 'end', values=(record))
```

```
self.databaseViewWindow.mainloop()

class SearchDeleteWindow:
def __init__(self, task):
window = tkinter.Tk()
window.wm_title(task + " data")

# Initializing all the variables
self.id = tkinter.StringVar()
self.fName = tkinter.StringVar()
self.lName = tkinter.StringVar()
self.heading = "Please enter Patient ID to " + task

# Labels
tkinter.Label(window, text = self.heading, width = 50).grid(pady = 20,
row = 1)
tkinter.Label(window, text = "Patient ID", width = 10).grid(pady = 5,
row = 2)

# Entry widgets
self.idEntry = tkinter.Entry(window, width = 5, textvariable = self.id)

self.idEntry.grid(pady = 5, row = 3)
```

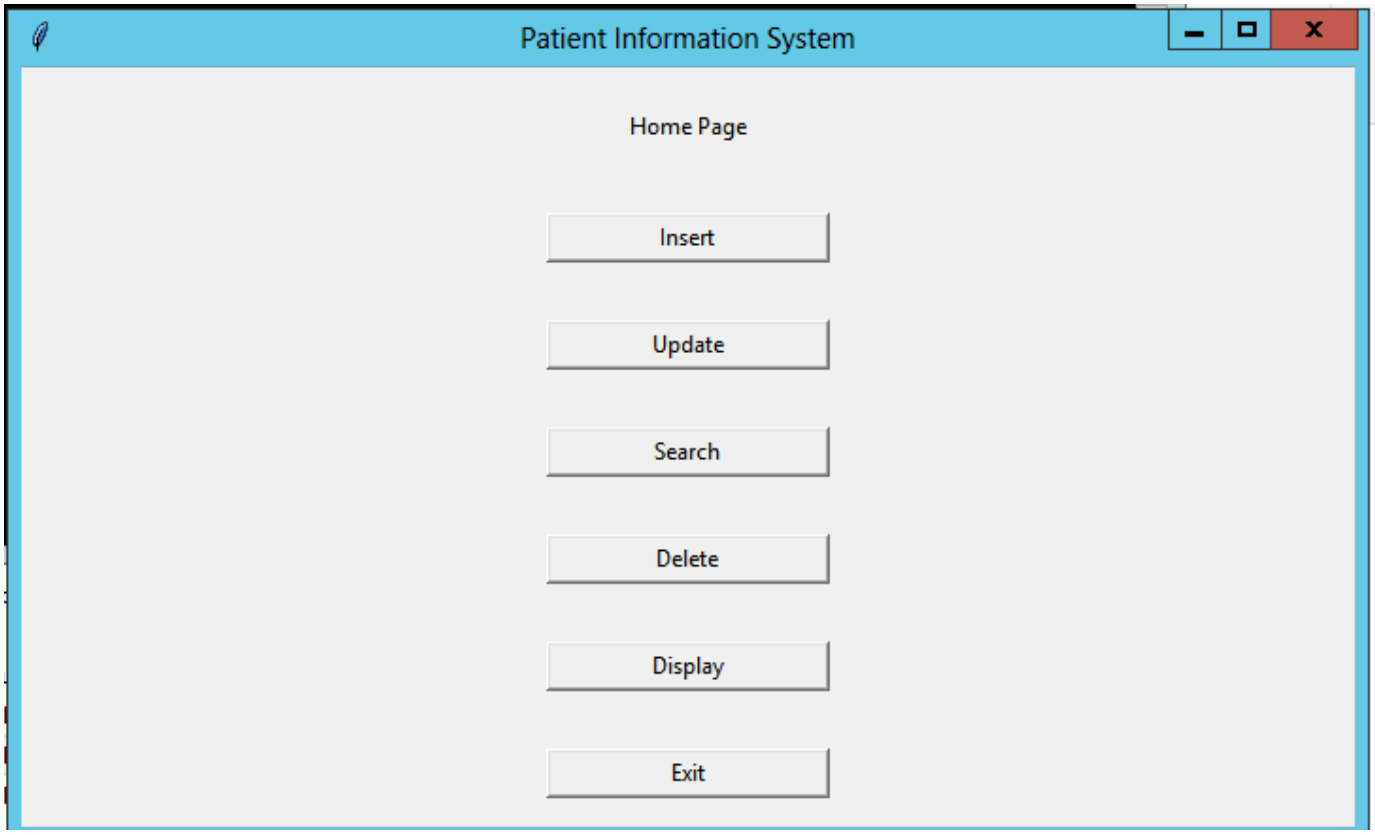
```
# Button widgets
if (task == "Search"):
tkinter.Button(window, width = 20, text = task, command =
self.Search).grid(pady = 15, padx = 5, column = 1, row = 14)
elif (task == "Delete"):
tkinter.Button(window, width = 20, text = task, command =
self.Delete).grid(pady = 15, padx = 5, column = 1, row = 14)
def Search(self):
self.database = Database()
self.data = self.database.Search(self.idEntry.get())
self.databaseView = DatabaseView(self.data)

def Delete(self):
self.database = Database()
self.database.Delete(self.idEntry.get())

class HomePage:
def __init__(self):
self.homePageWindow = tkinter.Tk()
self.homePageWindow.wm_title("Patient Information System")
```

```
tkinter.Label(self.homePageWindow, text = "Home Page",  width =
100).grid(pady = 20, column = 1, row = 1)


tkinter.Button(self.homePageWindow, width = 20, text = "Insert",
command = self.Insert).grid(pady = 15, column = 1, row = 2)
tkinter.Button(self.homePageWindow, width = 20, text = "Update",
command = self.Update).grid(pady = 15, column = 1, row = 3)
tkinter.Button(self.homePageWindow, width = 20, text = "Search",
command = self.Search).grid(pady = 15, column = 1, row = 4)
tkinter.Button(self.homePageWindow, width = 20, text = "Delete",
command = self.Delete).grid(pady = 15, column = 1, row = 5)
tkinter.Button(self.homePageWindow, width = 20, text = "Display",
command = self.Display).grid(pady = 15, column = 1, row = 6)
tkinter.Button(self.homePageWindow, width = 20, text = "Exit",
command = self.homePageWindow.destroy).grid(pady = 15, column = 1,
row = 7)


self.homePageWindow.mainloop()


def Insert(self):
self.insertWindow = InsertWindow()


def Update(self):
self.updateIDWindow = tkinter.Tk()
```

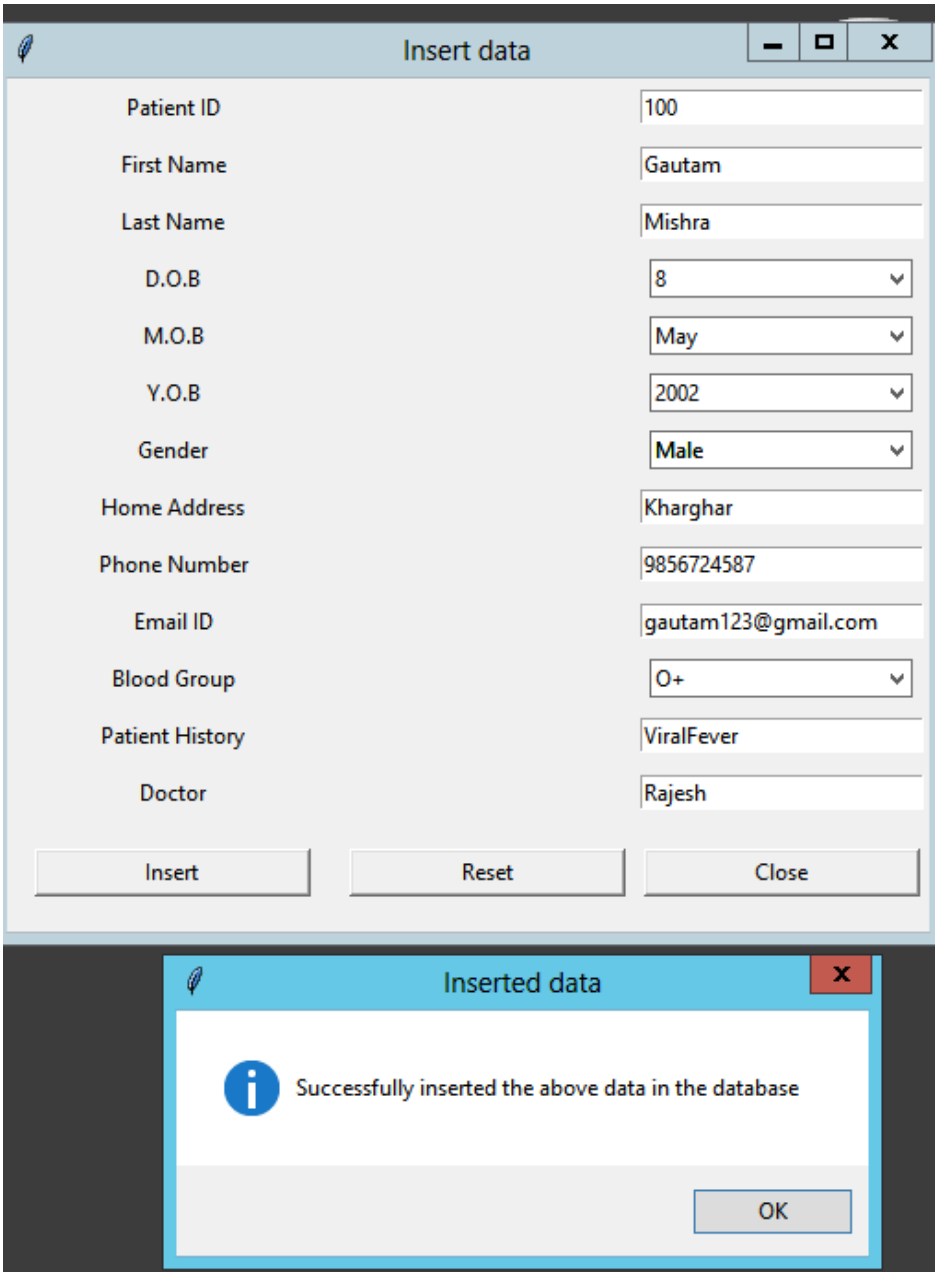
```
self.updateIDWindow.wm_title("Update data")

# Initializing all the variables
self.id = tkinter.StringVar()

# Label
tkinter.Label(self.updateIDWindow, text = "Enter the ID to update",
width = 50).grid(pady = 20, row = 1)

# Entry widgets
self.idEntry = tkinter.Entry(self.updateIDWindow, width = 5,
textvariable = self.id)

self.idEntry.grid(pady = 10, row = 2)

# Button widgets
tkinter.Button(self.updateIDWindow, width = 20, text = "Update",
command = self.updateID).grid(pady = 10, row = 3)

self.updateIDWindow.mainloop()

def updateID(self):
self.updateWindow = UpdateWindow(self.idEntry.get())
```

```
self.updateIDWindow.destroy()

def Search(self):
self.searchWindow = SearchDeleteWindow("Search")

def Delete(self):
self.deleteWindow = SearchDeleteWindow("Delete")

def Display(self):
self.database = Database()
self.data = self.database.Display()
self.displayWindow = DatabaseView(self.data)

homePage = HomePage()
```

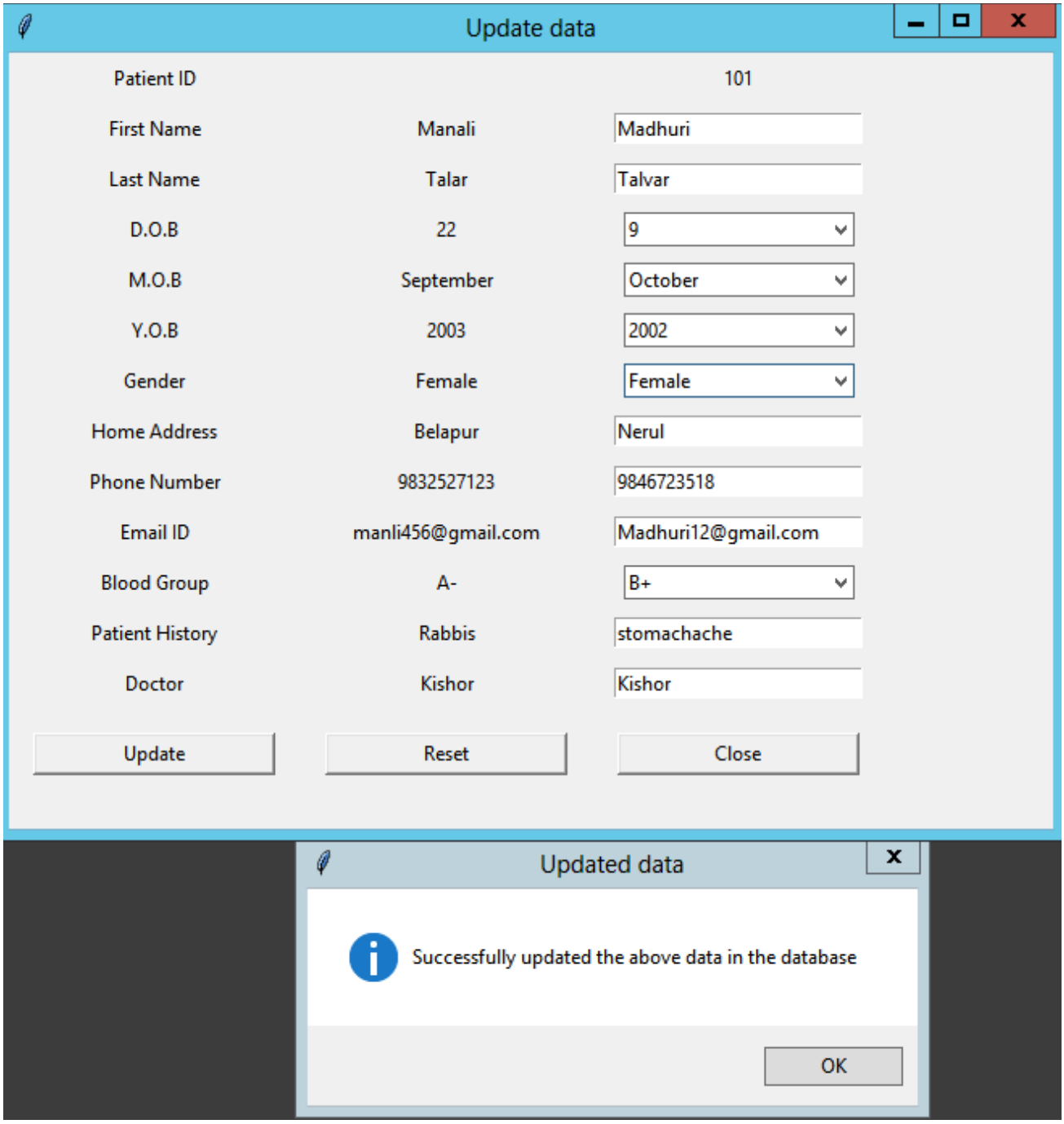
output for the home page



Output for the inserting info

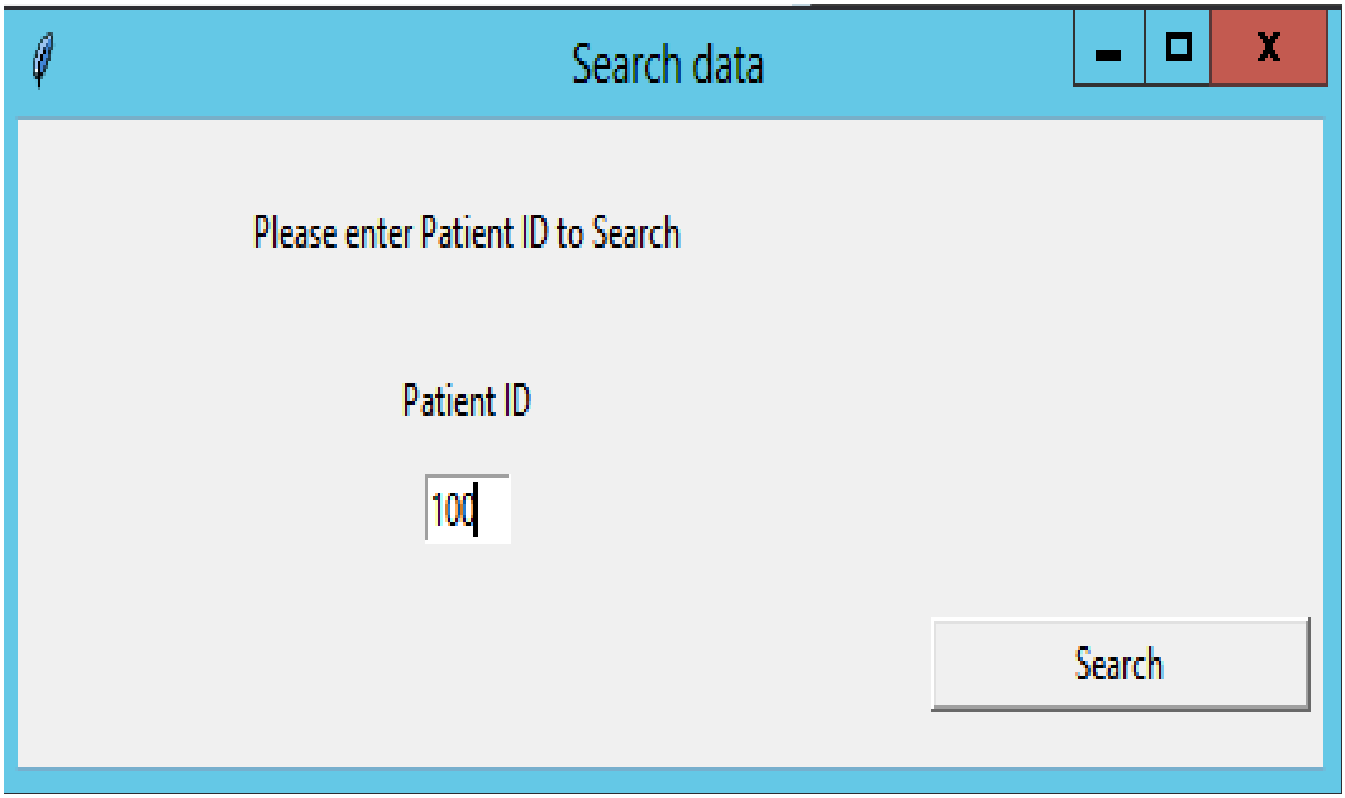


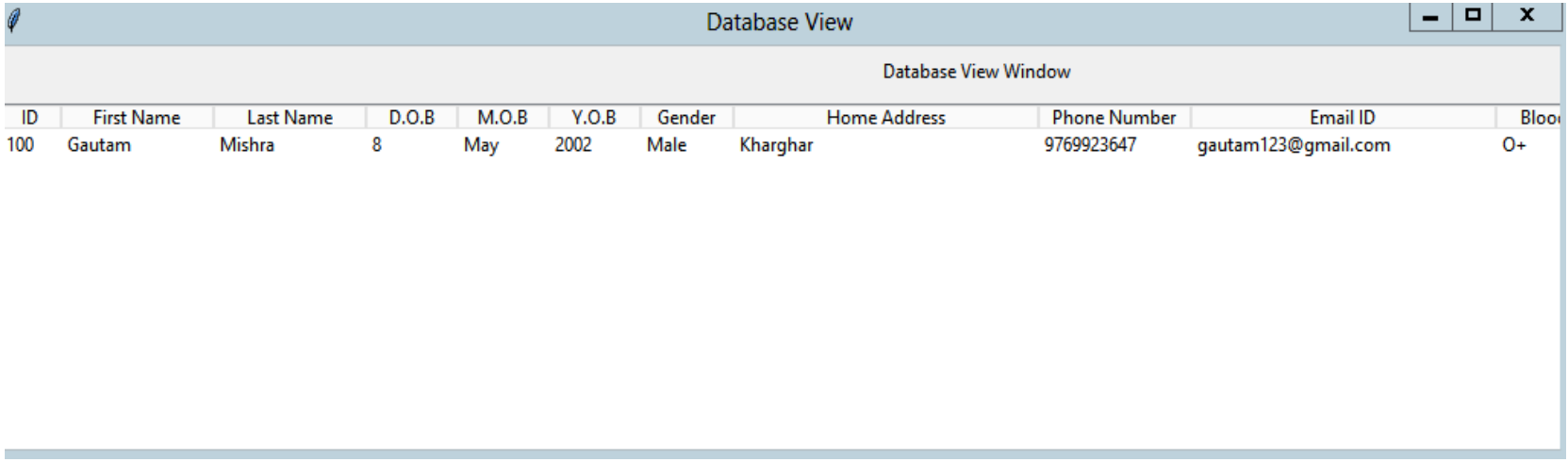
Output for updating database


Update data
Patient ID
101
First Name
Manali
Madhuri
Last Name
Talar
Talvar
D.O.B
22
9
M.O.B
September
October
Y.O.B
2003
2002
Gender
Female
Female
Home Address
Belapur
Nerul
Phone Number
9832527123
9846723518
Email ID
manli456@gmail.com
Madhuri12@gmail.com
Blood Group
A-
B+
Patient History
Rabbis
stomachache
Doctor
Kishor
Kishor
Update
Reset
Close
Updated data
Successfully updated the above data in the database
OK

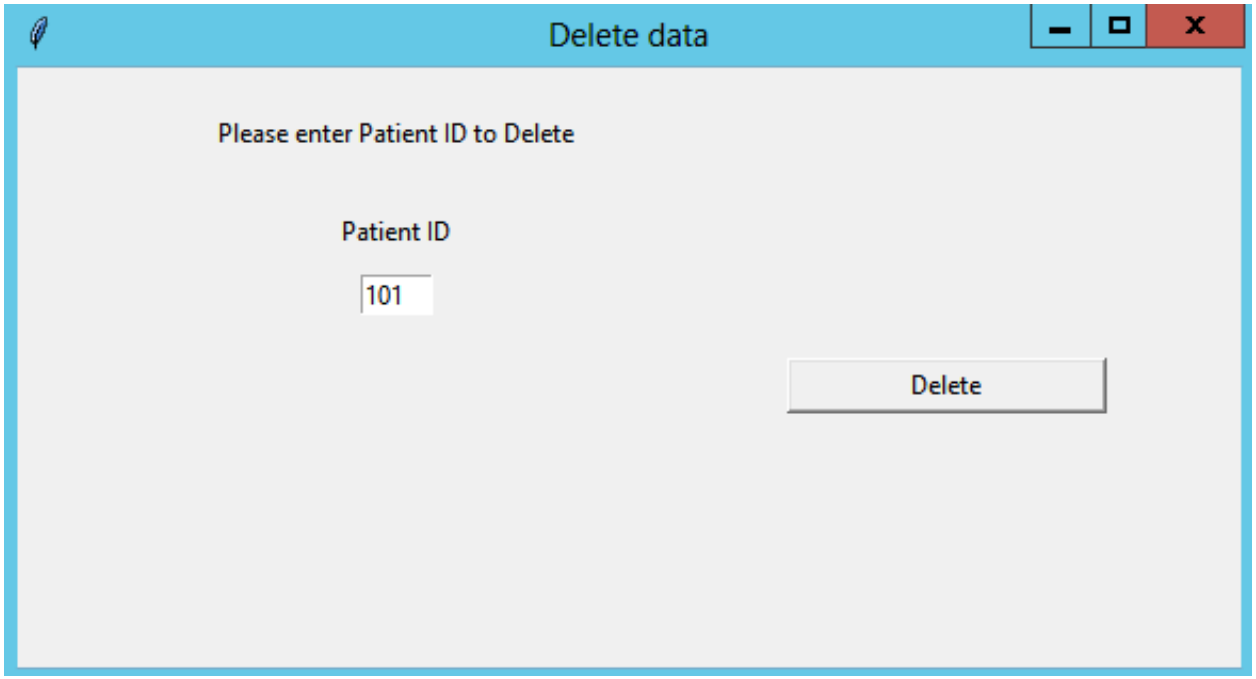
Output for search in database

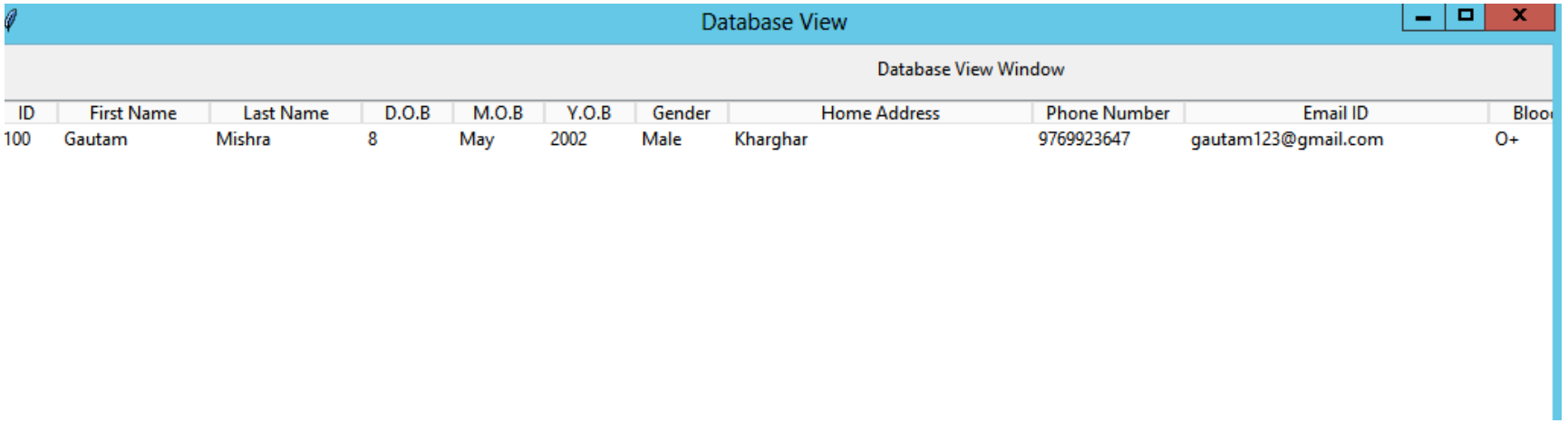
2



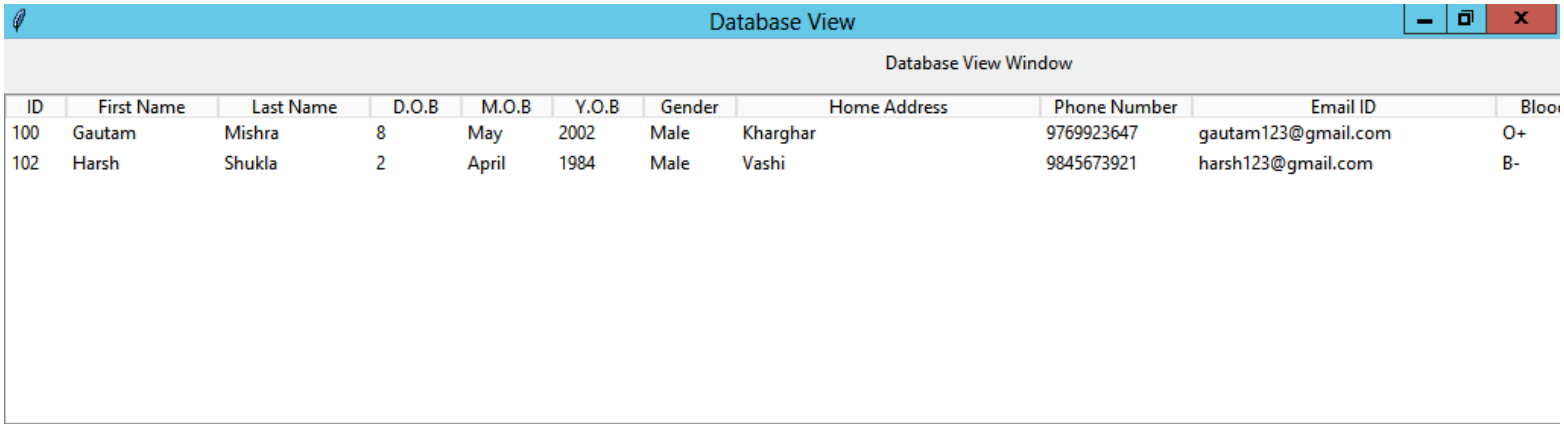

Database View
Database View Window
ID
First Name
Last Name
D.O.B
M.O.B
Y.O.B
Gender
Home Address
Phone Number
Email ID
Bloo
100
Gautam
Mishra
8
May
2002
Male
Kharghar
9769923647
gautam123@gmail.com
O+

Output for delete data





Output for display data



SYSTEM REQUIREMENT

Hardware:

CPU: Intel Pentium P6200

Hard Disk: 500GB

RAM: 4GB

VDU: Video Display Adaptar

Software:

Platform: Windows 7 and above

Programming Languages: Python, Mysql

BIBLIOGRAPHY

1.TEXTBOOK

2. REFERENCE BOOK – Think Python

The complete reference - MySQL

3. ONLINE TUTORIALS- www.w3schools.com

www.tutorialspoints.com

www.python.org

4.ONLINE VISUAL REFERNCES- www.edureka.com

www.Udemy.com