**System Implementations**

**Recommended System Requirements**

Processors: Intel® Core™ i3 processor 4300M at 2.60 GHz.

Disk space: 4 to 8 GB.

Operating systems: Windows® 10, MACOS, and UBUNTU.

Python Versions: 3.X.X or Higher.

**Minimum System Requirements**

Processors: Intel Atom® processor or Intel® Core™ i3 processor.

Disk space: 1 GB.

Operating systems: Windows 7 or later, MACOS, and UBUNTU.

Python Versions: 2.7.X, 3.9.X.

**ACKNOWLEDGEMENT**TTT

First and foremost, praises and thanks to the God, the Almighty, for His showers of blessings throughout my research work to complete the research successfully.

We would like to express my deep and sincere gratitude to my subject teacher, Mr. Amit Udiwal, for giving me the opportunity to do research and providing invaluable guidance throughout this research. His dynamism, vision, sincerity and motivation have deeply inspired me. He has taught me the methodology to carry out the research and to present the research works as clearly as and honour to work and study under his guidance. We are very much thankful to our Sr. Jasmin for giving valuable time and moral support to develop this software. We would like to take opportunity to extend my sincere thanks and gratitude to our parents for being a source of inspiration and providing time and freedom to develop this software project. We also feel indebted to my friends for the valuable suggestions during the project work.

Ishika Bairagi

[Roll No.

Class XII

**CERTIFICATE**

This is to certify that the project on ‘Patient Information System’ is a work done by Gayatri Yadav fulfilment of CBSE’S AISSCE EXAMINATION 2022-23 and has been carried out under my direct supervision and guidance. This report or a similar report on the topic has not been submitted for any other examination and does not form any other examination and does not form any other course undergone by the candidate.

Name: Ishika Bairagi [Roll No.

………………….

Signature of Teacher / Guide

Name: Mr. Amit Udiwal

Designation:

……………….

….………………

**REFERENCE**

The order to work on this project on ‘Sales And Inventory Management System’ the following books & literature are referred by me during the various phrases of department of the project.

• http://www.python.org/.

• http://www.itsourcecode.org/.

• http://www.wikipedia.org/.

• Informatics Practices for Class XII

- By Sumita Arora

• Together with informatics practices.

Other than the above mentioned books, the suggestions and supervision of my teacher and my class experience also helped me to develop this software project.

**Introduction**

The Patient Information System project is written in Python. The project file contains python scripts and a database file. The system is GUI based with the simple controls for the users. The system contains only admin side. The admin user can perform various tasks such as add a new record, update a record, search for a record, delete a record and also display all the records in the database.

This Patient Information System The project has a graphical user interface provided by the Python programming language. It provides a GUI where the user can enter the details of a patient which will act as a record in the database. The user can also perform various operations on the records. He/she can add a new record, update, search for a records. Moreover, the admin can delete the existing record and also can display all the records in the database. The system design is simple that the user won’t find it difficult to use and navigate.

**Objective and**

**Scope of The Project**

The main objective of the Python Project on Patient Information System is to manage the details of Doctors, Medicines, Patient, Tests, Reports. It manages all the information about Doctors, Treatments, Reports, Doctors. The project is totally built at administrative end and thus only the administrator is guaranteed the access. The purpose of the project is to build an application program to reduce the manual work for managing the Doctors, Medicines, Treatments, Patient. It tracks all the details about the Patient, Tests, Reports.

***Functions:***

* Insert Patients Information
* Update Patients Information
* Delete patients information
* Display Patients Information
* Search Patients Information

**Patients Information System**

import tkinter

import tkinter.ttk

import tkinter.messagebox

import sqlite3

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()