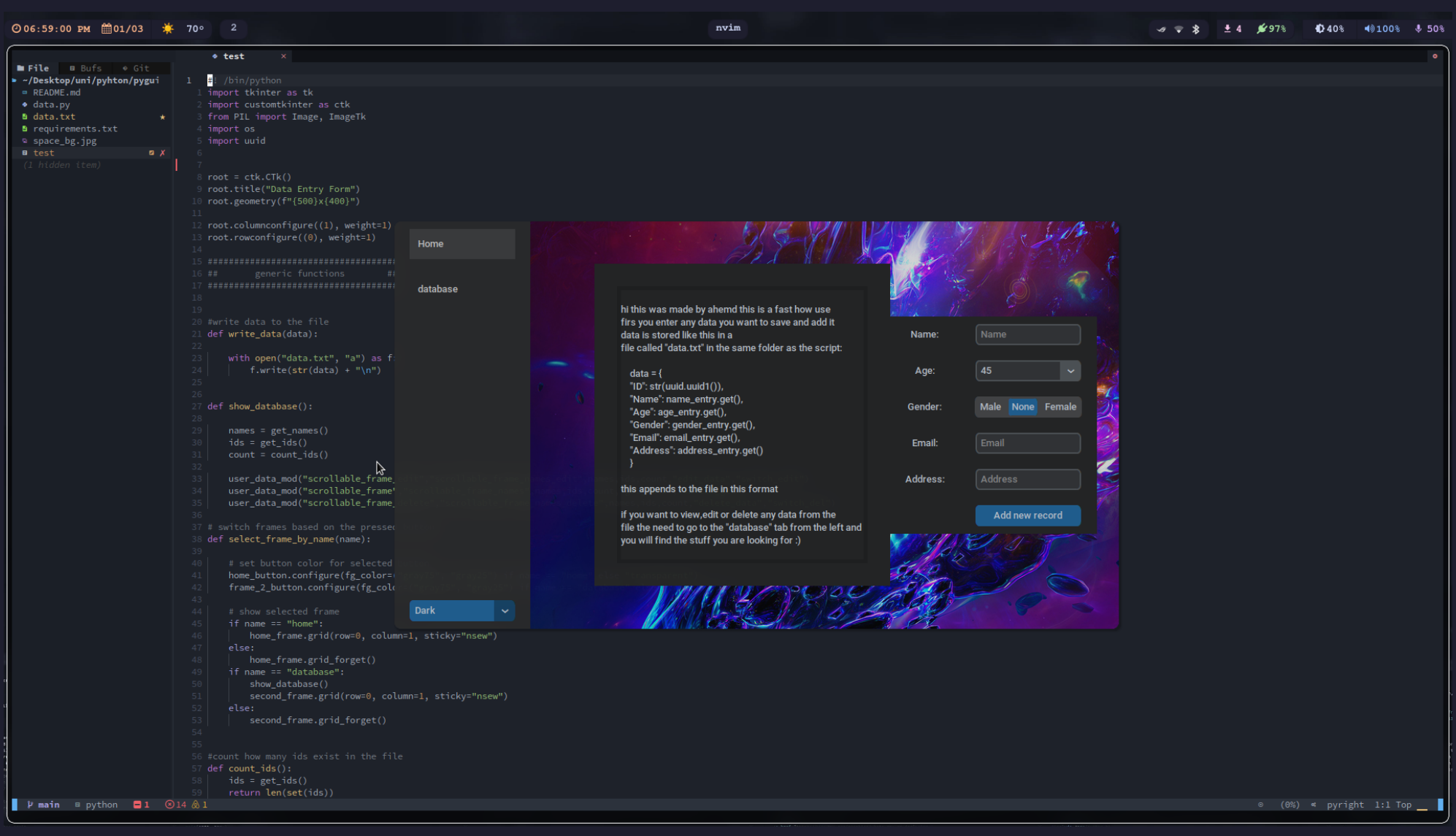
save



this is the function linked to the button and is the one that formats the data and pass it to the function that writes it to the file

```
def save_data():

        # format the data and append it to the file
        data = {
        "ID": str(uuid.uuid1()),
        "Name": name_entry.get(),
        "Age": age_entry.get(),
        "Gender": gender_entry.get(),
        "Email": email_entry.get(),
        "Address": address_entry.get()
        }

        write_data(data)
        # deletes the data from the entries
        name_entry.delete(0, tk.END)
        age_entry.delete(0, tk.END)
        gender_entry.delete(0, tk.END)
        email_entry.delete(0, tk.END)
        address_entry.delete(0, tk.END)
```

this function opens the file in append mode so it doesnt delete the other records

```
def write_data(data):
        with open("data.txt", "a") as f:
                f.write(str(data) + "\n")
```

database it self

this is the function used to generate the buttons on all the database tab

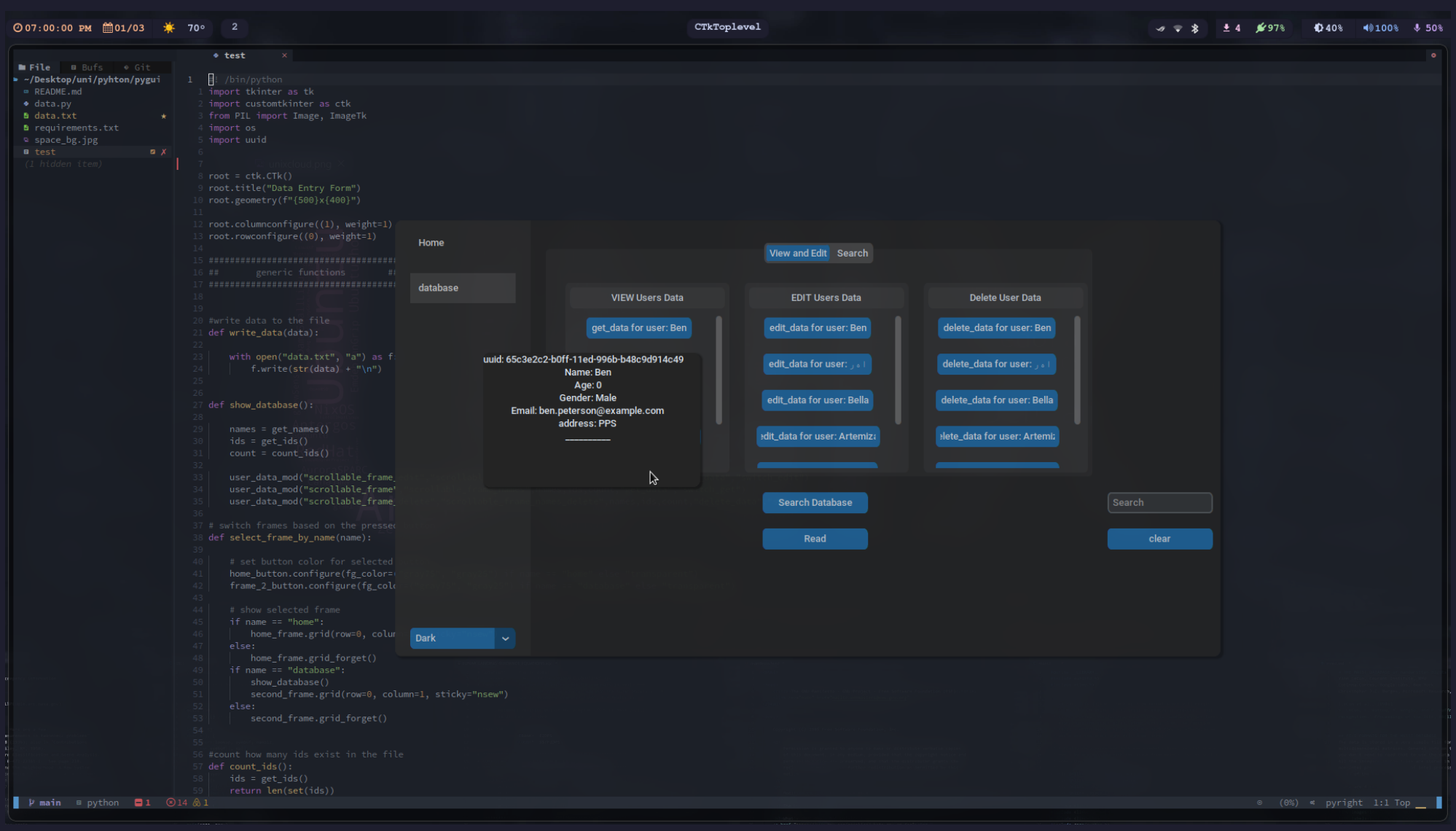
```
def user_data_mod(master,list, names,ids,count, call_type, switch):

        for i in range(count):
                name = names[i]
                id = ids[i]

                switch = ctk.CTkButton(master=eval(master), text=f"{call_type} for user: {name}")
                switch.grid(row=i, column=0, padx=10, pady=(0, 20))
                switch.configure(command=lambda id=id ,call_type=call_type : button_make(id,call_type))
                print(f"dynamic {id}")
                eval(list).append(switch)

        return list
```

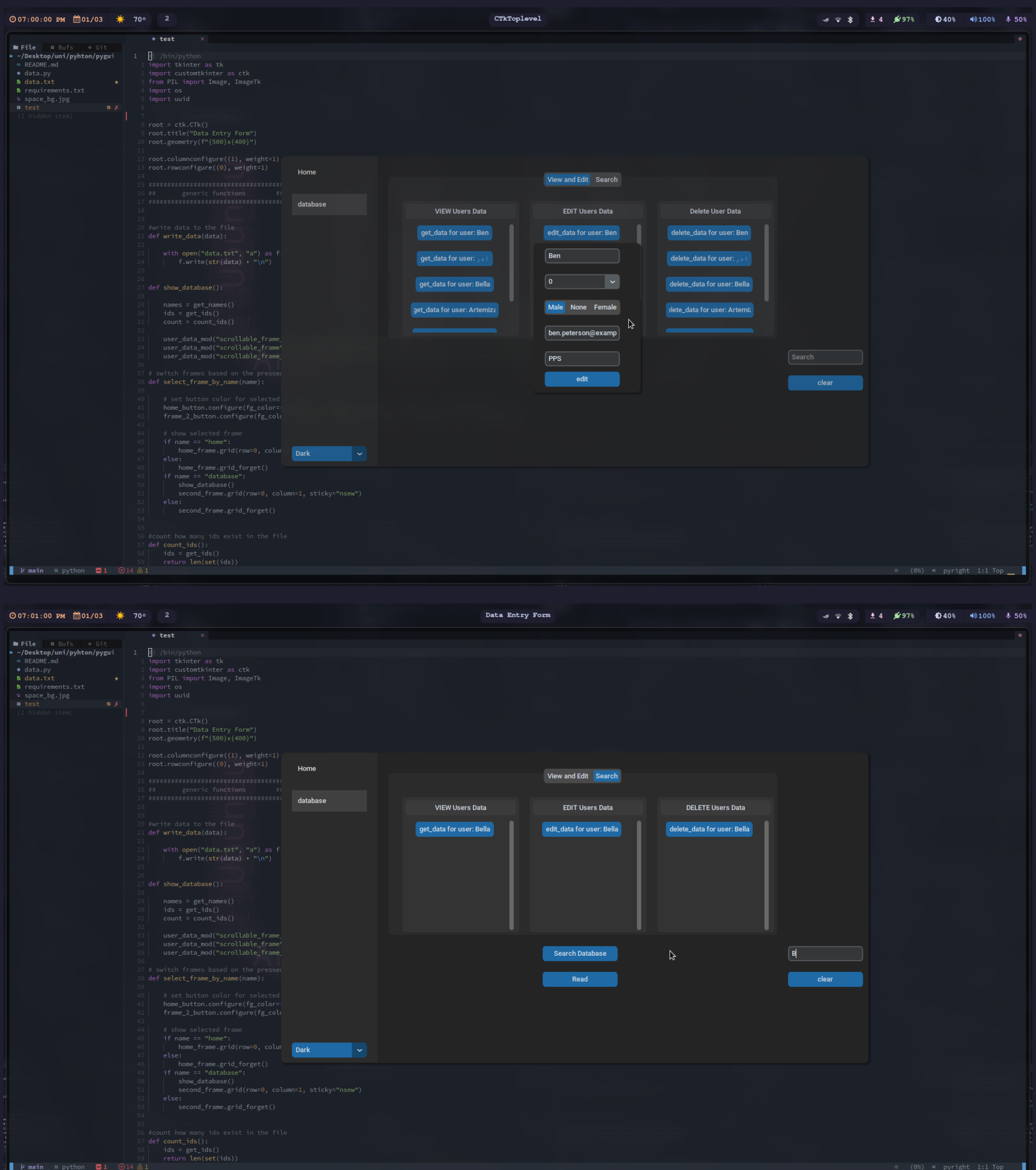
view data



this function reads the file and finds the button clicked and shows the data for the user

```
def read_user_data(user_id):
                with open("data.txt", "r") as file:
                        for line in file:
                                user = eval(line)
                                if user["ID"] == user_id:
                                        return line
        return None
```

edit/delete



this functions is used to edit and delete any records

```
def edit_data(user_id, mode,data=[]):
    with open("data.txt", "r") as file:
        lines = file.readlines()

    with open("data.txt", "w") as file:
        for line in lines:
            line = eval(line)
            if line['ID'] != user_id:
                file.write(str(line) + "\n")
                print(line)
            else:

                if (mode == "edit"):
                    data = {
                        "ID": data['ID'],
                        "Name": data["Name"].get(),
                        "Age": data["Age"].get(),
                        "Gender": data["Gender"].get(),
                        "Email": data["Email"].get(),
                        "Address": data["Address"].get()
                    }
                    print(data)
                    file.write(str(data) + "\n")
                    print(f"got edited: {line}")
```

```
            elif (mode == "elete"):
                print(f"got deleted: {line}")
```

this is all the code with the gui elements but it wont work since you dont have the background image on the same folder as the script you can name any image `space_bg.jpg` and put it in the same folder as the script to get it to work
if you dont want to you can clone my github repo and it should work fine

https://github.com/HqNw/pygui

```
#! /bin/python
import tkinter as tk
import customtkinter as ctk
from PIL import Image, ImageTk
import os
import uuid


root = ctk.CTk()
root.title("Data Entry Form")
root.geometry(f"{500}x{400}")

root.columnconfigure((1), weight=1)
root.rowconfigure((0), weight=1)

####################################
##       generic functions        ##
####################################


#write data to the file
def write_data(data):

    with open("data.txt", "a") as f:
        f.write(str(data) + "\n")


def show_database():

    names = get_names()
    ids = get_ids()
    count = count_ids()

    user_data_mod("scrollable_frame_edit","scrollable_frame_names_edit",names,ids,count,"edit_data","switch_edit")
    user_data_mod("scrollable_frame","scrollable_frame_names",names,ids,count,"get_data","switch_get")
    user_data_mod("scrollable_frame_delete","scrollable_frame_names_delete",names,ids,count,"delete_data","switch_del")

# switch frames based on the pressed button
def select_frame_by_name(name):

    # set button color for selected button
    home_button.configure(fg_color=("gray75", "gray25") if name == "home" else "transparent")
    frame_2_button.configure(fg_color=("gray75", "gray25") if name == "database" else "transparent")

    # show selected frame
    if name == "home":
        home_frame.grid(row=0, column=1, sticky="nsew")
    else:
        home_frame.grid_forget()
    if name == "database":
        show_database()
        second_frame.grid(row=0, column=1, sticky="nsew")
    else:
        second_frame.grid_forget()


#count how many ids exist in the file
def count_ids():
    ids = get_ids()
    return len(set(ids))

#put every name in the file in a list
def get_names():
    with open("data.txt","r") as f:
        lines = f.readlines()
        names = [eval(line)["Name"] for line in lines]
        return names

#put every id in the file in a list
def get_ids():
    with open("data.txt", "r") as f:
        lines = f.readlines()
        ids = [eval(line)["ID"] for line in lines]
        return ids


def button_make(id,call_type):
    if call_type == "get_data":
        get_data(id)
    elif call_type == "edit_data":
        edit_window(id)
    elif call_type == "delete_data":
        delete_data(id)


def search_data():
    input = search_entry.get()
    print("looking for username: ", input)
    search_result(input)
```

```python
def search_result(username):
    i=0
    with open("data.txt","r") as f:
        lines = f.readlines()
        for line in lines:
            line = eval(line)
            name = line["Name"]
            id = line["ID"]
            if (name.find(username) != -1 ):
                names_list=[]
                names_list.append(name)

                ids_list =[]
                ids_list.append(id)

                print(f"found user in search: {name}  uuid: {id}")
    result_window(names_list,ids_list)


def result_window(name_list,id_list):
    count = len(id_list)
    user_data_mod("search_frame_edit","search_frame_edit_list",name_list,id_list,count,"edit_data","switch_edit")
    user_data_mod("search_frame_names","search_frame_names_list",name_list,id_list,count,"get_data","switch_get")
    user_data_mod("search_frame_delete","search_frame_delete_list",name_list,id_list,count,"delete_data","switch_del")


def user_data_mod(master,list, names,ids,count, call_type, switch):
    for i in range(count):
        name = names[i]
        id = ids[i]

        switch = ctk.CTkButton(master=eval(master), text=f"{call_type} for user: {name}")
        switch.grid(row=i, column=0, padx=10, pady=(0, 20))
        switch.configure(command=lambda id=id ,call_type=call_type : button_make(id,call_type))
        print(f"dynamic {id}")
        eval(list).append(switch)
    return list


def get_data(id):
    user_data = read_user_data(id)
    user_data = eval(user_data)

    window = ctk.CTkToplevel(second_frame)
    window_label = ctk.CTkLabel(window)
    window_label.grid(row=0 , column=0)

    window_label.configure(text =f"""uuid: {user_data["ID"]}
    Name: {user_data["Name"]}
    Age: {user_data["Age"]}
    Gender: {user_data["Gender"]}
    Email: {user_data["Email"]}
    address: {user_data["Address"]}
    ----------
    """)

    print(user_data)


def read_user_data(user_id):
    with open("data.txt", "r") as file:
        for line in file:
            user = eval(line)
            if user["ID"] == user_id:
                return line
    return None

def delete_user(user_id):
    with open("data.txt", "r") as file:
        lines = file.readlines()

    with open("data.txt", "w") as file:
        for line in lines:
            line = eval(line)
            if line['ID'] != user_id:
                file.write(str(line) + "\n")
                print(line)
            else:
                print(f"got deleted: {line}")


def edit_window(data):
    search_result_window = None

    if search_result_window is None or not search_result_window.winfo_exists():
        search_result_window = ctk.CTkToplevel(second_frame)

    else:
        search_result_window.focus()

    # Input fields
    data = eval(read_user_data(data))

    name_edit_entry_var = ctk.StringVar(value=data["Name"])
    name_edit_entry = ctk.CTkEntry(search_result_window ,textvariable=name_edit_entry_var)
    name_edit_entry.grid(row=0, column=1,padx=20, pady=10)

    combobox_var = ctk.StringVar(value=data["Age"])
    age_edit_entry = ctk.CTkComboBox(search_result_window, values=age_to_100, command=combobox_callback, variable=combobox_var)
    age_edit_entry.grid(row=1, column=1,padx=20, pady=10)

    segemented_button_var = ctk.StringVar(value=data["Gender"].capitalize())
    gender_edit_entry = ctk.CTkSegmentedButton(search_result_window, values=["Male", "None", "Female"], variable=segemented_button_var)
```

```
    gender_edit_entry.grid(row=2, column=1,padx=20, pady=10)

    email_edit_entry_var = ctk.StringVar(value=data["Email"])
    email_edit_entry = ctk.CTkEntry(search_result_window ,textvariable=email_edit_entry_var)
    email_edit_entry.grid(row=3, column=1,padx=20, pady=10)

    address_edit_entry_var = ctk.StringVar(value=data["Address"])
    address_edit_entry = ctk.CTkEntry(search_result_window ,textvariable=address_edit_entry_var)
    address_edit_entry.grid(row=4, column=1,padx=20, pady=10)

    line = {
        "ID": data['ID'],
        "Name": name_edit_entry,
        "Age": age_edit_entry,
        "Gender": gender_edit_entry,
        "Email": email_edit_entry,
        "Address": address_edit_entry
        }
    print(line)
    edit_window_button = ctk.CTkButton(search_result_window, text="edit", command=lambda id=data["ID"],line= line:edit_data(id,"edit",line)  )
    edit_window_button.grid(row=5, column=1, padx=10, pady=(0, 20))

    return line


def edit_data(user_id, mode,data=[]):
    with open("data.txt", "r") as file:
        lines = file.readlines()

    with open("data.txt", "w") as file:
        for line in lines:
            line = eval(line)
            if line['ID'] != user_id:
                file.write(str(line) + "\n")
                print(line)
            else:

                if (mode == "edit"):
                    data = {
                        "ID": data['ID'],
                        "Name": data["Name"].get(),
                        "Age": data["Age"].get(),
                        "Gender": data["Gender"].get(),
                        "Email": data["Email"].get(),
                        "Address": data["Address"].get()
                    }
                    print(data)
                    file.write(str(data) + "\n")
                    print(f"got edited: {line}")

                elif (mode == "elete"):
                    print(f"got deleted: {line}")


###################################
##          main window          ##
###################################

########################
#       frames         #
########################

def home_button_event():
    select_frame_by_name("home")

def frame_2_button_event():
    select_frame_by_name("database")

def change_appearance_mode_event( new_appearance_mode: str):
    ctk.set_appearance_mode(new_appearance_mode)


# create navigation frame
navigation_frame = ctk.CTkFrame(root,width=140, corner_radius=0)
navigation_frame.grid(row=0, column=0,rowspan=3, sticky="nsew")
navigation_frame.grid_rowconfigure(3, weight=1)

home_button = ctk.CTkButton(navigation_frame, corner_radius=0, height=40, border_spacing=10, text="Home", fg_color="transparent", text_color=("gray10",
"gray90"), hover_color=("gray70", "gray30"), anchor="w", command=home_button_event)
home_button.grid(row=0, column=0, padx=20, pady=10, sticky="ew")

frame_2_button = ctk.CTkButton(navigation_frame, corner_radius=0, height=40, border_spacing=10, text="database", fg_color="transparent", text_color=
("gray10", "gray90"), hover_color=("gray70", "gray30"), anchor="w", command=frame_2_button_event)
frame_2_button.grid(row=1, column=0, padx=20, pady=10, sticky="ew")

appearance_mode_optionemenu = ctk.CTkOptionMenu(navigation_frame, values=["Dark", "Light", "System"],command=change_appearance_mode_event)
appearance_mode_optionemenu.grid(row=6, column=0, padx=20, pady=(10, 10))

# create home frame
home_frame = ctk.CTkFrame(root, corner_radius=0, fg_color="transparent")
home_frame.grid_columnconfigure(0, weight=1)

# create second frame
second_frame = ctk.CTkFrame(root, corner_radius=0, fg_color="transparent")
second_frame.grid_columnconfigure(0, weight=1)



##########################################
#       data input fields and buttons       #
##########################################
```

```
IMAGE_PATH = 'space_bg.jpg'
IMAGE_PATH = os.path.abspath(IMAGE_PATH)


img = ImageTk.PhotoImage(Image.open(IMAGE_PATH))
lbl = tk.Label(home_frame, image=img)
lbl.img = img  # Keep a reference in case this code put is in a function.
lbl.place(relx=0.5, rely=0.5, anchor='center')


def combobox_callback(choice):
    print("combobox dropdown clicked:", choice)



home_frame.columnconfigure((0,5), weight=1)
home_frame.columnconfigure(0,weight=3)
home_frame.rowconfigure((1), weight=1)

# login frame
login_frame = ctk.CTkFrame(home_frame, corner_radius=10,height=300, fg_color="transparent" )
login_frame.grid_columnconfigure((0,1,2), weight=1)
login_frame.grid(row=1,column=2)

data_frame = ctk.CTkFrame(home_frame, corner_radius=10,height = login_frame.cget('height'), fg_color="transparent" )
data_frame.grid_columnconfigure((0,1,2), weight=1)
data_frame.grid(row=1,column=1)

data_frame_2 = ctk.CTkFrame(data_frame, corner_radius=10,bg_color="#1E1E1E", fg_color="transparent" )
data_frame_2.grid_columnconfigure(0, weight=1)
data_frame_2.grid(row=0,column=0,padx=30, pady=30)

data_label = ctk.CTkLabel(data_frame_2,justify="left", text="""
hi this was made by ahemd this is a fast how use
firs you enter any data you want to save and add it
data is stored like this in a
file called "data.txt" in the same folder as the script:

    data = {
    "ID": str(uuid.uuid1()),
    "Name": name_entry.get(),
    "Age": age_entry.get(),
    "Gender": gender_entry.get(),
    "Email": email_entry.get(),
    "Address": address_entry.get()
    }

this appends to the file in this format

if you want to view,edit or delete any data from the
file the need to go to the "database" tab from the left and
you will find the stuff you are looking for :)
""")
data_label.grid(row=0 ,column=0,padx=5, pady=5)

############
name_label = ctk.CTkLabel(login_frame, text="Name:")
name_label.grid(row=0 ,column=0,padx=20, pady=10)

age_label = ctk.CTkLabel(login_frame, text="Age:")
age_label.grid(row=1 ,column=0,padx=20, pady=10)

gender_label = ctk.CTkLabel(login_frame, text="Gender:")
gender_label.grid(row=2 ,column=0,padx=20, pady=10)

email_label = ctk.CTkLabel(login_frame, text="Email:")
email_label.grid(row=3 ,column=0,padx=20, pady=10)

address_label = ctk.CTkLabel(login_frame, text="Address:")
address_label.grid(row=4 ,column=0,padx=20, pady=10)
############


# Input fields
name_entry = ctk.CTkEntry(login_frame ,placeholder_text="Name")
name_entry.grid(row=0, column=1,padx=20, pady=10)


age_to_100=[]
for i in range(100):
    age_to_100 = tuple(list(age_to_100)+[str(i)])

combobox_var = ctk.StringVar(value="0")  # set initial value
age_entry = ctk.CTkComboBox(login_frame, values=age_to_100, command=combobox_callback, variable=combobox_var)
age_entry.grid(row=1, column=1,padx=20, pady=10)



segemented_button_var = ctk.StringVar(value="None")
gender_entry = ctk.CTkSegmentedButton(login_frame, values=["Male", "None", "Female"], variable=segemented_button_var)
gender_entry.grid(row=2, column=1,padx=20, pady=10)


email_entry = ctk.CTkEntry(login_frame ,placeholder_text="Email")
email_entry.grid(row=3, column=1,padx=20, pady=10)

address_entry = ctk.CTkEntry(login_frame ,placeholder_text="Address")
address_entry.grid(row=4, column=1,padx=20, pady=10)


def add_tab():
```

```
    tabview = ctk.CTkTabview(second_frame, width=250)
    tabview.grid(row=0, column=0,padx=(20,10), pady=(20, 20), sticky="nsew")
    tabview.add("View and Edit")
    tabview.add("Search")
    tabview.tab("View and Edit").grid_columnconfigure((0,1,2), weight=1)
    tabview.tab("Search").grid_columnconfigure((0,1,2), weight=1)
    return tabview

tabview = add_tab()


def clear():
    list = scrollable_frame.grid_slaves()
    print(list)
    for l in list:
        print(l)
        l.destroy()
    add_tab()


clear_button = ctk.CTkButton(second_frame, text="clear", command=clear)
clear_button.grid(row=5, column=1, padx=10, pady=(0, 20))



def save_data():
    # format the data and append it to the file
    data = {
    "ID": str(uuid.uuid1()),
    "Name": name_entry.get(),
    "Age": age_entry.get(),
    "Gender": gender_entry.get(),
    "Email": email_entry.get(),
    "Address": address_entry.get()
    }
    write_data(data)
    # deletes the data from the entries
    name_entry.delete(0, tk.END)
    age_entry.delete(0, tk.END)
    gender_entry.delete(0, tk.END)
    email_entry.delete(0, tk.END)
    address_entry.delete(0, tk.END)

save_button = ctk.CTkButton(login_frame, text="Add new record", command=save_data)




scrollable_frame = ctk.CTkScrollableFrame(tabview.tab("View and Edit"), label_text="VIEW Users Data")
scrollable_frame.grid(row=0, column=0, padx=(20, 0), pady=(20, 0), sticky="nsew")
scrollable_frame.grid_columnconfigure(0, weight=1)
scrollable_frame_names = []

scrollable_frame_edit = ctk.CTkScrollableFrame(tabview.tab("View and Edit"), label_text="EDIT Users Data")
scrollable_frame_edit.grid(row=0, column=1, padx=(20, 0), pady=(20, 0), sticky="nsew")
scrollable_frame_edit.grid_columnconfigure(0, weight=1)
scrollable_frame_names_edit = []

scrollable_frame_delete = ctk.CTkScrollableFrame(tabview.tab("View and Edit"), label_text="Delete User Data")
scrollable_frame_delete.grid(row=0, column=2, padx=(20, 0), pady=(20, 0), sticky="nsew")
scrollable_frame_delete.grid_columnconfigure(0, weight=1)
scrollable_frame_names_delete = []


search_frame_names = ctk.CTkScrollableFrame(tabview.tab("Search"), label_text="VIEW Users Data")
search_frame_names.grid(row=0, column=0, padx=(20, 0), pady=(20, 0), sticky="nsew")
search_frame_names.grid_columnconfigure(0, weight=1)
search_frame_names_list = []

search_frame_edit = ctk.CTkScrollableFrame(tabview.tab("Search"), label_text="EDIT Users Data")
search_frame_edit.grid(row=0, column=1, padx=(20, 0), pady=(20, 0), sticky="nsew")
search_frame_edit.grid_columnconfigure(0, weight=1)
search_frame_edit_list = []

search_frame_delete = ctk.CTkScrollableFrame(tabview.tab("Search"), label_text="DELETE Users Data")
search_frame_delete.grid(row=0, column=2, padx=(20, 0), pady=(20, 0), sticky="nsew")
search_frame_delete.grid_columnconfigure(0, weight=1)
search_frame_delete_list = []


def delete_data (user_id):
    delete_user(user_id)


search = ctk.CTkButton(second_frame, text="Search Database", command=search_data)
search.grid(row=2, column=0, padx=10, pady=(0, 20))

search_entry = ctk.CTkEntry(second_frame ,placeholder_text="Search")
search_entry.grid(row=2, column=1, padx=10, pady=(0, 20))

######################
#       debug        #
######################

def read_data():
    with open("data.txt", "r") as f:
        lines = f.readlines()
        for line in lines:
            data = eval(line)
            print("ID:", data["ID"])
            print("Name:", data["Name"])
            print("Age:", data["Age"])
```

```
            print("Gender:", data["Gender"])
            print("Email:", data["Email"])
            print("Address:", data["Address"])
            print("\n")
        print("__________")
    show_database()
    print(scrollable_frame_names)

read_button = ctk.CTkButton(second_frame, text="Read", command=read_data)

#############################################################################################

# settings

select_frame_by_name("home")
change_appearance_mode_event("Dark")
# Place input fields and button on form

save_button.grid(row=5, column=1, padx=20, pady=10,)
read_button.grid(row=5, column=0, padx=10, pady=(0, 20))

root.mainloop()
```