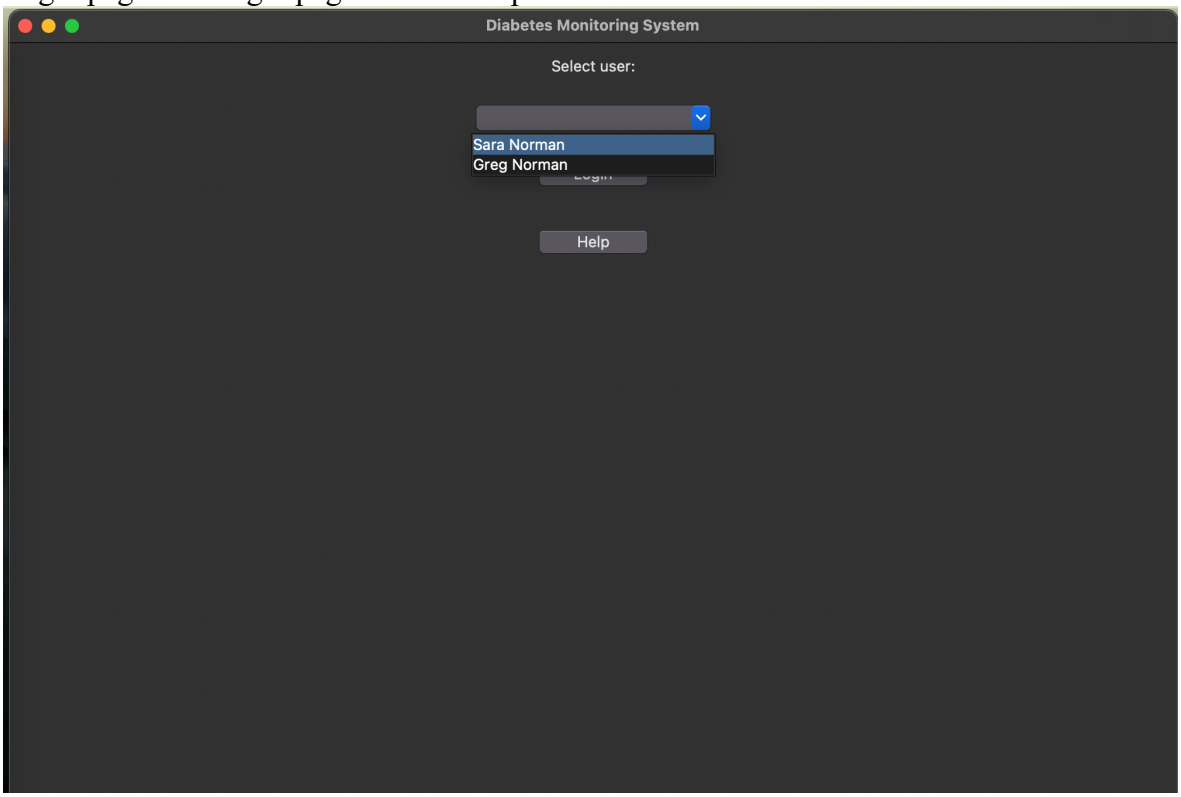
Emily Ridge
HCI
OU ID: 113585745

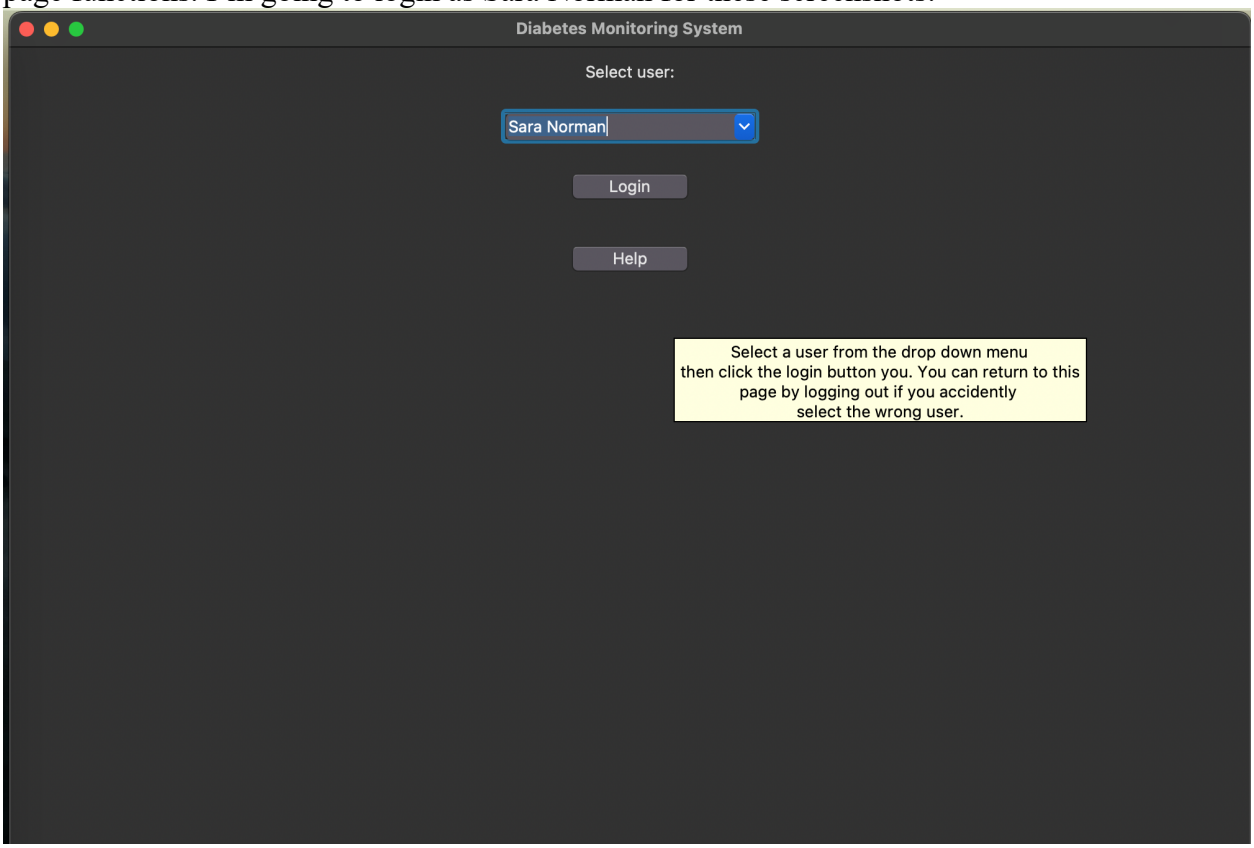
Diabetes Monitoring System

Functionality:

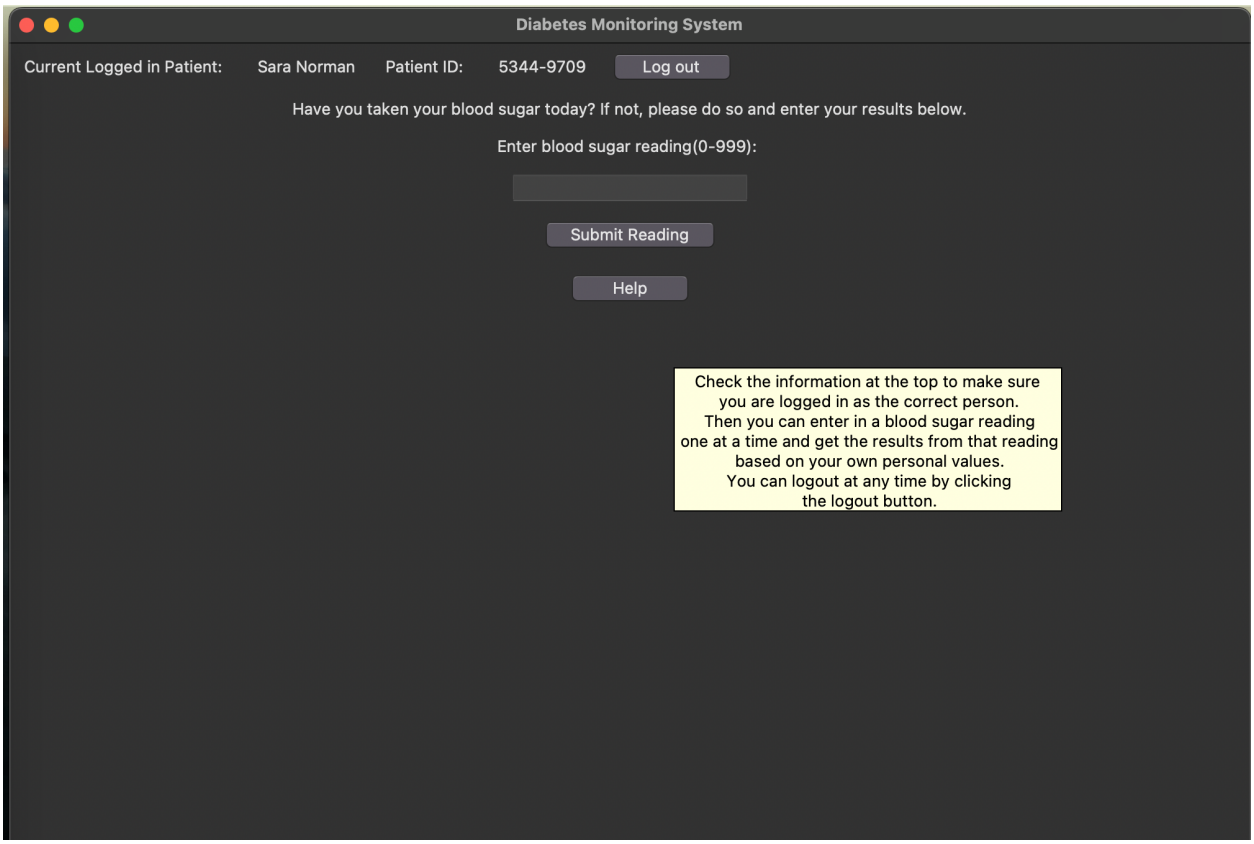
Login page: The login page offers a drop-down menu where users can select their name to log in:



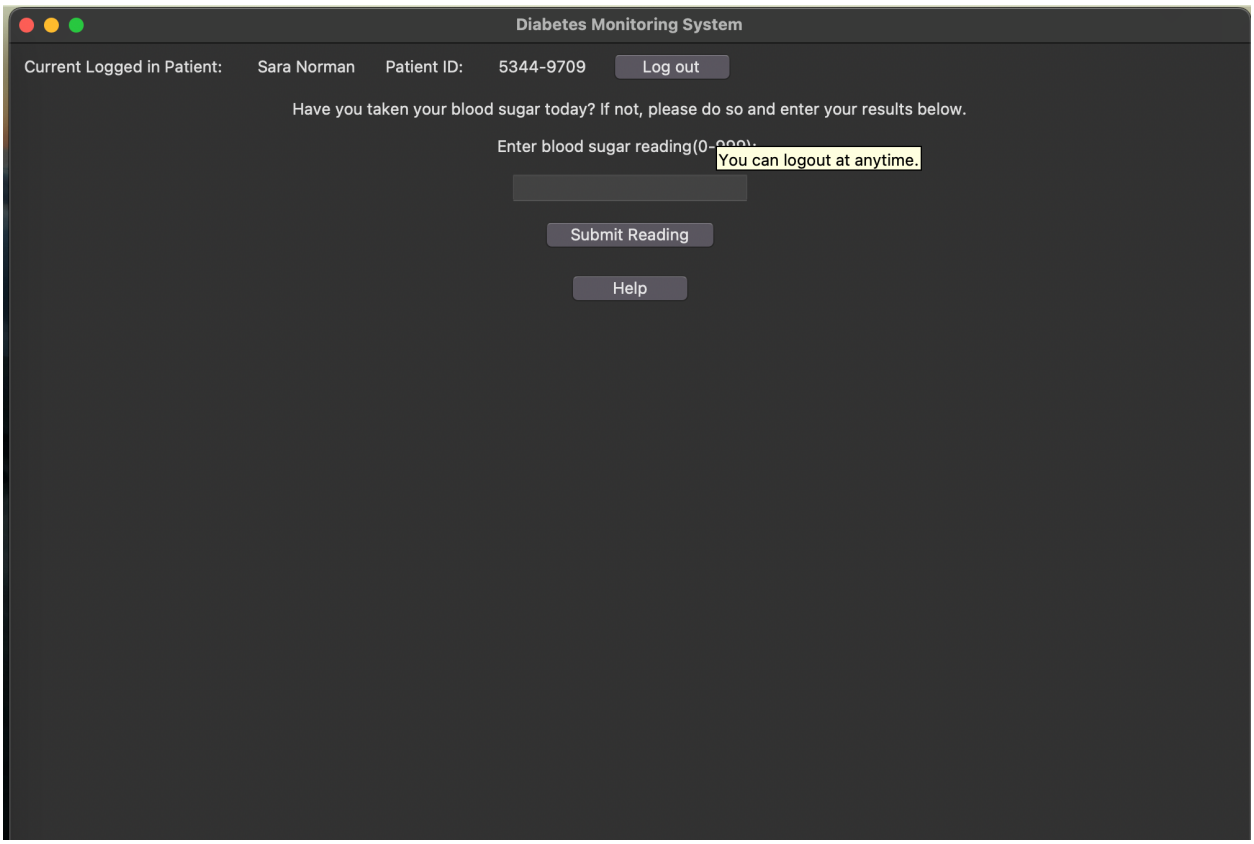
It also has a help option that the user can hover over, and it gives them a description of how the page functions. I'm going to login as Sara Norman for these screenshots:



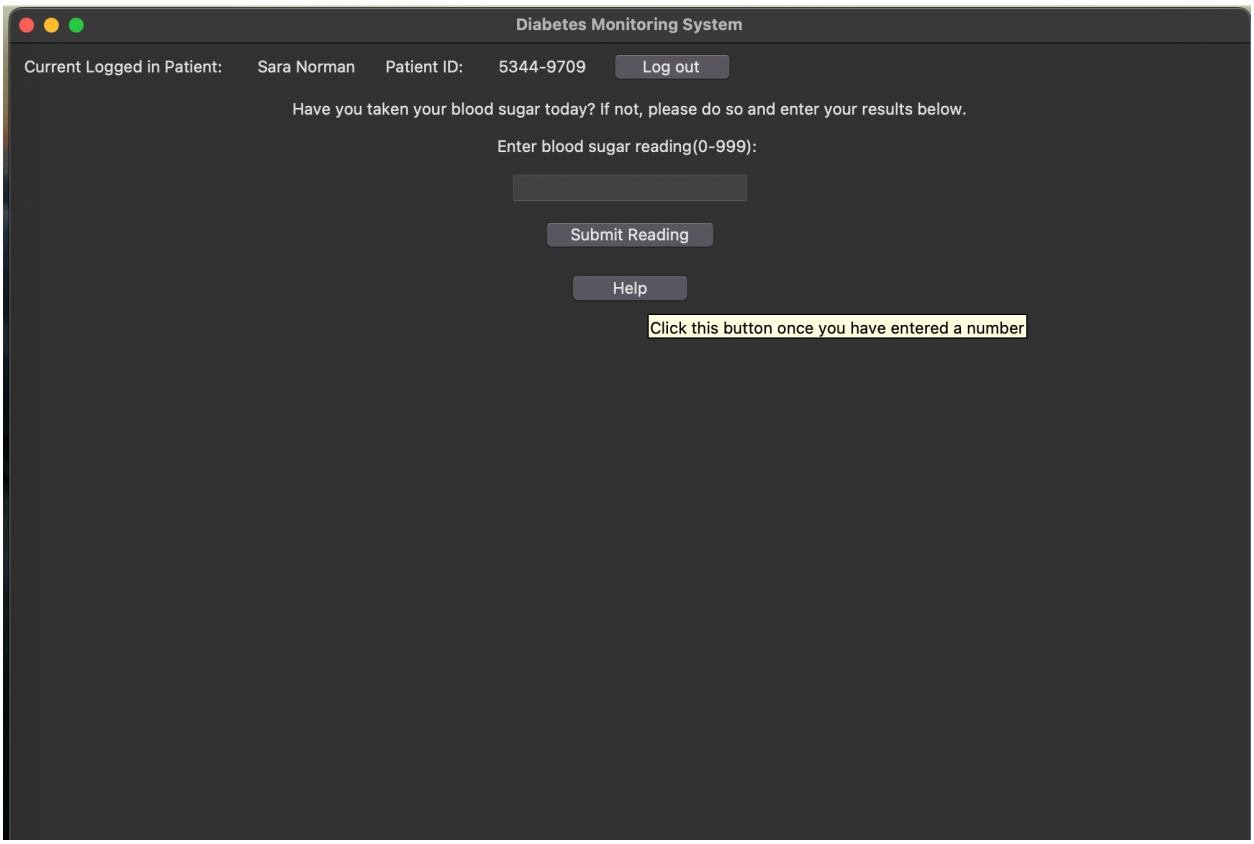
Once the login button is clicked the user is directed to the diabetes monitoring page, that also has the help option, this is what appear when you hover over the help:



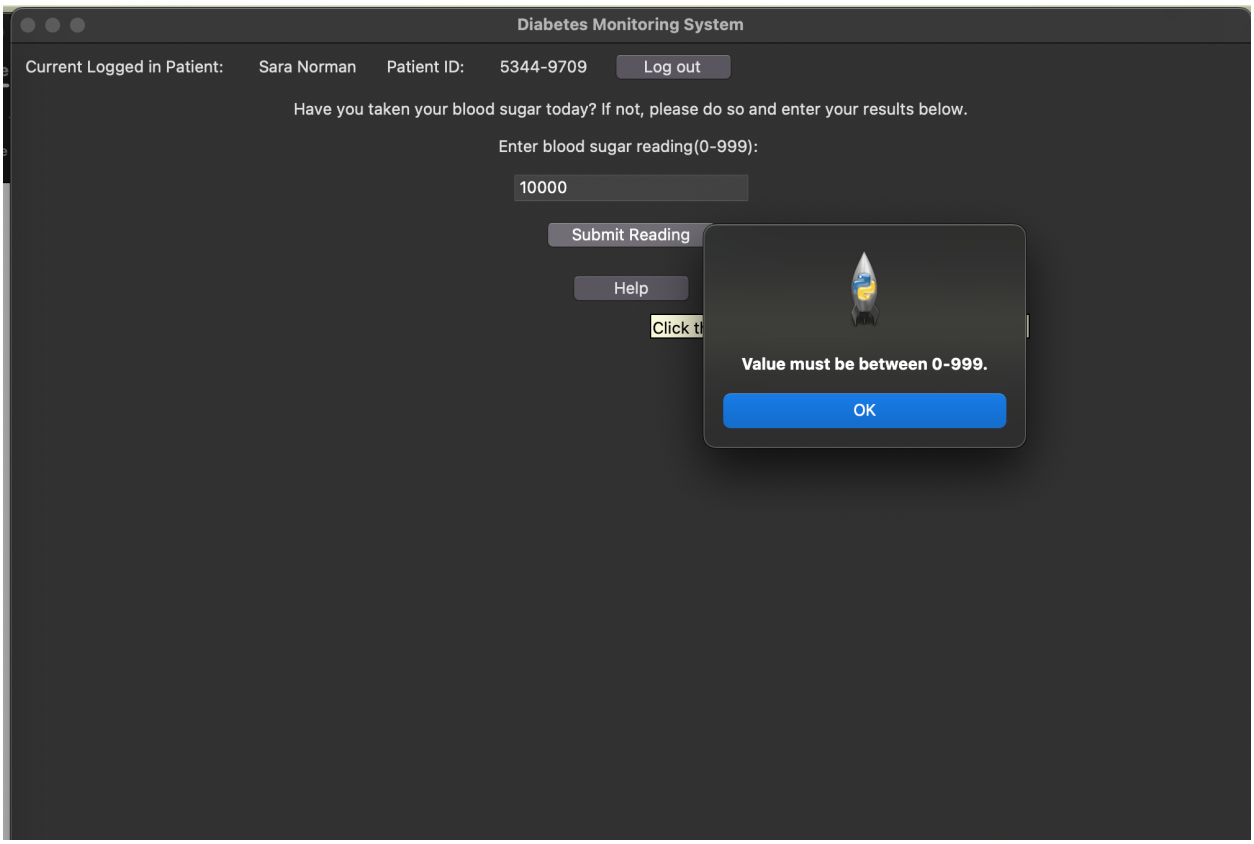
As you can see the user that is logged in, in this case Sara Norman, has their name and patient ID displayed on the top of the page. As well as a logout button with a hover option that displays:



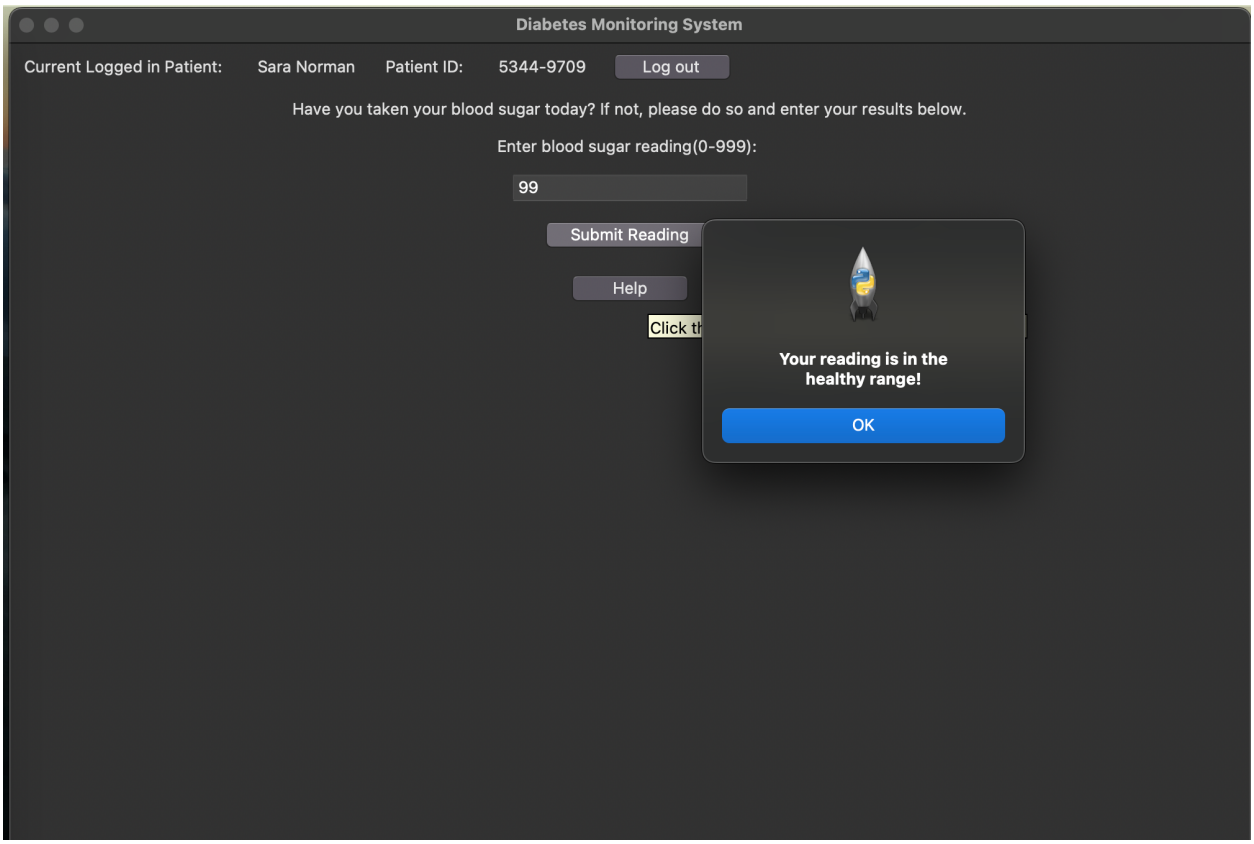
The submit button has a hover action that displays:

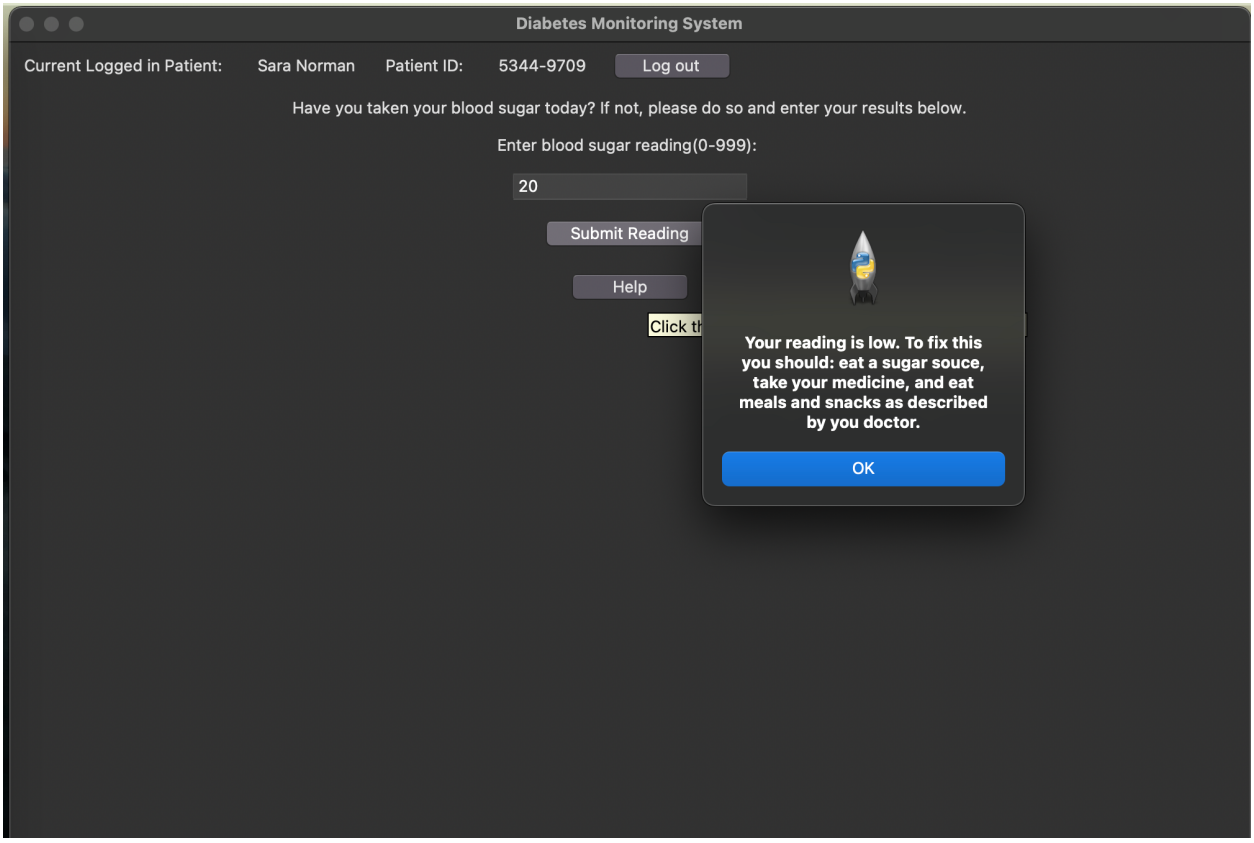


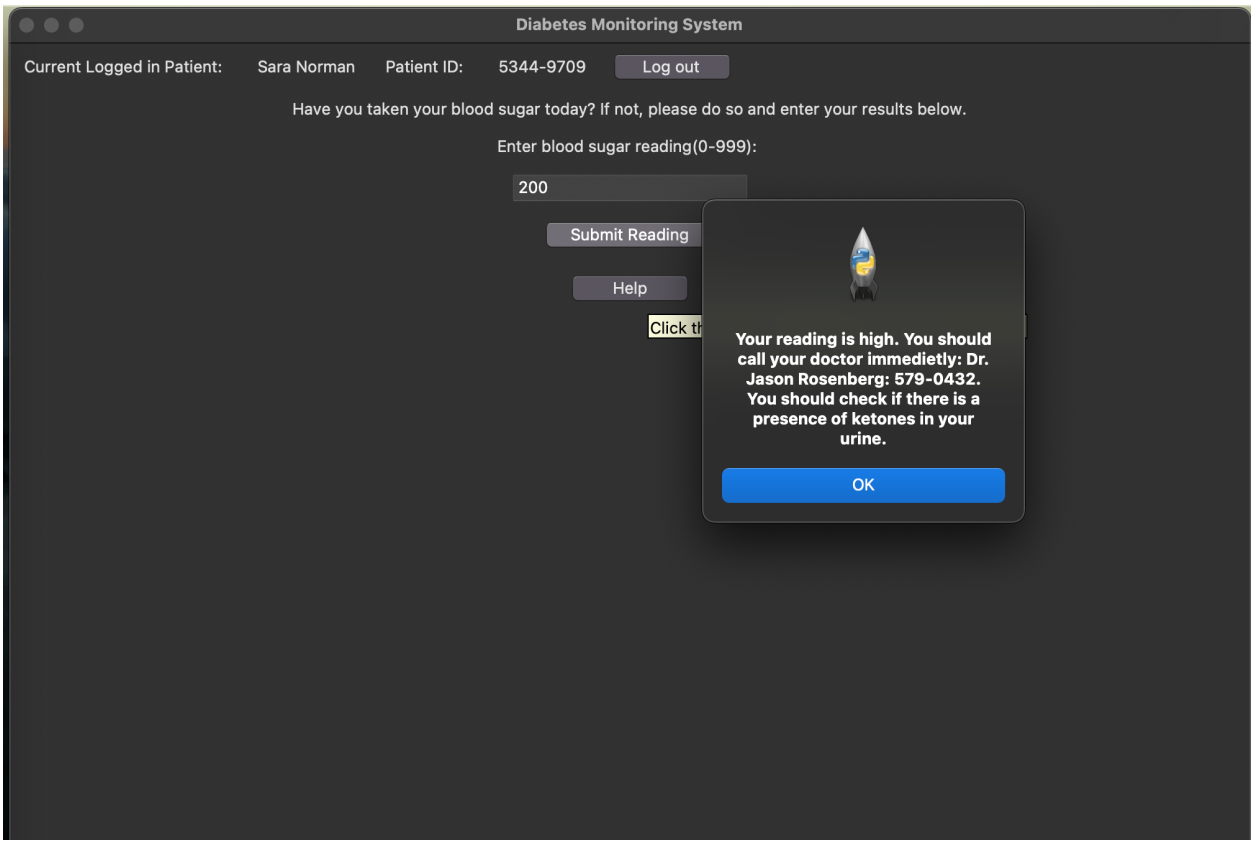
The user can enter in a value between 0-200, this is error checked in the code and if they fail to meet specifications with their input an error message appears, and they get another chance to enter a valid input:



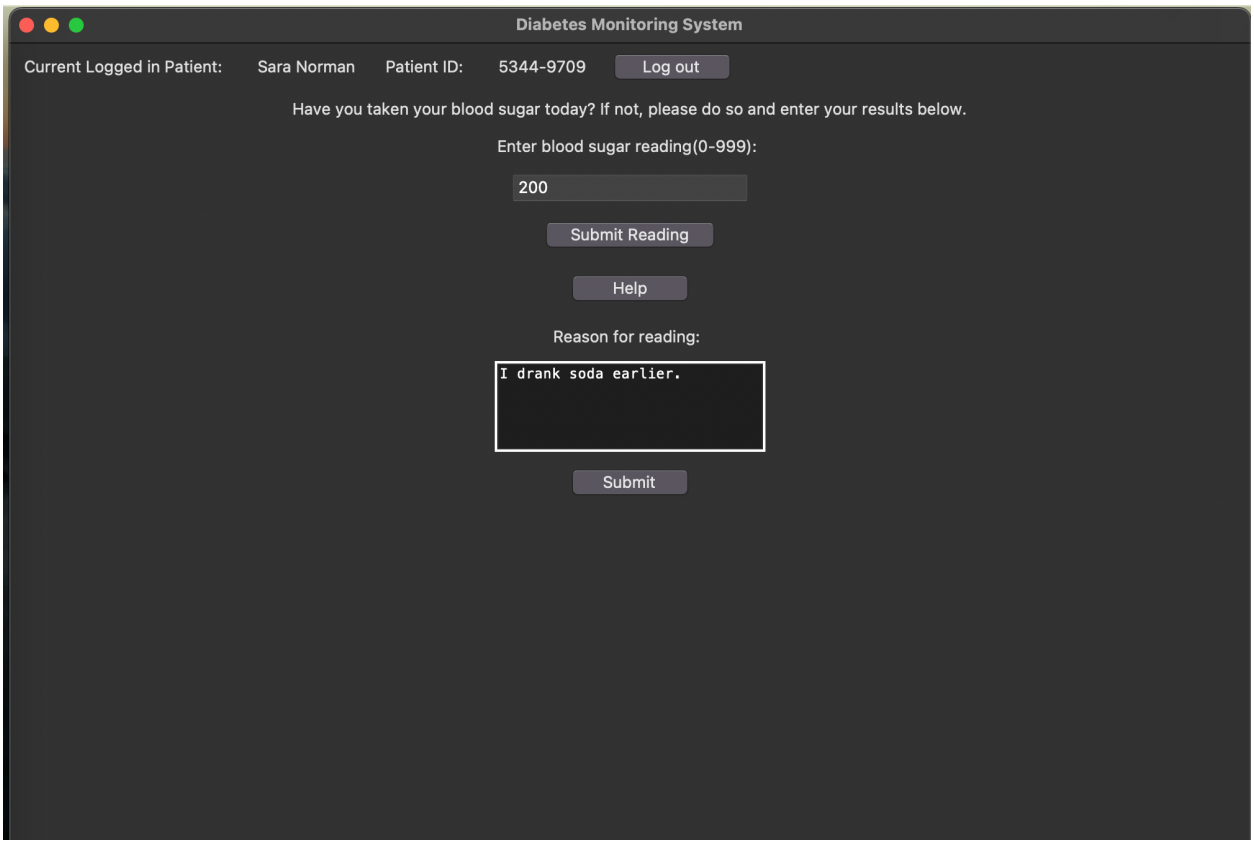
Once a valid input in entered, they will get either the normal, low, or high alert:


Diabetes Monitoring System
Current Logged in Patient:
Sara Norman
Patient ID:
5344-9709
Log out
Have you taken your blood sugar today? If not, please do so and enter your results below.
Enter blood sugar reading(0-999):
99
Submit Reading
Help
Click th
Your reading is in the healthy range!
OK


Diabetes Monitoring System
Current Logged in Patient:
Sara Norman
Patient ID:
5344-9709
Log out
Have you taken your blood sugar today? If not, please do so and enter your results below.
Enter blood sugar reading(0-999):
20
Submit Reading
Help
Click th
Your reading is low. To fix this you should: eat a sugar souce, take your medicine, and eat meals and snacks as described by you doctor.
OK



For the low and high alerts, after they click OK a text box appears for them to enter in a reason for this reading:

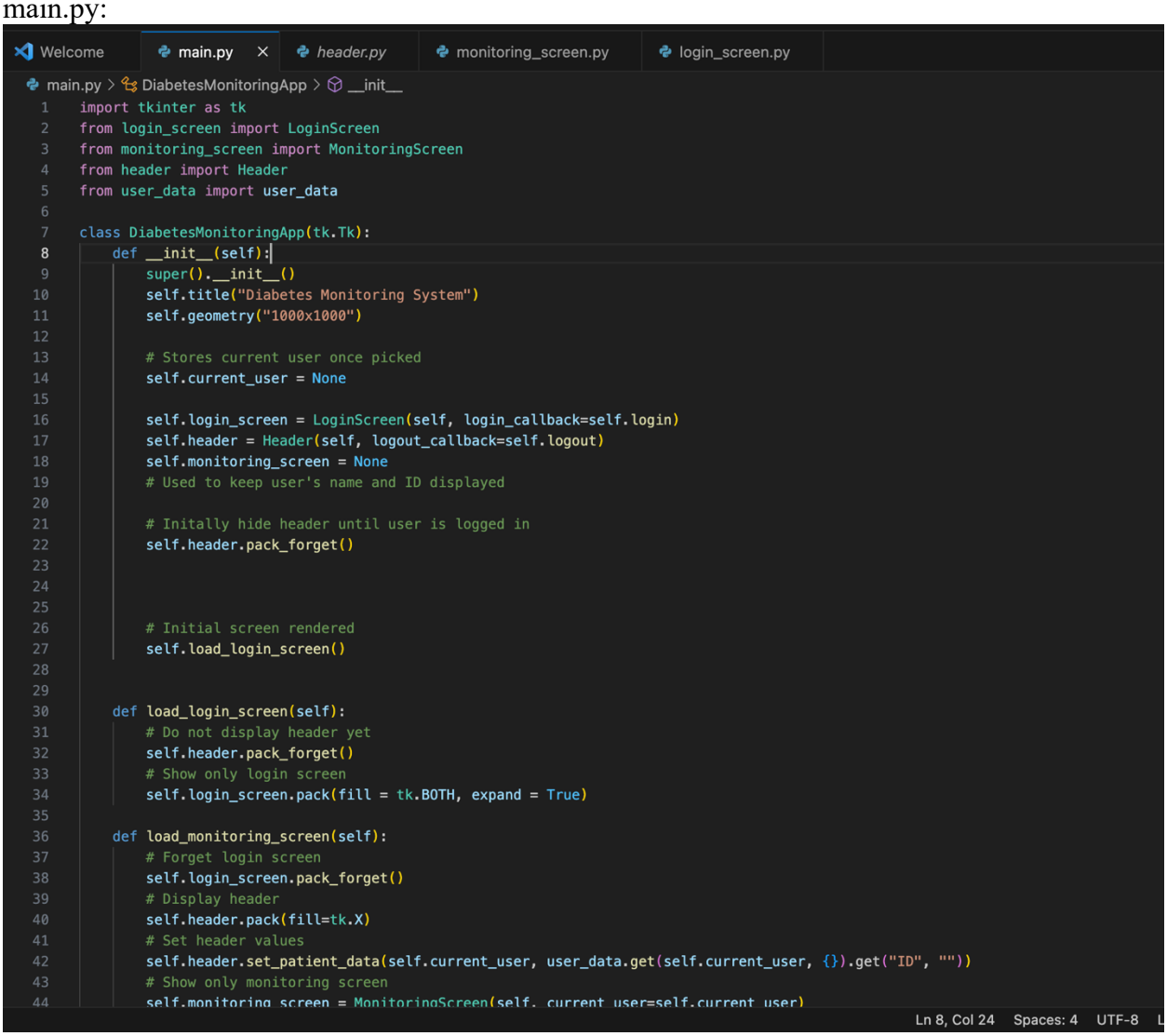


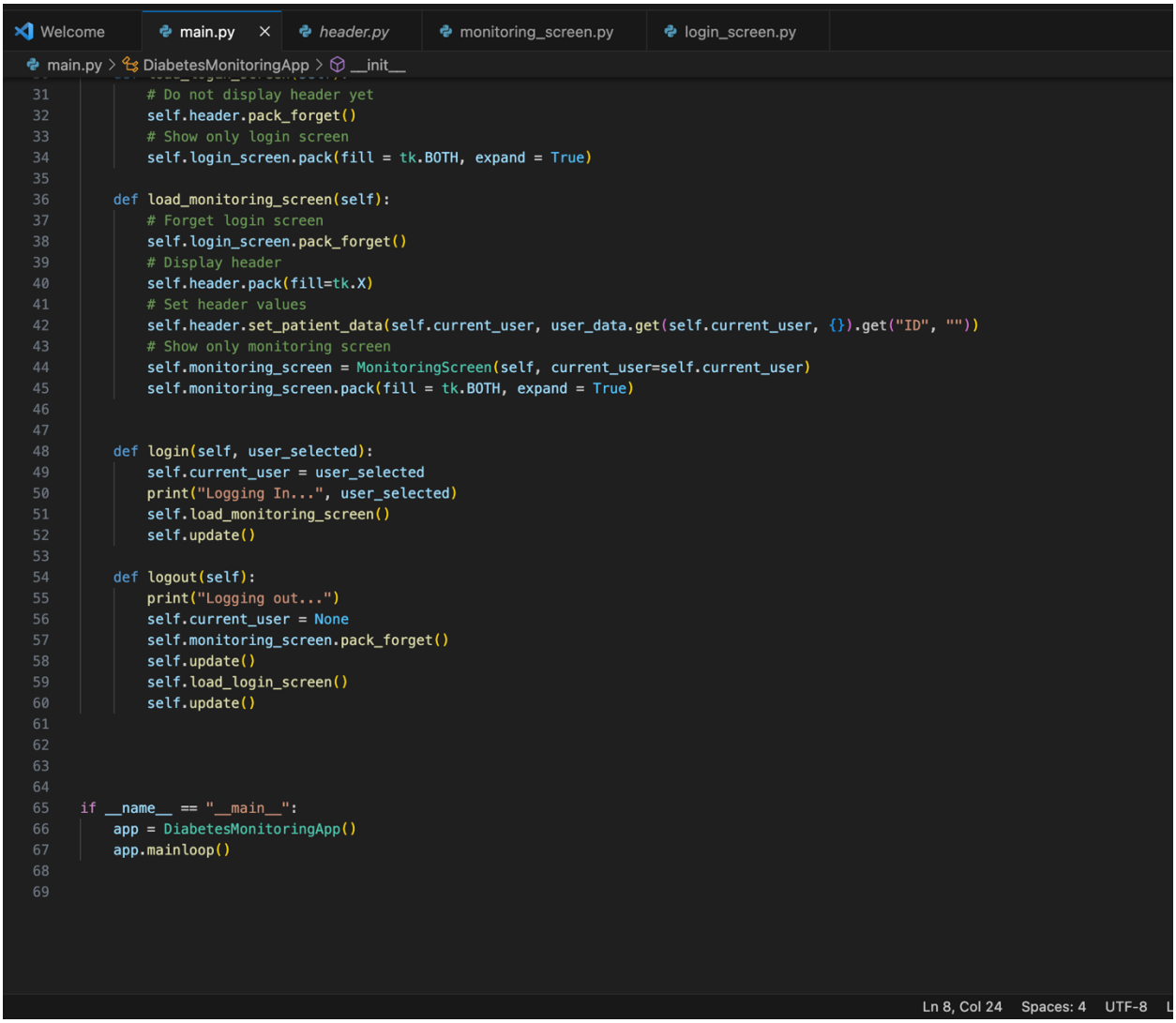
Once they hit submit, the textbox disappears, and they can submit another reading if they want. Or they can click logout and be taken back to the original login page.

Code Implementation:

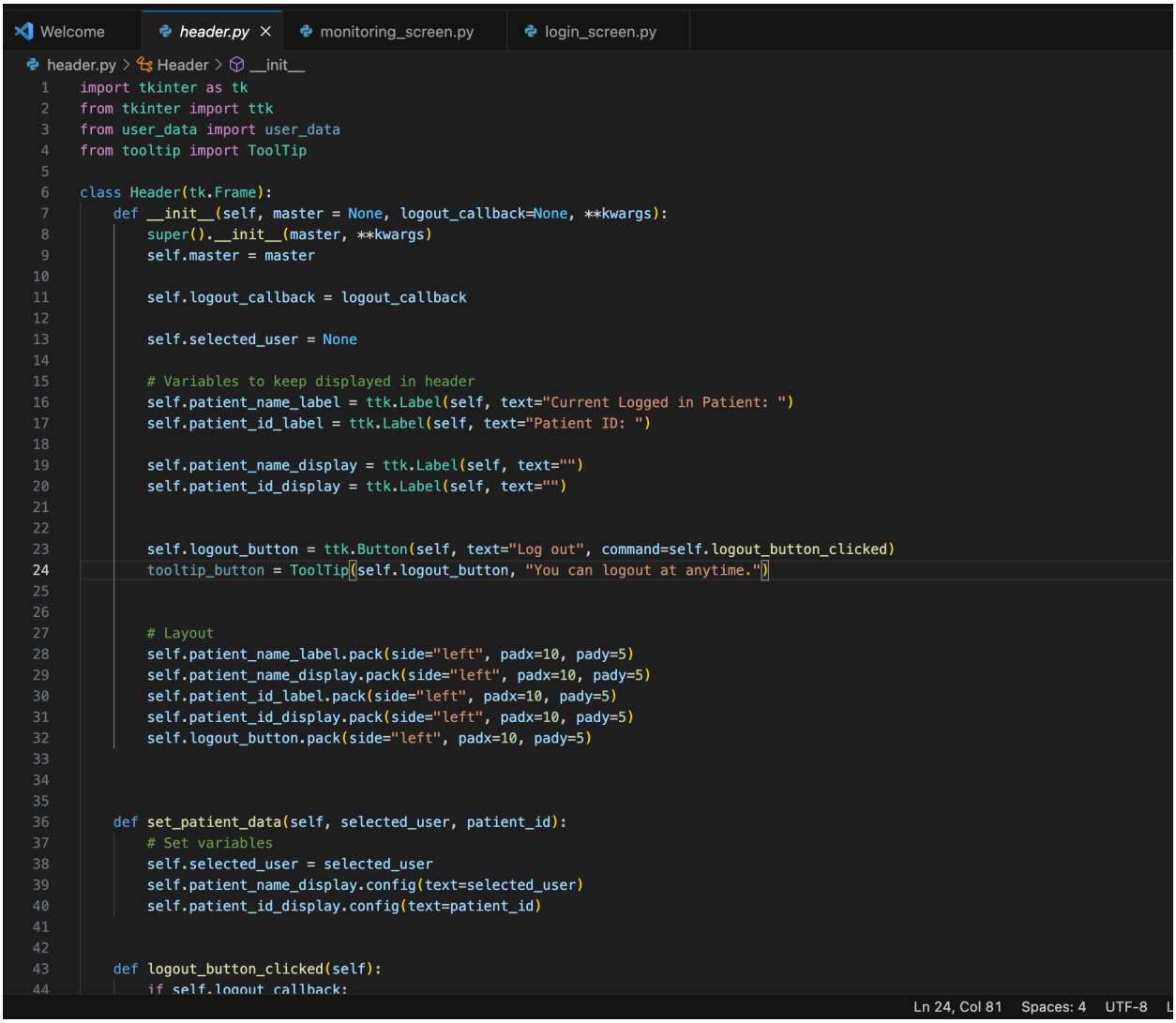
GitHub Repo: https://github.com/emilyridge/HCI_Diabeties_Monitoring.git

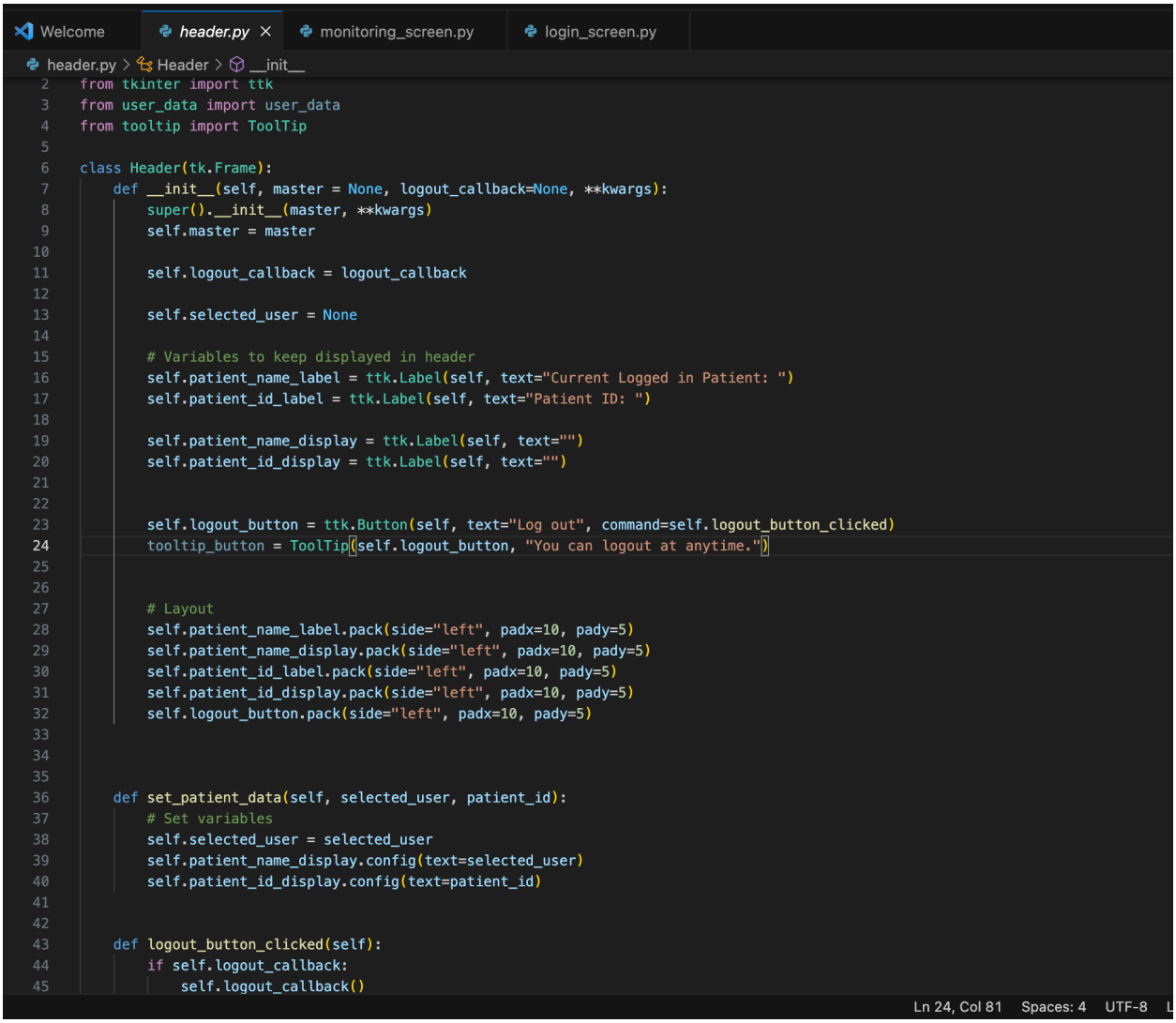
main.py:


Welcome
main.py
header.py
monitoring_screen.py
login_screen.py
main.py > DiabetesMonitoringApp > __init__
import tkinter as tk
from login_screen import LoginScreen
from monitoring_screen import MonitoringScreen
from header import Header
from user_data import user_data
class DiabetesMonitoringApp(tk.Tk):
def __init__(self):
super().__init__()
self.title("Diabetes Monitoring System")
self.geometry("1000x1000")
# Stores current user once picked
self.current_user = None
self.login_screen = LoginScreen(self, login_callback=self.login)
self.header = Header(self, logout_callback=self.logout)
self.monitoring_screen = None
# Used to keep user's name and ID displayed
# Initally hide header until user is logged in
self.header.pack_forget()
# Initial screen rendered
self.load_login_screen()
def load_login_screen(self):
# Do not display header yet
self.header.pack_forget()
# Show only login screen
self.login_screen.pack(fill = tk.BOTH, expand = True)
def load_monitoring_screen(self):
# Forget login screen
self.login_screen.pack_forget()
# Display header
self.header.pack(fill=tk.X)
# Set header values
self.header.set_patient_data(self.current_user, user_data.get(self.current_user, {}).get("ID", ""))
# Show only monitoring screen
Ln 8, Col 24
Spaces: 4
UTF-8

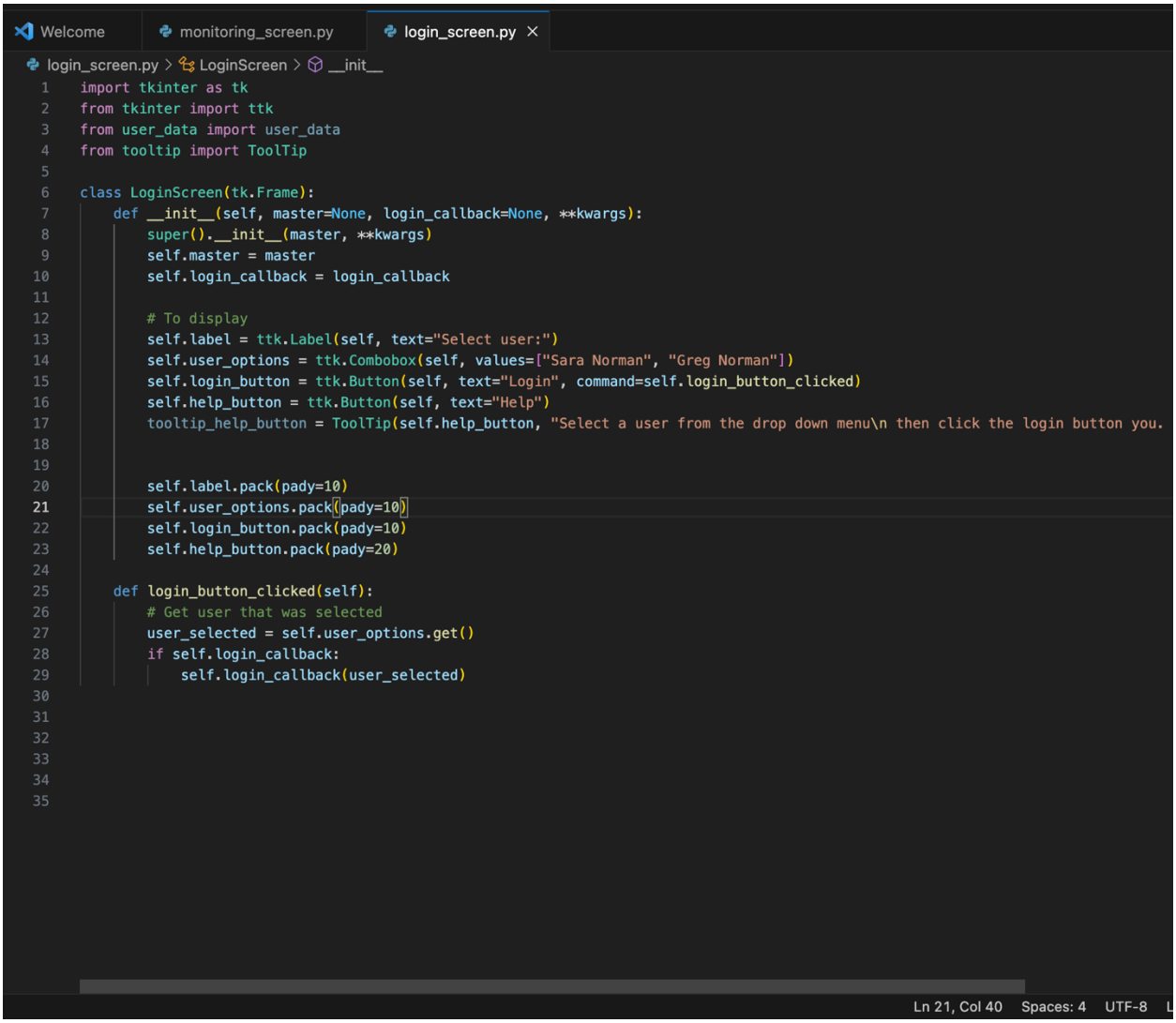

Welcome
main.py
header.py
monitoring_screen.py
login_screen.py
main.py > DiabetesMonitoringApp > __init__
# Do not display header yet
self.header.pack_forget()
# Show only login screen
self.login_screen.pack(fill = tk.BOTH, expand = True)
def load_monitoring_screen(self):
# Forget login screen
self.login_screen.pack_forget()
# Display header
self.header.pack(fill=tk.X)
# Set header values
self.header.set_patient_data(self.current_user, user_data.get(self.current_user, {}).get("ID", ""))
# Show only monitoring screen
self.monitoring_screen = MonitoringScreen(self, current_user=self.current_user)
self.monitoring_screen.pack(fill = tk.BOTH, expand = True)
def login(self, user_selected):
self.current_user = user_selected
print("Logging In...", user_selected)
self.load_monitoring_screen()
self.update()
def logout(self):
print("Logging out...")
self.current_user = None
self.monitoring_screen.pack_forget()
self.update()
self.load_login_screen()
self.update()
if __name__ == "__main__":
app = DiabetesMonitoringApp()
app.mainloop()
Ln 8, Col 24
Spaces: 4
UTF-8

header.py:

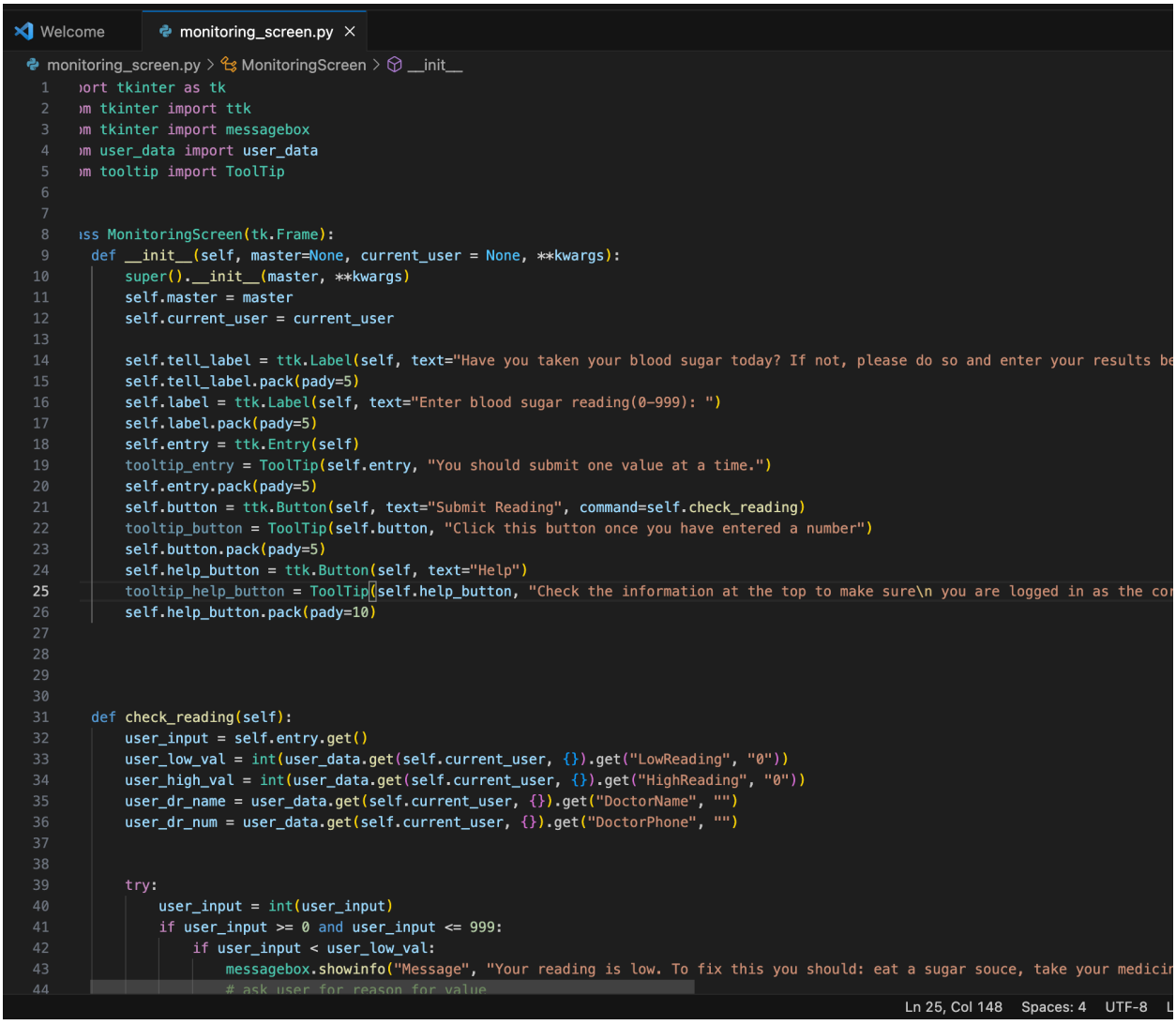

Welcome
header.py
monitoring_screen.py
login_screen.py
header.py > Header > __init__
1 import tkinter as tk
2 from tkinter import ttk
3 from user_data import user_data
4 from tooltip import ToolTip
5
6 class Header(tk.Frame):
7 def __init__(self, master = None, logout_callback=None, **kwargs):
8 super().__init__(master, **kwargs)
9 self.master = master
10
11 self.logout_callback = logout_callback
12
13 self.selected_user = None
14
15 # Variables to keep displayed in header
16 self.patient_name_label = ttk.Label(self, text="Current Logged in Patient: ")
17 self.patient_id_label = ttk.Label(self, text="Patient ID: ")
18
19 self.patient_name_display = ttk.Label(self, text="")
20 self.patient_id_display = ttk.Label(self, text="")
21
22
23 self.logout_button = ttk.Button(self, text="Log out", command=self.logout_button_clicked)
24 tooltip_button = ToolTip(self.logout_button, "You can logout at anytime.")
25
26
27 # Layout
28 self.patient_name_label.pack(side="left", padx=10, pady=5)
29 self.patient_name_display.pack(side="left", padx=10, pady=5)
30 self.patient_id_label.pack(side="left", padx=10, pady=5)
31 self.patient_id_display.pack(side="left", padx=10, pady=5)
32 self.logout_button.pack(side="left", padx=10, pady=5)
33
34
35
36 def set_patient_data(self, selected_user, patient_id):
37 # Set variables
38 self.selected_user = selected_user
39 self.patient_name_display.config(text=selected_user)
40 self.patient_id_display.config(text=patient_id)
41
42
43 def logout_button_clicked(self):
44 if self.logout_callback:
Ln 24, Col 81
Spaces: 4
UTF-8

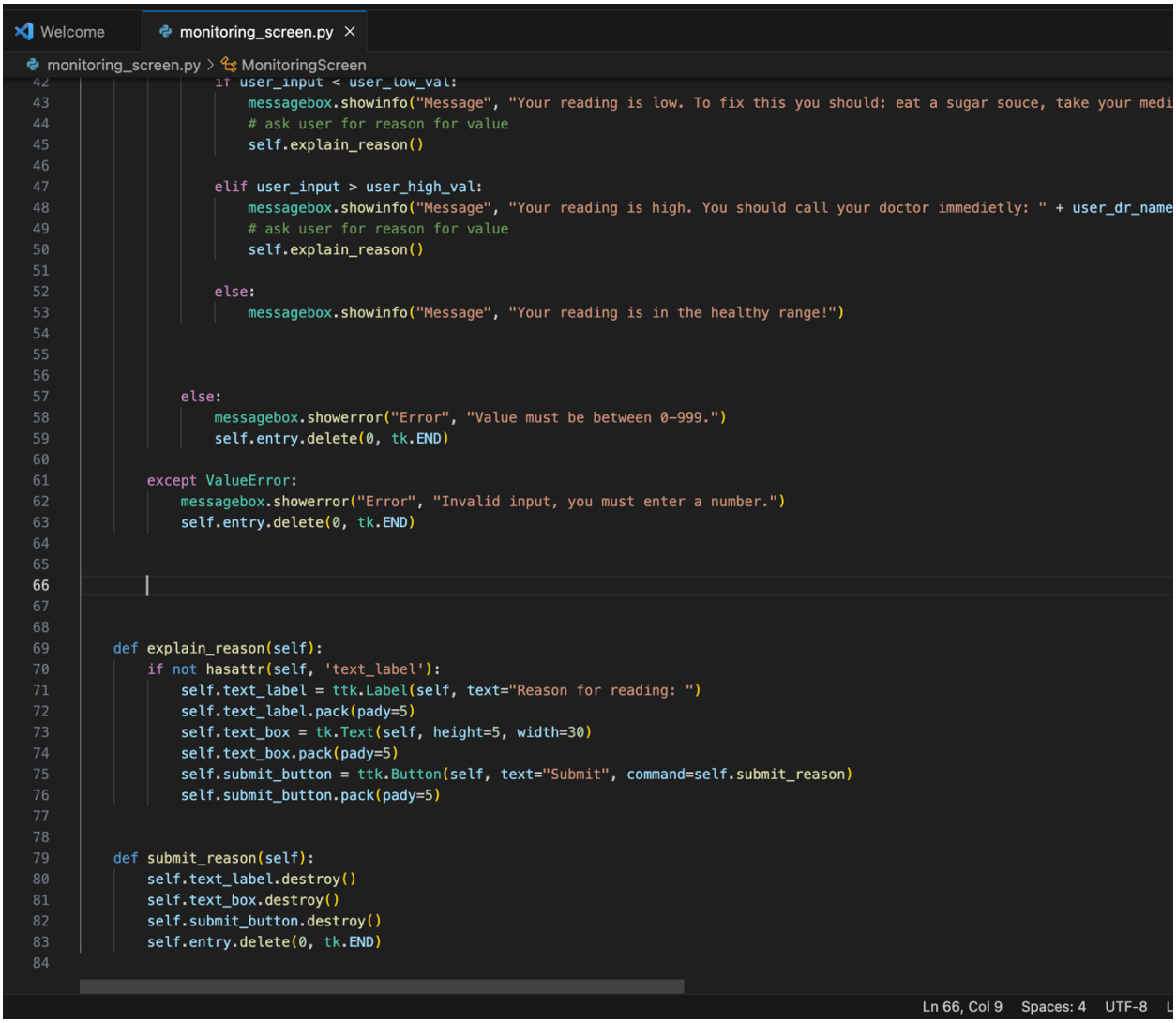


login_screen.py:

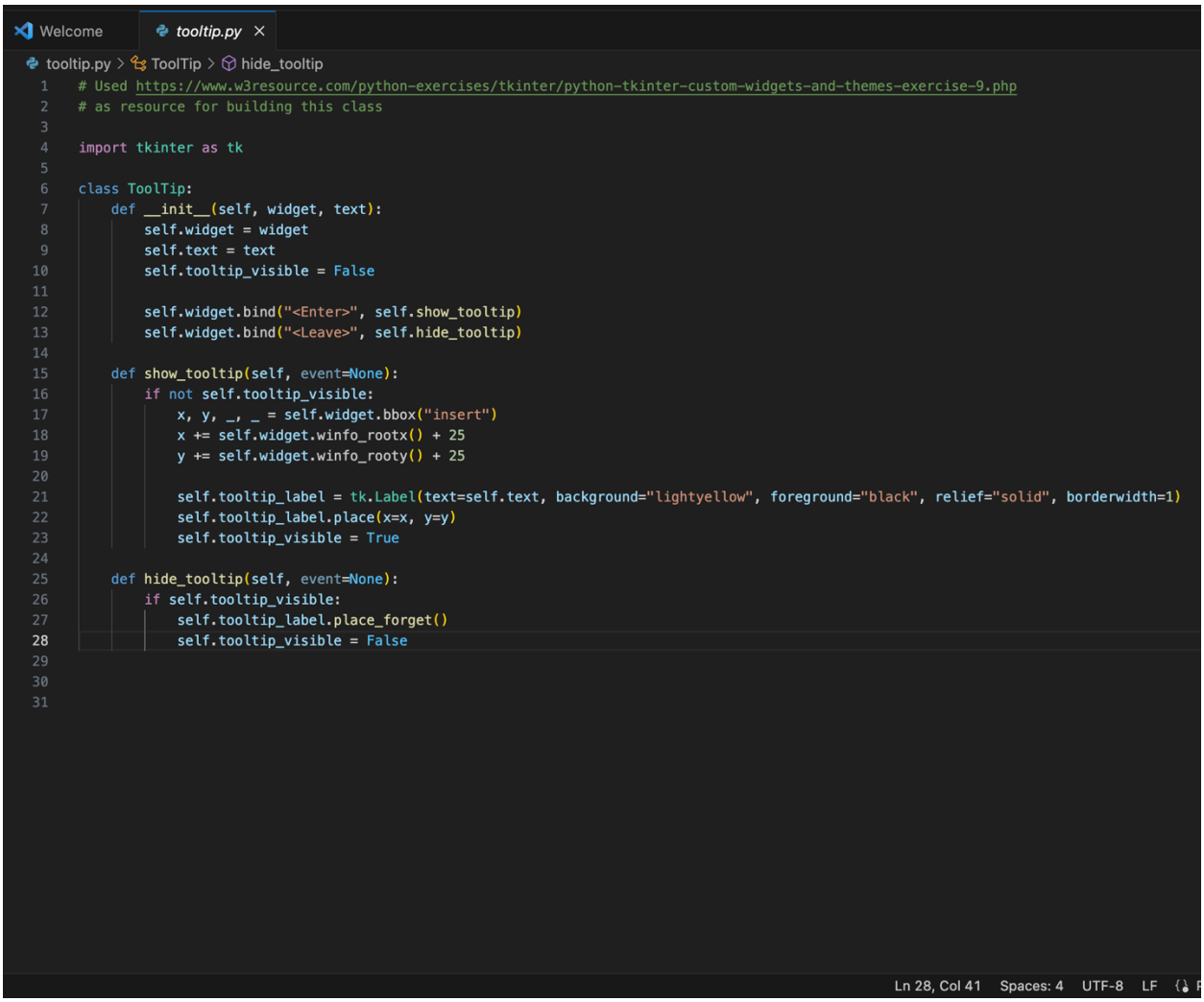


monitoring_screen.py:

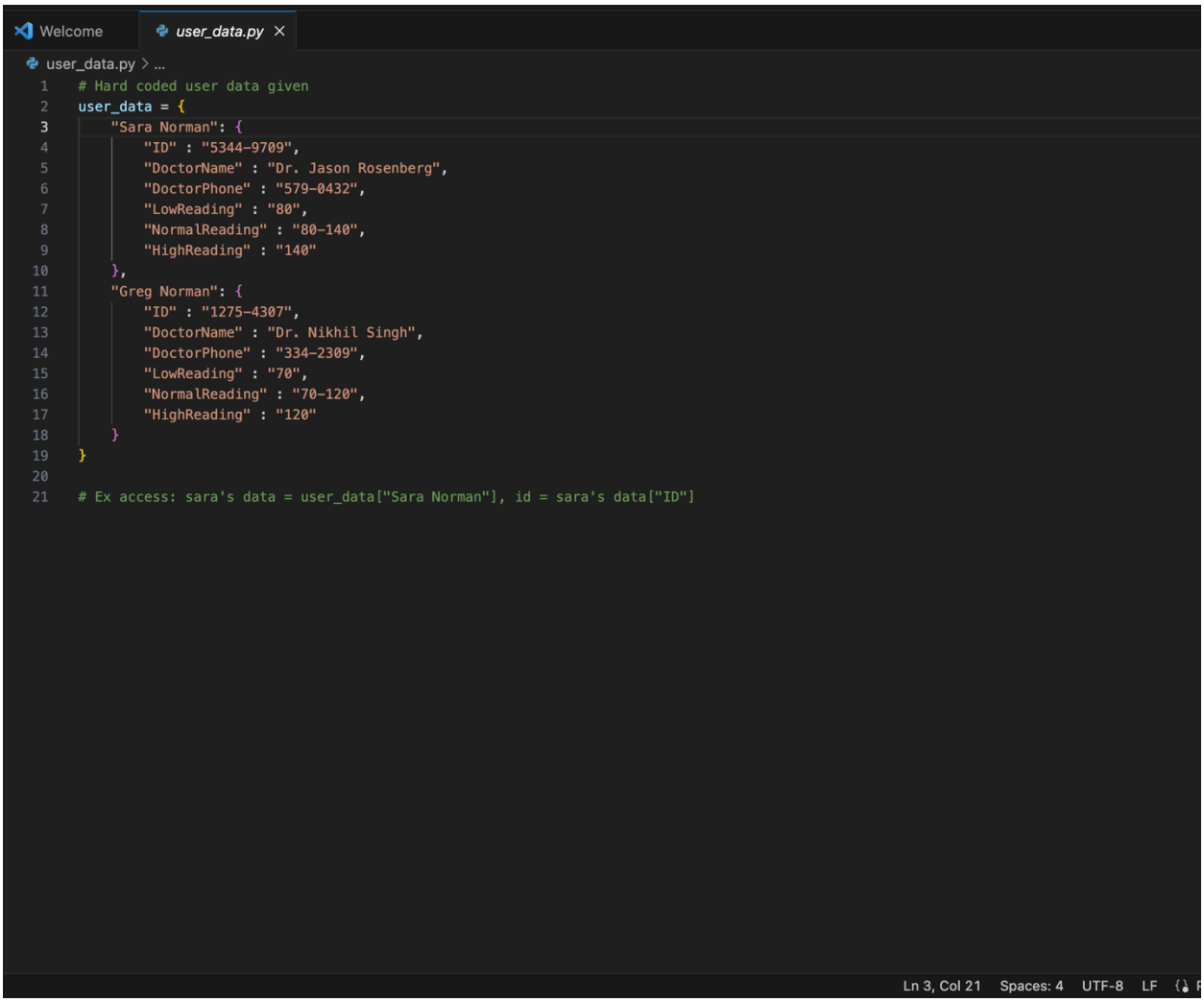

Welcome
monitoring_screen.py
monitoring_screen.py > MonitoringScreen > __init__
ort tkinter as tk
m tkinter import ttk
m tkinter import messagebox
m user_data import user_data
m tooltip import ToolTip
ss MonitoringScreen(tk.Frame):
def __init__(self, master=None, current_user = None, **kwargs):
super().__init__(master, **kwargs)
self.master = master
self.current_user = current_user
self.tell_label = ttk.Label(self, text="Have you taken your blood sugar today? If not, please do so and enter your results be
self.tell_label.pack(pady=5)
self.label = ttk.Label(self, text="Enter blood sugar reading(0-999): ")
self.label.pack(pady=5)
self.entry = ttk.Entry(self)
tooltip_entry = ToolTip(self.entry, "You should submit one value at a time.")
self.entry.pack(pady=5)
self.button = ttk.Button(self, text="Submit Reading", command=self.check_reading)
tooltip_button = ToolTip(self.button, "Click this button once you have entered a number")
self.button.pack(pady=5)
self.help_button = ttk.Button(self, text="Help")
tooltip_help_button = ToolTip(self.help_button, "Check the information at the top to make sure\n you are logged in as the co
self.help_button.pack(pady=10)
def check_reading(self):
user_input = self.entry.get()
user_low_val = int(user_data.get(self.current_user, {}).get("LowReading", "0"))
user_high_val = int(user_data.get(self.current_user, {}).get("HighReading", "0"))
user_dr_name = user_data.get(self.current_user, {}).get("DoctorName", "")
user_dr_num = user_data.get(self.current_user, {}).get("DoctorPhone", "")
try:
user_input = int(user_input)
if user_input >= 0 and user_input <= 999:
if user_input < user_low_val:
messagebox.showinfo("Message", "Your reading is low. To fix this you should: eat a sugar souce, take your medicin
ask user for reason for value
Ln 25, Col 148
Spaces: 4
UTF-8


Welcome
monitoring_screen.py
monitoring_screen.py > MonitoringScreen
if user_input < user_low_val:
messagebox.showinfo("Message", "Your reading is low. To fix this you should: eat a sugar souce, take your medi
# ask user for reason for value
self.explain_reason()
elif user_input > user_high_val:
messagebox.showinfo("Message", "Your reading is high. You should call your doctor immedietly: " + user_dr_name
# ask user for reason for value
self.explain_reason()
else:
messagebox.showinfo("Message", "Your reading is in the healthy range!")
else:
messagebox.showerror("Error", "Value must be between 0-999.")
self.entry.delete(0, tk.END)
except ValueError:
messagebox.showerror("Error", "Invalid input, you must enter a number.")
self.entry.delete(0, tk.END)
def explain_reason(self):
if not hasattr(self, 'text_label'):
self.text_label = ttk.Label(self, text="Reason for reading: ")
self.text_label.pack(pady=5)
self.text_box = tk.Text(self, height=5, width=30)
self.text_box.pack(pady=5)
self.submit_button = ttk.Button(self, text="Submit", command=self.submit_reason)
self.submit_button.pack(pady=5)
def submit_reason(self):
self.text_label.destroy()
self.text_box.destroy()
self.submit_button.destroy()
self.entry.delete(0, tk.END)
Ln 66, Col 9
Spaces: 4
UTF-8

tooltip.py:


Welcome
tooltip.py
tooltip.py > ToolTip > hide_tooltip
# Used https://www.w3resource.com/python-exercises/tkinter/python-tkinter-custom-widgets-and-themes-exercise-9.php
# as resource for building this class
import tkinter as tk
class ToolTip:
def __init__(self, widget, text):
self.widget = widget
self.text = text
self.tooltip_visible = False
self.widget.bind("<Enter>", self.show_tooltip)
self.widget.bind("<Leave>", self.hide_tooltip)
def show_tooltip(self, event=None):
if not self.tooltip_visible:
x, y, _, _ = self.widget.bbox("insert")
x += self.widget.winfo_rootx() + 25
y += self.widget.winfo_rooty() + 25
self.tooltip_label = tk.Label(text=self.text, background="lightyellow", foreground="black", relief="solid", borderwidth=1)
self.tooltip_label.place(x=x, y=y)
self.tooltip_visible = True
def hide_tooltip(self, event=None):
if self.tooltip_visible:
self.tooltip_label.place_forget()
self.tooltip_visible = False
Ln 28, Col 41
Spaces: 4
UTF-8
LF

user_data.py:


Welcome
user_data.py
user_data.py > ...
# Hard coded user data given
user_data = {
"Sara Norman": {
"ID" : "5344-9709",
"DoctorName" : "Dr. Jason Rosenberg",
"DoctorPhone" : "579-0432",
"LowReading" : "80",
"NormalReading" : "80-140",
"HighReading" : "140"
},
"Greg Norman": {
"ID" : "1275-4307",
"DoctorName" : "Dr. Nikhil Singh",
"DoctorPhone" : "334-2309",
"LowReading" : "70",
"NormalReading" : "70-120",
"HighReading" : "120"
}
}
# Ex access: sara's data = user_data["Sara Norman"], id = sara's data["ID"]
Ln 3, Col 21
Spaces: 4
UTF-8
LF