

Body Temperature Monitoring System - Project Report

This project is a Python-based Body Temperature Monitoring System designed for real-time temperature tracking using a graphical interface. It is useful for basic healthcare monitoring, simulations, and educational purposes.

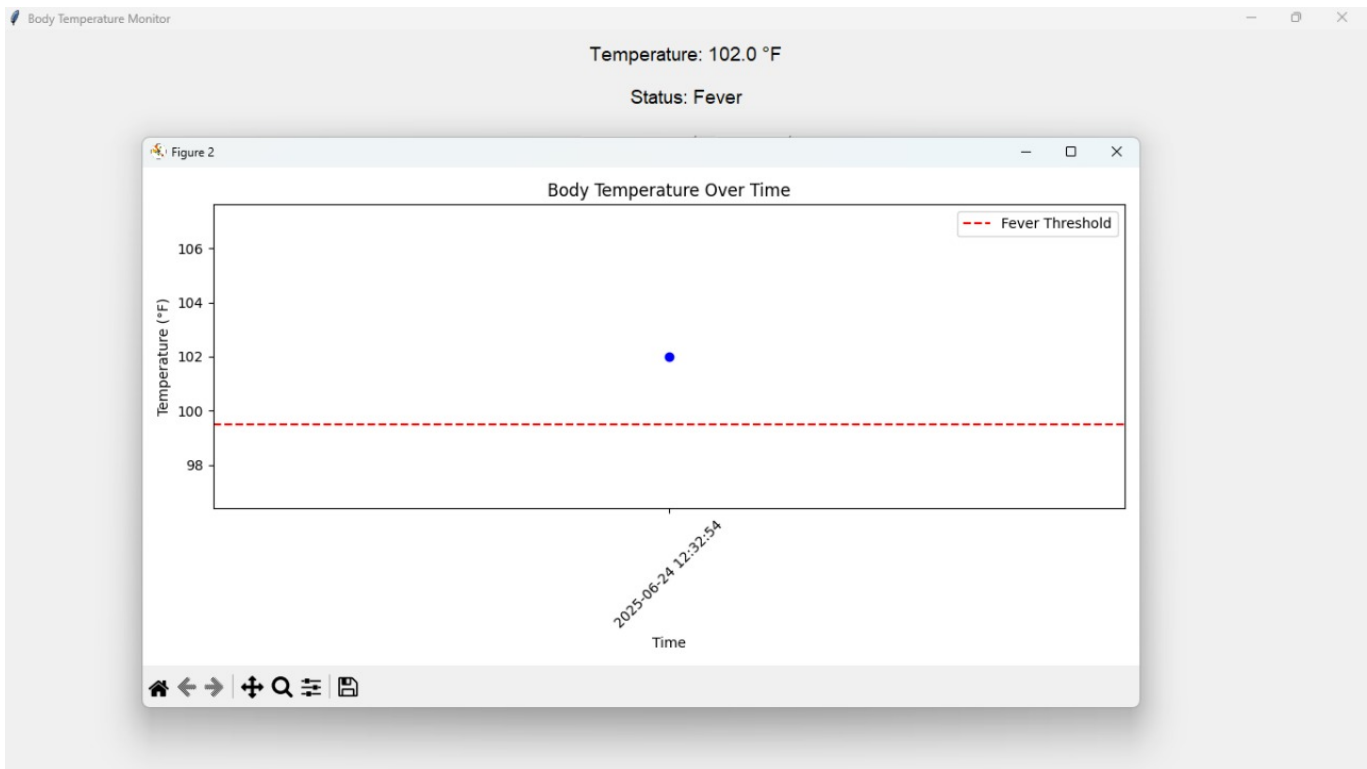
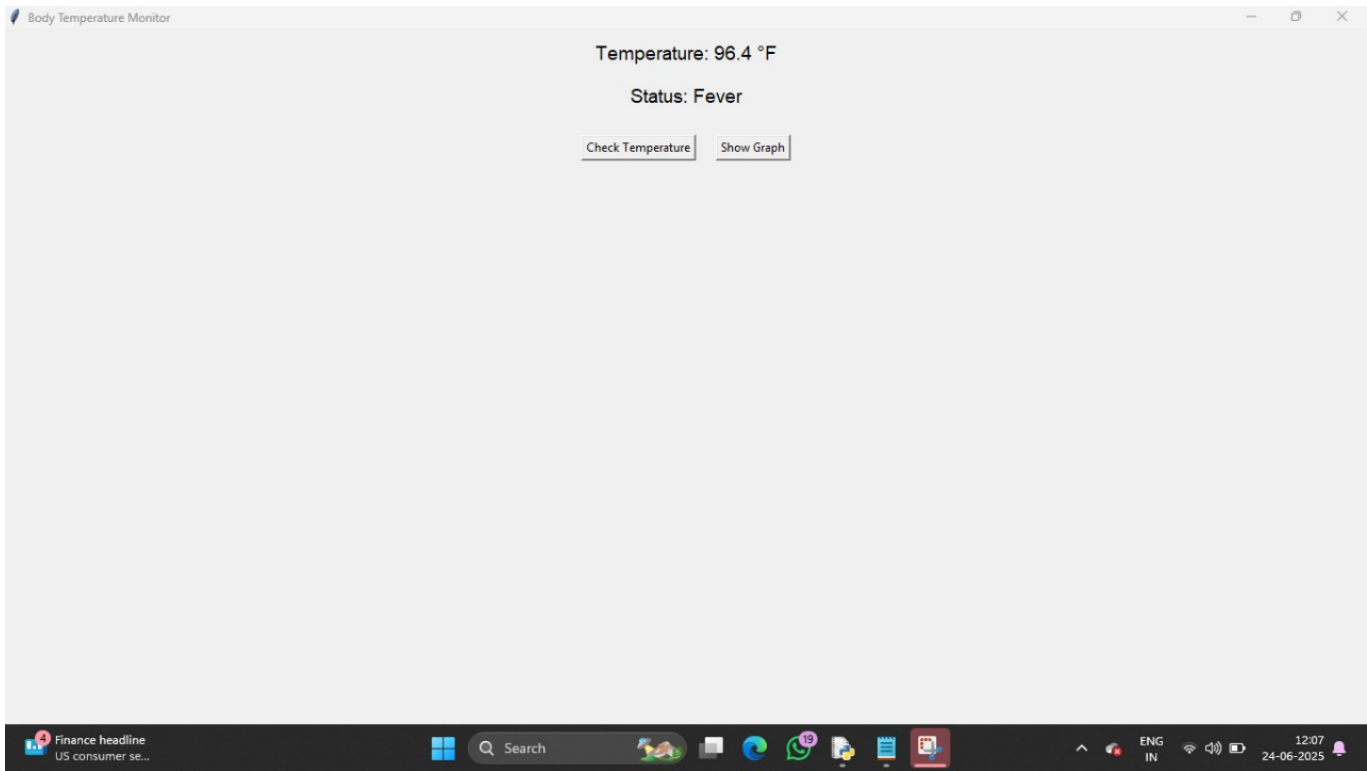
Key Features:

- Simulates realistic temperature readings.
- Detects abnormal body temperature ("Fever") and issues alerts.
- Automatically logs data in CSV format for analysis.
- Provides visual trends using line graphs.
- Built using Tkinter (GUI), Matplotlib (graph), and Pandas (data logging).

Working Overview:

The application starts with a GUI showing the current temperature status. Users can check temperature with a single click. If a fever is detected (temperature $\geq 99.5^{\circ}\text{F}$), a warning alert appears. The temperature log can be visualized through an interactive graph for trend analysis.

Screenshots of the Application:



Temperature: 102.0 °F


Status: Fever

Check Temperature

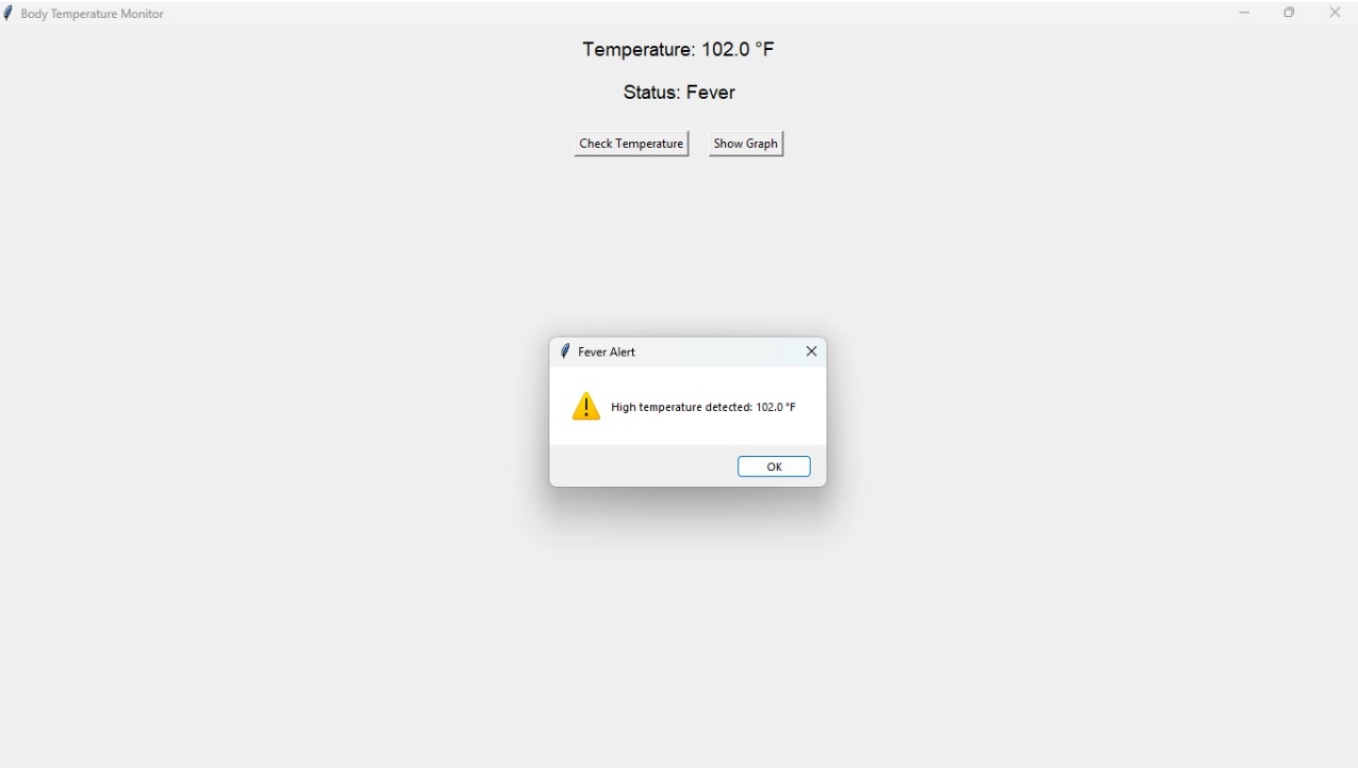
Show Graph

Fever Alert

×

 High temperature detected: 102.0 °F

OK



Source Code:

```
import tkinter as tk
from tkinter import messagebox
import pandas as pd
import random
from datetime import datetime
import matplotlib.pyplot as plt

temperature_data = []

def get_temperature():
    return round(random.uniform(96.0, 104.0), 1)

def update_temperature():
    temp = get_temperature()
    now = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
    status = "Normal" if 97.0 <= temp <= 99.5 else "Fever"

    temperature_data.append({"Time": now, "Temperature": temp, "Status": status})
    df = pd.DataFrame(temperature_data)
    df.to_csv("temperature_log.csv", index=False)

    temperature_label.config(text=f"Temperature: {temp} °F")
    status_label.config(text=f"Status: {status}")

    if status == "Fever":
        messagebox.showwarning("Fever Alert", f"High temperature detected: {temp} °F")

def show_graph():
    if not temperature_data:
        messagebox.showinfo("No Data", "No temperature data to plot.")
        return

    df = pd.DataFrame(temperature_data)
    plt.figure(figsize=(10, 5))
    plt.plot(df["Time"], df["Temperature"], marker='o', linestyle='-', color='blue')
    plt.axhline(99.5, color='red', linestyle='--', label='Fever Threshold')
    plt.title("Body Temperature Over Time")
    plt.xlabel("Time")
    plt.ylabel("Temperature (°F)")
    plt.xticks(rotation=45)
    plt.legend()
    plt.tight_layout()
    plt.show()

# GUI Setup
root = tk.Tk()
root.title("Body Temperature Monitor")
root.geometry("300x200")

temperature_label = tk.Label(root, text="Temperature: -- °F", font=('Arial', 14))
temperature_label.pack(pady=10)
```

```
status_label = tk.Label(root, text="Status: --", font=('Arial', 14))
status_label.pack(pady=5)

btn_frame = tk.Frame(root)
btn_frame.pack(pady=20)

update_btn = tk.Button(btn_frame, text="Check Temperature", command=update_temperature)
update_btn.grid(row=0, column=0, padx=10)

graph_btn = tk.Button(btn_frame, text="Show Graph", command=show_graph)
graph_btn.grid(row=0, column=1, padx=10)

root.mainloop()
```