Intel Unnati

INDUSTRIAL TRAINING REPORT

PS01: Power Manager Telemetry

RISC TAKERS

Aditya Minocha
✉ adityaminocha10@gmail.com

Devansh Raut
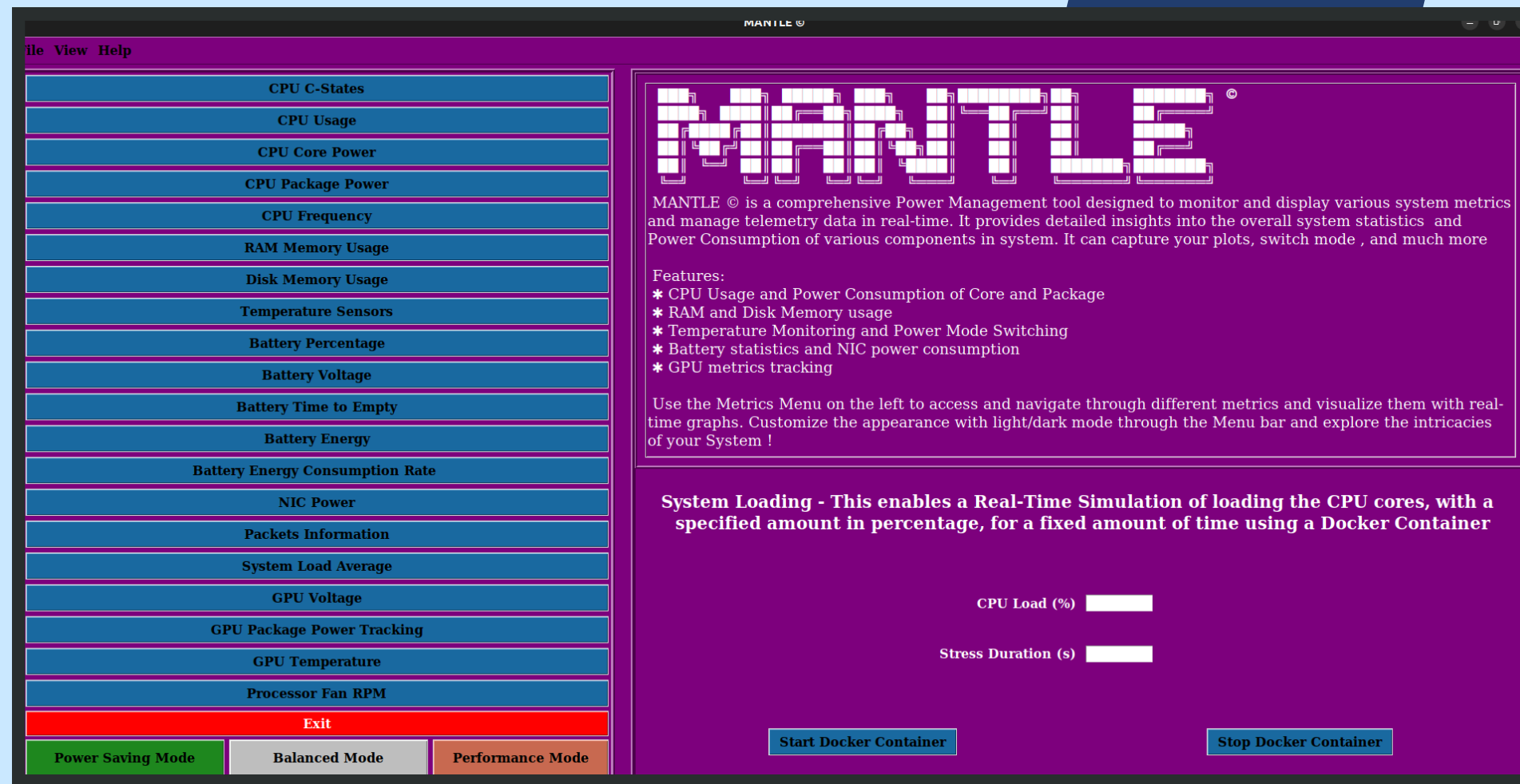✉ devansh.raut@learner.manipal.edu

Our Unique Solution

For the Power Manager Telemetry project, we developed a user-friendly GUI using tkinter and matplotlib to display real-time telemetry data for CPU, memory, NIC, and TDP. We utilized tools such as psutil for collecting system metrics, turbostat for capturing CPU states, and powerstat for measuring power consumption. To achieve specific system utilization percentages, including 100%, we employed Docker containers along with stress tools to simulate the desired load.
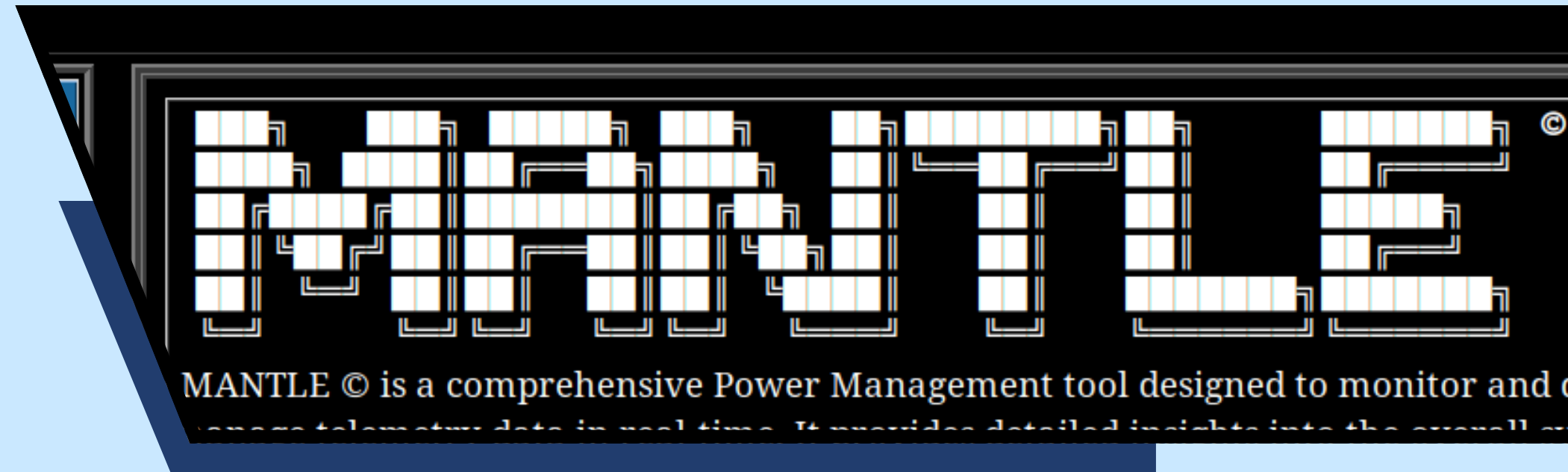
The GUI features real-time graphs, buttons for selecting various parameters, and controls to exit the program or force-stop the system load. By integrating multiprocessing, we ensured that telemetry data continues to update without interruption, even when Docker containers are running in the background. This design provides an intuitive and interactive way to monitor and manage system performance.

With this solution, we can input a desired system utilization percentage and visualize the corresponding power consumption metrics. This capability offers valuable insights for our sustainability initiative, helping us understand and optimize power usage in the era of 5G and edge computing.

Features Offered

MANTLE © is a comprehensive Power Management tool designed to monitor and o...
...anage telemetry data in real time. It provides detailed insights into the overall p...

01 **Real-time Graphs:** The GUI shows real-time graphs for different system metrics like CPU usage, memory usage, NIC power consumption, and TDP, giving you up-to-the-minute information.

02 **Parameter Selection Buttons:** You can easily switch between views of various telemetry data using straightforward buttons for CPU, memory, NIC, and TDP, making it simple to focus on what you need.

03 **Control Buttons:** There are handy buttons to exit the program and force-stop the system load, giving you immediate control over the monitoring process.

04 **Continuous Data Updates:** Thanks to multiprocessing, the GUI keeps telemetry data constantly updated, even when Docker containers are running stress tests in the background.

05 **Load Generation Integration:** The system can create specific load percentages, including 100% utilization, using Docker containers and stress tools, helping you analyze power consumption under different conditions.

06 **Interactive and Intuitive Interface:** The GUI is designed to be attractive and easy to understand, making it user-friendly for monitoring and managing system performance effectively.

Process Flow
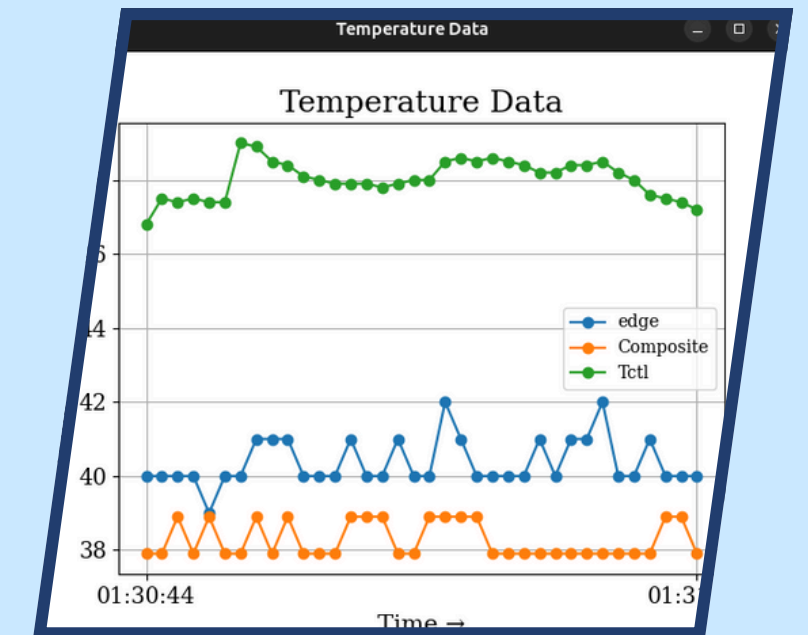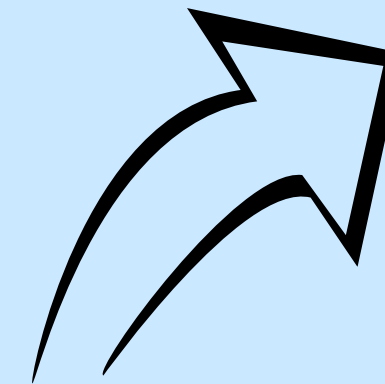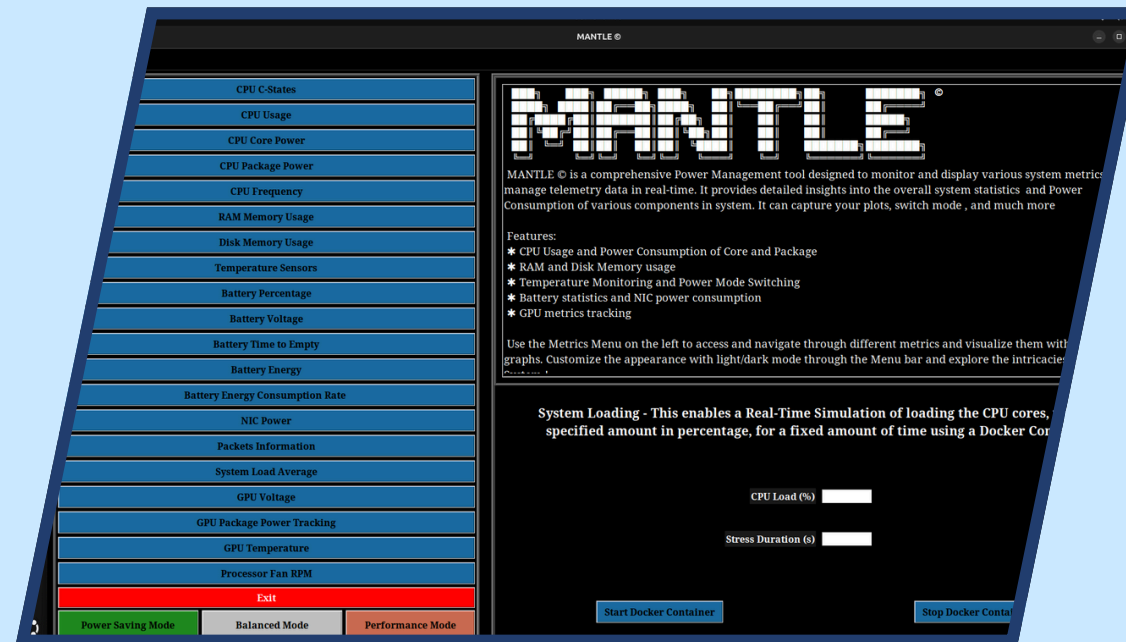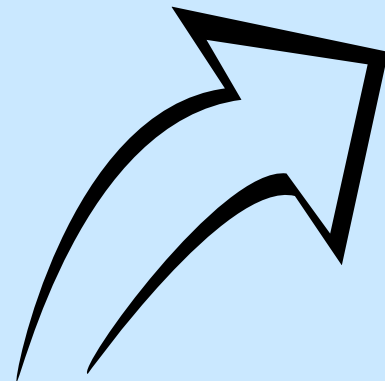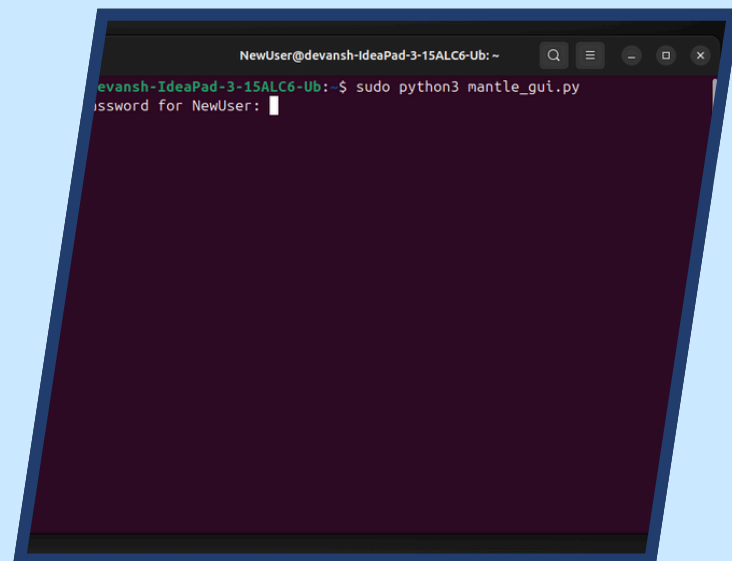
The process starts when you launch the GUI application. Using tkinter, the main window comes to life, featuring a sidebar with buttons for selecting different parameters (CPU, Memory, NIC, TDP) and control buttons to exit the program or force-stop the system load. On the right side, there's a dedicated area for displaying real-time graphs.

Once the GUI initializes, it kicks off background processes to continuously gather telemetry data. This data is collected using tools like psutil for system metrics, turbostat for detailed CPU states, and powerstat for power consumption measurements. These processes run in parallel due to multiprocessing, ensuring that the data stays updated without disrupting the user interface.

As you interact with the GUI, you can choose different parameters to monitor by clicking the buttons. For example, selecting the CPU button will show real-time graphs of CPU usage. The same goes for Memory, NIC, and TDP buttons, each showing their respective metrics. The control buttons let you exit the application or force-stop any running system load tests. This setup provides a clear and continuous view of your system's performance and power consumption, making it easy to monitor and manage effectively.

Architecture Diagram
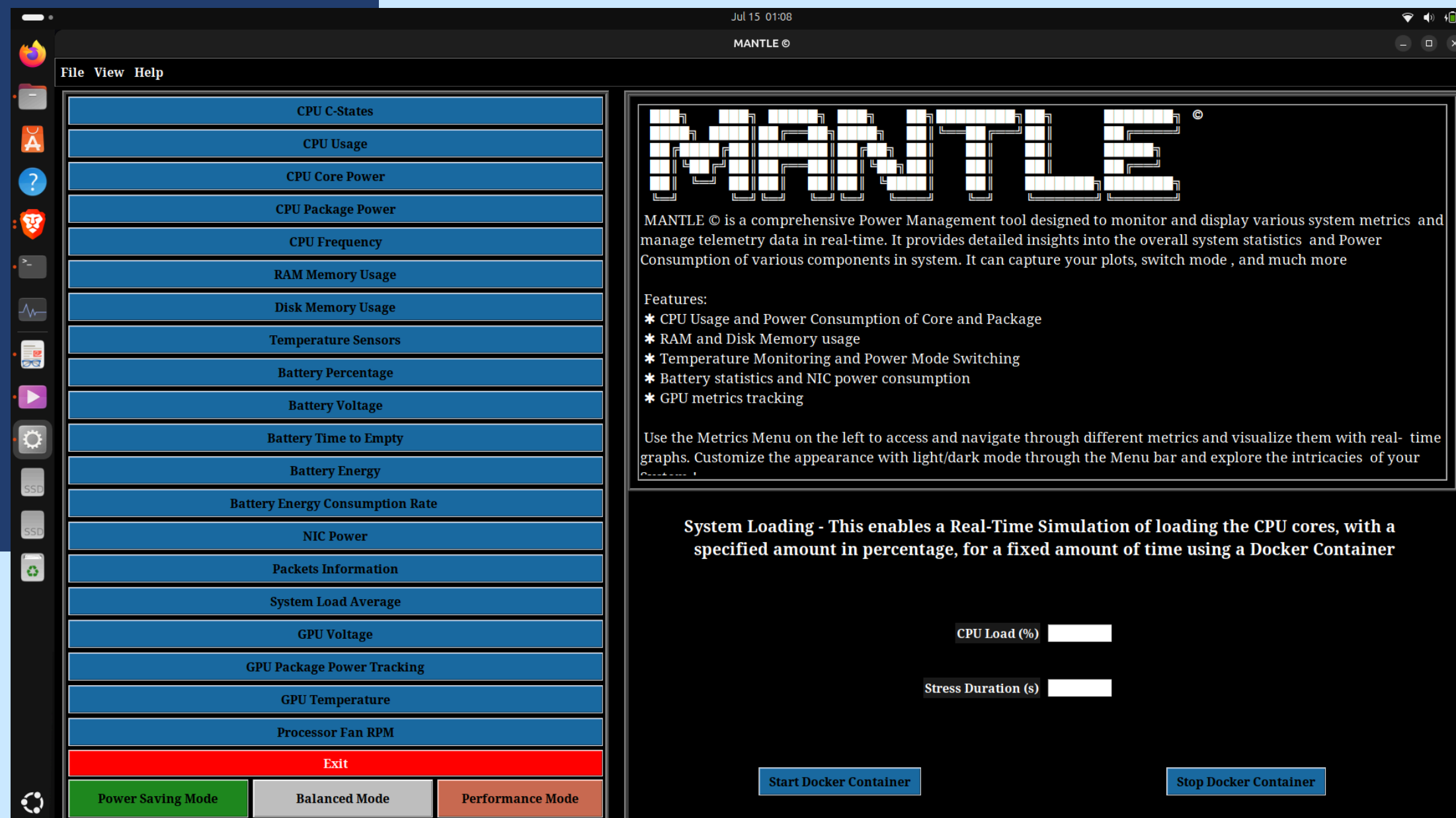
This is our innovative Power Telemetry GUI, named Mantle. Meticulously crafted to showcase real-time CPU, memory, NIC, and TDP metrics with interactive graphs and seamless parameter selection.

Technologies Used

In our Power Telemetry GUI, we use a variety of technologies to make sure everything runs smoothly and is easy to use:

01 System Utilities: Tools like turbostat and powerstat are also part of our toolkit, providing insights into CPU states and power usage metrics, which are crucial for optimizing energy efficiency.

02 Docker: We use Docker to simulate different loads on your system, helping us measure power consumption accurately under various conditions.

03 tkinter: This lets us create the graphical interface itself, providing all the buttons and windows you interact with.

04 matplotlib: We use this for showing real-time graphs of CPU usage, memory stats, NIC power, and TDP data. It's great for visualizing how your system is performing.

05 psutil: This helps us gather detailed system metrics like CPU usage and memory stats, giving you accurate data to monitor your system.

06 multiprocessing: To keep everything responsive while collecting data in the background, we use multiprocessing. This means you can view real-time updates without any lag.

Team Members & Contribution

Devansh Raut

Worked on creating the stress test using Docker and integration of Docker in Mantle along with its beautification. Created the Github repository for submission

📞 +91 7021256312

✉️ devansh.raut@learner.manipal.edu

Aditya Minocha

Worked on creating Mantle's graphical user interface using python and added various parameters to monitor the telemetry data in an error and bug free manner. Created the PPT for submission

📞 +91 9060011365

✉️ adityaminocha10@gmail.com

Special Thanks to our institution mentor for their support and guidance during this project

Vinay Kumar Jadoun

📞 +91 7891377638

✉️ vinay.jadoun@learner.manipal.edu

Conclusion

In wrapping up our Power Telemetry project,Through our user-friendly GUI, powered by softwares like tkinter, matplotlib, psutil, Docker, and multiprocessing, we've enabled users to easily monitor real-time metrics such as CPU usage, memory stats, NIC power consumption, and TDP data. This helps us better understand system performance and provides crucial insights for reducing overall power usage.

Our Power Telemetry project is highly relevant in today's world where energy use is increasing. With technologies like 5G and edge computing becoming more prevalent, there's a greater need to manage power wisely.Insight into system performance and power usage not only supports better resource management but also helps in making our technology more sustainable. It's crucial for addressing the challenges of energy efficiency and environmental impact in today's fast-paced tech-driven world.

```python
1  import psutil
2  import matplotlib.py
3  from matplotlib.backends
4  from datetime import dateti
5  import tkinter as tk
6  import subprocess
7  import re
8  import time
9  import threading
10 from multiprocessing import Process
11 from tkinter import ttk
12 from tkinter import messagebox
13 from tkinter.font import Font
14 import os
15 from tkinter import PhotoImage
16 from PIL import Image, ImageTk
17
18 plt.rcParams['font.family'] = 'serif'
19
20 # Track Docker process
21 docker_process = None
22
23 global x
24
25 # Initialize data storage for plotting
26 data_series = {
27     'cpu_cstates': {'time': [], 'data'
28     'cpu_usage': {'time': [], 'data':
29     'cpu_core_power': {'time': [], 'da      ng
30     'cpu_pkg_power': {'time': [], 'dat
31     'cpu_frequency': {'time': [], 'dat a': {'C1':[],'C2':[],'C3':[]}},
32     'memory_usage': {'time': [], 'data : [], 'per_core': []},
'cached': []}},                       data': []},
33     'disk_usage': {'time': [], 'data'  ata': []},
34     'temperature': {'time': [], 'data'  ata': {'current': []}},
35     'battery_percent': {'time': [], 'd a': {'total': [], 'available': [], 'free':
```