



Bahria University, Islamabad
Department of Software Engineering

Operating Systems Lab
(Fall-2025)

Teacher: Engr. Saad Mazhar Khan

Student: Hasan Zahid

Enrollment: 01-131232-028

Lab Journal: Project Report
Date: December 17, 2025



Repository Link:
<https://github.com/hasanzafzal/OS-Semester-Project>

Comments:

Signature

SYSTEM HEALTH MONITORING AND VISUALIZATION SYSTEM

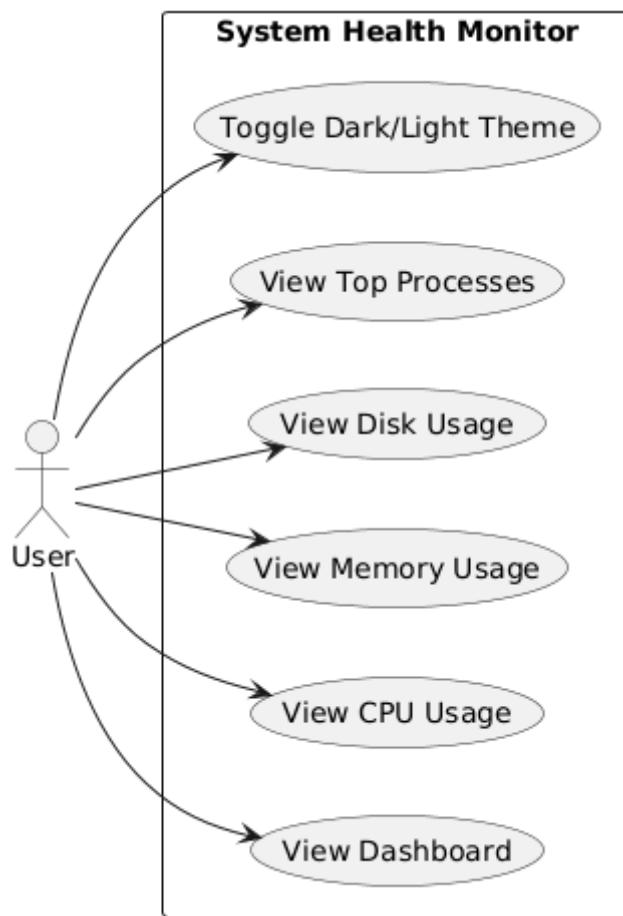
1. Introduction

- Problem Statement:
Modern operating systems run multiple processes simultaneously, consuming CPU, memory, disk, and other system resources. Users often lack a simple and unified tool to monitor system health in real time.
- Objectives of the Project:
 - Monitor CPU, memory, disk usage and uptime
 - Implement backend monitoring using Bash scripting
 - Develop a GUI frontend using PyQt6
 - Display top CPU-consuming processes and real-time graphs
 - Demonstrate core Operating System concepts
- Scope and Limitations:
Scope includes Fedora Linux systems with real-time monitoring and visualization. Limitations include lack of remote monitoring, process control, and persistent database storage.
- Significance of the Project:
The project bridges CLI and GUI monitoring, helps understand OS resource management, and provides practical Linux administration skills.

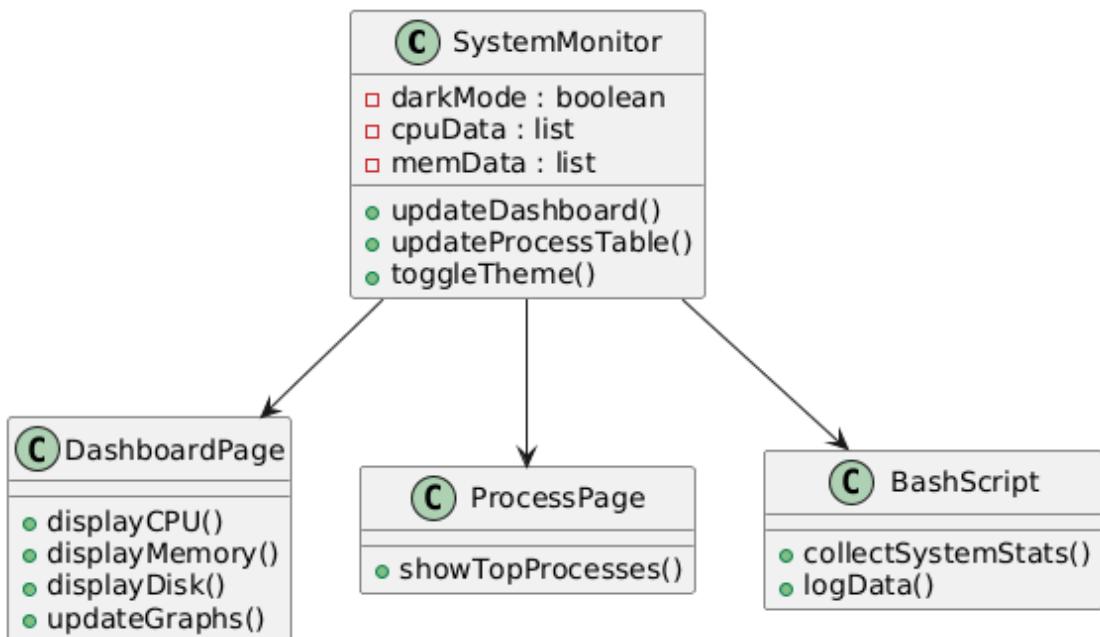
2. System Analysis

- Requirement Analysis:
 - Functional requirements include real-time monitoring, process listing, and theme toggling.
 - Non-functional requirements include stability in Fedora 43 and responsiveness.
- Feasibility Study:
 - Technical feasibility is ensured by using open-source tools.
 - Operational feasibility is high due to user-friendly interface.
 - Financial feasibility is ensured with zero cost.

➤ Use Case Diagram:



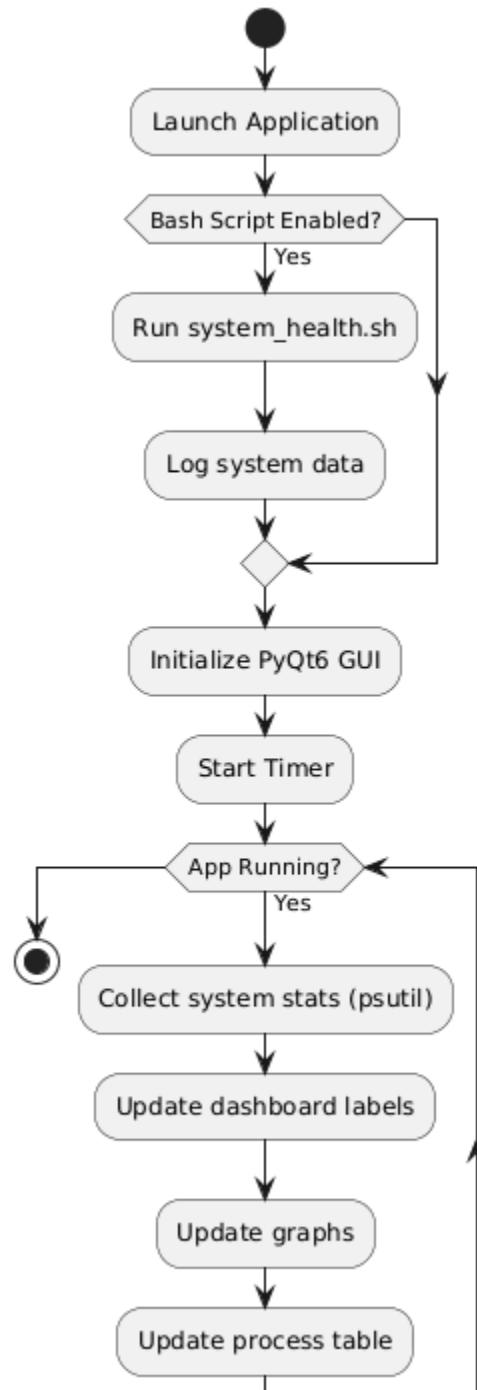
➤ Class Diagram:



3. System Design

➤ Architectural Design:

- The system follows a hybrid Bash backend and PyQt6 frontend architecture.
- Flow-Chart:



➤ Database Design:

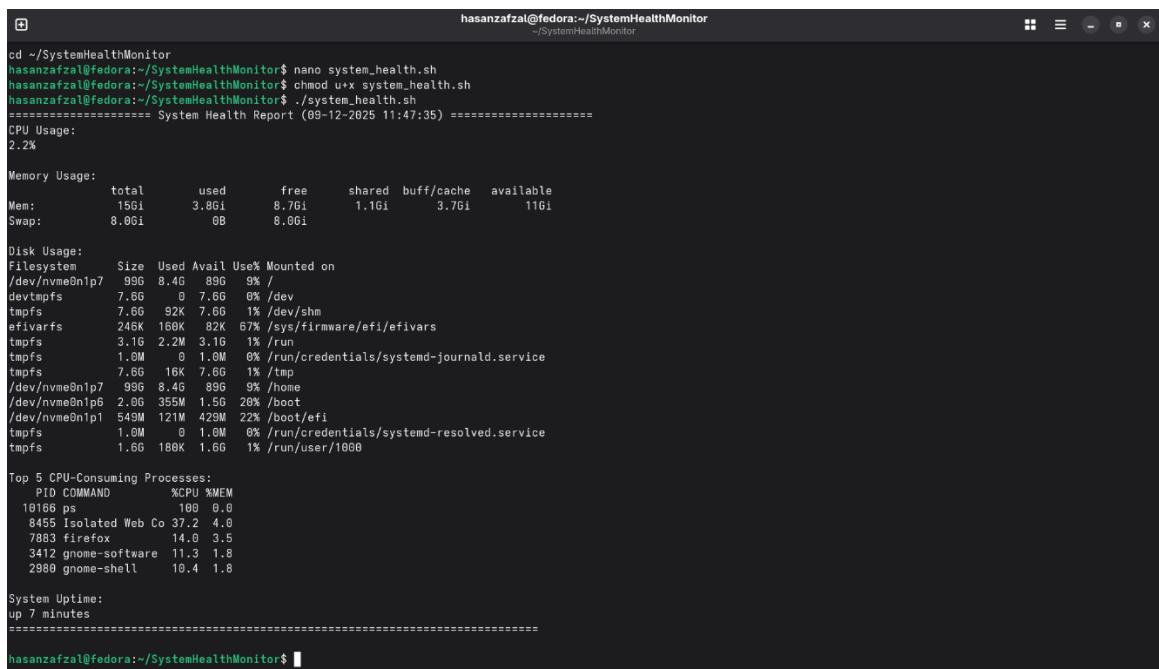
No database is used; logging is done via text files.

➤ User Interface Design:

The UI includes a dashboard, graphs, process tables, and dark/light mode.

4. Implementation

- Programming Languages and Tools Used:
 - Bash
 - Python (PyQt6)
 - psutil
 - pyqtgraph
- Modules Description:
Includes monitoring module, GUI dashboard, process viewer, and graph module.
- Code Snippets:
Complete code is available on GitHub. (Link attached above)
- Screenshots:



```
hasanzafzal@fedora:~/SystemHealthMonitor
hasanzafzal@fedora:~/SystemHealthMonitor$ nano system_health.sh
hasanzafzal@fedora:~/SystemHealthMonitor$ chmod u+x system_health.sh
hasanzafzal@fedora:~/SystemHealthMonitor$ ./system_health.sh
=====
System Health Report (09-12-2025 11:47:35)
=====

CPU Usage:
2.2K

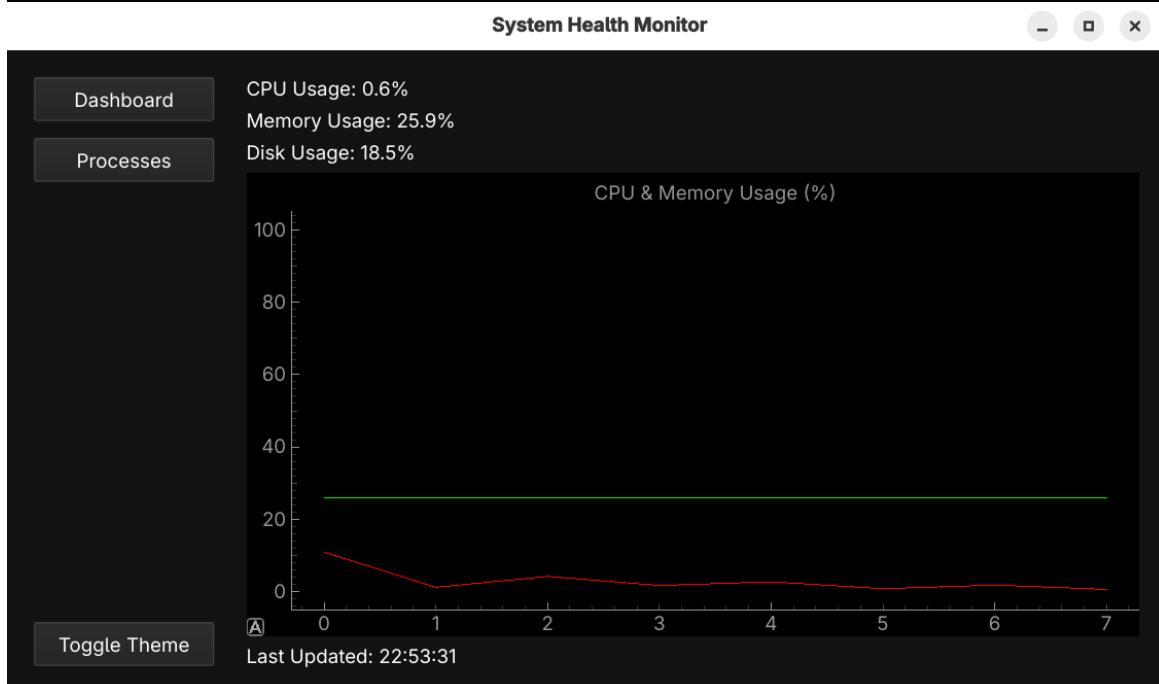
Memory Usage:
total        used        free      shared  buff/cache   available
Mem:       1561       3.8G       8.7G       1.1G       3.7G       11G
Swap:      8.0G       0B       8.0G

Disk Usage:
Filesystem  Size  Used Avail Use% Mounted on
/dev/nvme0n1p7  99G  8.4G  89G  9% /
devtmpfs    7.6G  0    7.6G  0% /dev
tmpfs       7.6G  92K  7.6G  1% /dev/shm
efivarsfs   246K  168K  82K  67% /sys/firmware/efi/efivars
tmpfs       3.1G  2.2M  3.1G  1% /run
tmpfs       1.8M  0    1.8M  0% /run/credentials/systemd-journald.service
tmpfs       7.6G  16K  7.6G  1% /tmp
tmpfs       95G  8.4G  89G  9% /home
/dev/nvme0n1p6  2.0G  355M  1.5G  20% /boot
/dev/nvme0n1p1  549M  121M  429M  22% /boot/efi
tmpfs       1.0M  0    1.0M  0% /run/credentials/systemd-resolved.service
tmpfs       1.6G  188M  1.6G  1% /run/user/1000

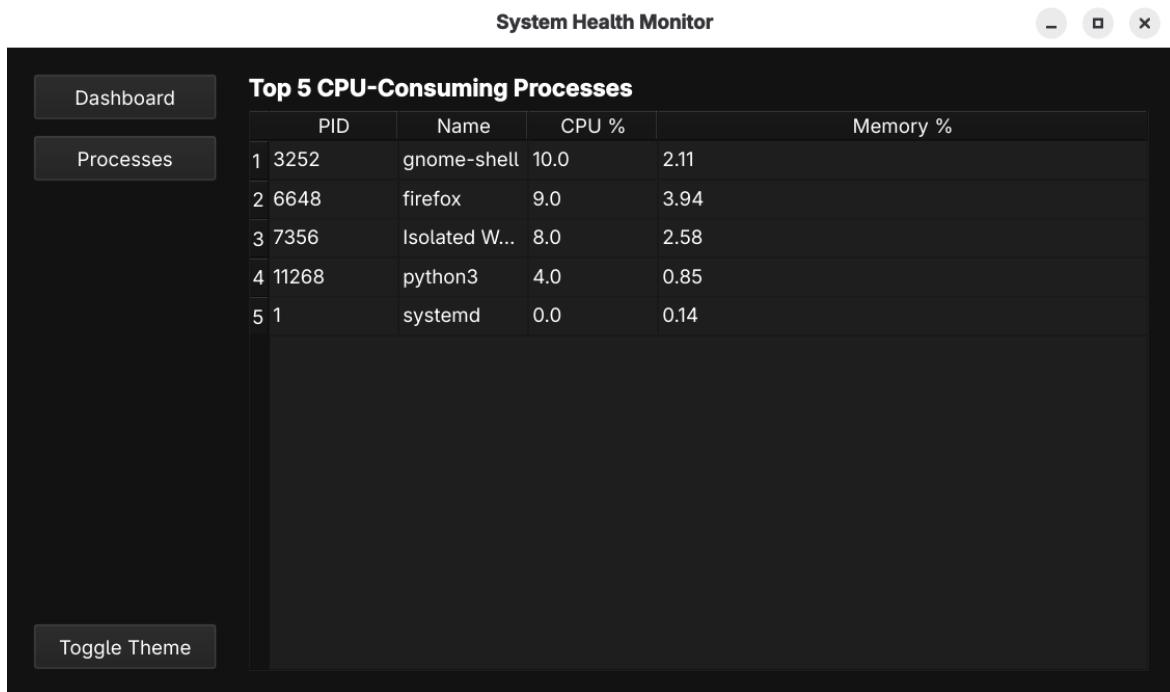
Top 5 CPU-Consuming Processes:
 PID COMMAND      %CPU %MEM
 10166 ps          100  0.0
 8455 Isolated Web Co 37.2  4.0
 7883 firefox      14.0  3.5
 3412 gnome-software 11.3  1.8
 2980 gnome-shell   10.4  1.8

System Uptime:
up 7 minutes
=====

hasanzafzal@fedora:~/SystemHealthMonitor$
```



The screenshot shows a dark-themed application window titled "System Health Monitor". At the top, there are three tabs: "Dashboard" (which is selected), "Processes", and "Toggle Theme". Below the tabs, there are three status indicators: "CPU Usage: 0.6%", "Memory Usage: 25.9%", and "Disk Usage: 18.5%". A large chart titled "CPU & Memory Usage (%)" displays two lines over time from 0 to 7 hours. A green line represents memory usage, which starts at approximately 25% and remains relatively stable. A red line represents CPU usage, which starts at approximately 10%, dips to near zero around hour 1, and then fluctuates between 1% and 5% for the remainder of the period. At the bottom of the chart, it says "Last Updated: 22:53:31".



5. Results and Discussion

- Outcomes:
A stable, real-time system monitoring application was developed.
- Challenges Faced:
 - Creating GUI with bash program
 - GUI incompatibility issues on Fedora 43 were resolved using PyQt6.
- Comparison with Existing Solutions:
The project provides GUI and graphs unlike traditional CLI tools.

6. Conclusion and Future Work

- Summary of Achievements:
Successfully implemented a hybrid system monitoring solution.
- Future Enhancements:
 - Alerts
 - Historical data storage
 - Network monitoring
 - RPM packaging

7. References

- Fedora Project:
Fedora Linux Documentation.

Available at: <https://docs.fedoraproject.org>

(Accessed for Fedora 43 system behavior, package management, and Wayland environment)

- GNU Bash Manual:

Bash Reference Manual.

Available at: <https://www.gnu.org/software/bash/manual/>

(Referenced for shell syntax, execution permissions, and scripting practices)

- Python Software Foundation:

Python 3 Documentation.

Available at: <https://docs.python.org/3/>

(Used for Python language constructs and subprocess integration)

- psutil Development Team:

psutil – Cross-platform system and process utilities.

Available at: <https://psutil.readthedocs.io>

(Used for retrieving CPU, memory, disk usage, uptime, and process statistics)

- PyQt Development Team (Riverbank Computing):

PyQt6 Documentation.

Available at: <https://www.riverbankcomputing.com/static/Docs/PyQt6/>

(Used for GUI design, widgets, layouts, timers, and styling)

- pyqtgraph Developers:

pyqtgraph Documentation.

Available at: <https://pyqtgraph.readthedocs.io>

(Used for real-time CPU and memory graph visualization)