Capstone Project Report

Topic - System Monitor Tool (LSP)

Objective

Create a system monitor tool in C++ that displays real-time information about system processes, memory usage, and CPU load, similar to the 'top' command.

Tools and Technologies

- **Language:** C++ (C++17)
- **Compiler:** `g++`
- **Operating System:** Ubuntu (Linux)
- **Running on : Terminal(ubuntu 24.04)**

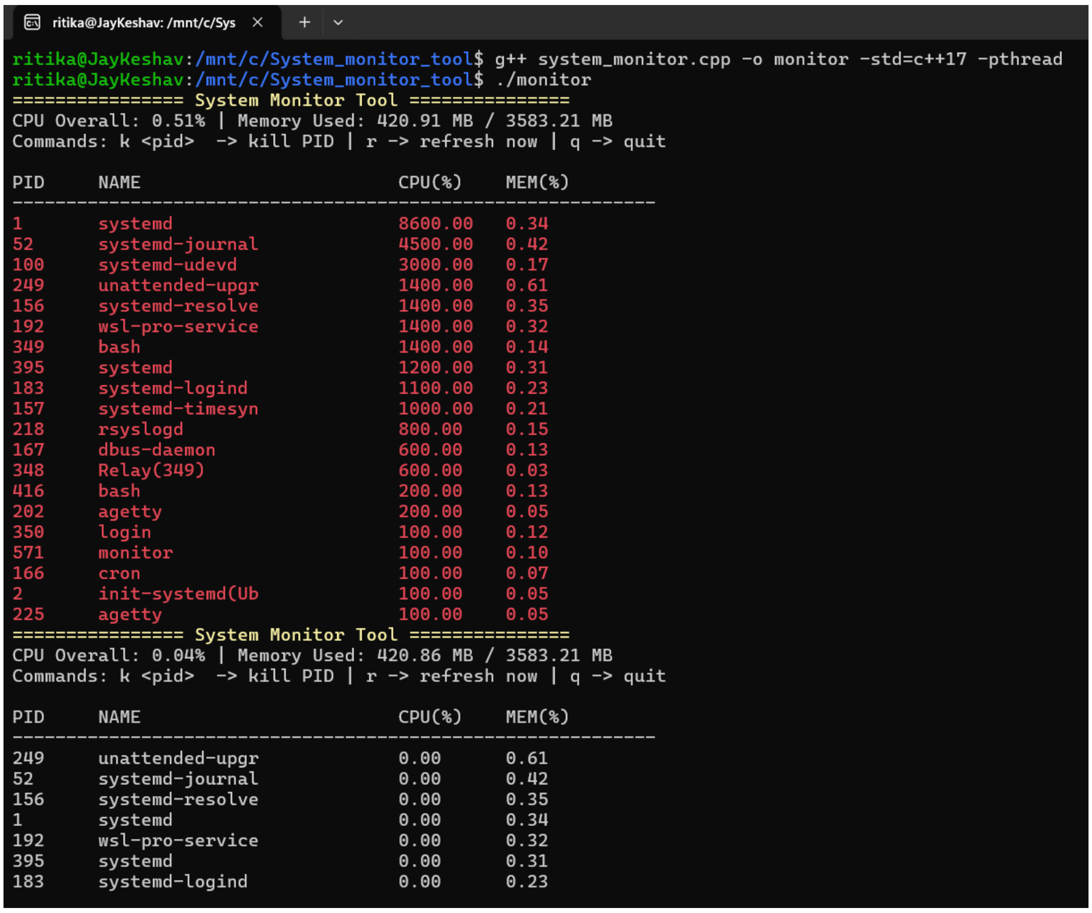
Code

```
#include <bits/stdc++.h>

#include <sys/types.h>
#include <signal.h>
#include <unistd.h>
#include <fcntl.h>
#include <poll.h>

using namespace std;

struct ProcSnapshot {
    unsigned long long totalTime; // utime + stime in clock ticks
    unsigned long rss;
};

struct ProcInfo {
    int pid;
    string name;
    double cpuPercent;
    double memPercent;
};
```

```
static long pageSize = sysconf(_SC_PAGESIZE);
static long ticksPerSec = sysconf(_SC_CLK_TCK);

unsigned long long readTotalJiffies() {
    ifstream f("/proc/stat");
    string line;
    if (!getline(f, line)) return 0;
    string cpu;
    unsigned long long user=0, nice=0, system=0, idle=0, iowait=0, irq=0, softirq=0, steal=0;
    stringstream ss(line);
    ss >> cpu >> user >> nice >> system >> idle >> iowait >> irq >> softirq >> steal;
    return user + nice + system + idle + iowait + irq + softirq + steal;
}

pair<unsigned long long, unsigned long long> getMemoryKB() {
    ifstream f("/proc/meminfo");
    string key;
    unsigned long long value;
    string unit;
    unsigned long long total=0, free=0, avail=0;
    while (f >> key >> value >> unit) {
        if (key=="MemTotal:") total = value;
        else if (key=="MemAvailable:") { avail = value; break; }
    }
    return {total, avail}; // in KB
}

bool isNumber(const string &s) {
    for(char c: s) if(!isdigit(c)) return false;
    return true;
}

ProcSnapshot readProcSnapshot(int pid) {
    ProcSnapshot snap{0,0};
    string base = "/proc/" + to_string(pid) + "/";
    // read stat for utime and stime (14th and 15th fields are utime and stime)
    ifstream fstat(base + "stat");
    if (!fstat.is_open()) return snap;
    string token;
    string line;
    getline(fstat, line);
    string comm;

    auto p1 = line.find('(');
    auto p2 = line.rfind(')');
    if (p1==string::npos || p2==string::npos || p2<=p1) return snap;
    comm = line.substr(p1+1, p2-p1-1);
    string rest = line.substr(p2+2);
    stringstream ss(rest);

    unsigned long long utime=0, stime=0;
    for (int i=0;i<11;i++) {
        if (!(ss >> token)) return snap;
    }
    // next should be utime (13th) and stime (14th)
    if (!(ss >> utime >> stime)) return snap;
    snap.totalTime = utime + stime;

    ifstream fstatm(base + "statm");
    if (fstatm.is_open()) {
        unsigned long size=0, rss=0;
        fstatm >> size >> rss;
        snap.rss = rss;
    } else {
        // fallback to status
        ifstream fstatus(base + "status");
        string line2;
        while (getline(fstatus, line2)) {
            if (line2.rfind("VmRSS:",0)==0) {
                stringstream s2(line2);
                string key; unsigned long val; string unit2;
                s2 >> key >> val >> unit2;
                // val is in kB; convert to pages
                snap.rss = (val*1024 + pageSize -1)/pageSize;
                break;
            }
        }
    }
    return snap;
}

string readProcName(int pid) {
    string name;
    string commPath = "/proc/" + to_string(pid) + "/comm";
    ifstream f(commPath);
    if (f.is_open()) {
        getline(f, name);
        return name;
    }
    // fallback to stat
    ifstream fstat("/proc/" + to_string(pid) + "/stat");
    if (fstat.is_open()) {
        string line;
        getline(fstat, line);
        auto p1 = line.find('(');
        auto p2 = line.rfind(')');
        if (p1!=string::npos && p2!=string::npos && p2>p1) {
            return line.substr(p1+1, p2-p1-1);
        }
    }
    return "";
}

vector<int> listPids() {
    vector<int> pids;
    for (const auto &entry : filesystem::directory_iterator("/proc")) {
        if (entry.is_directory()) {
            string name = entry.path().filename();
            if (isNumber(name)) pids.push_back(stoi(name));
        }
    }
    return pids;
}

void clearScreen() {
    cout << "\033[2J\033[H";
}

void setNonBlockingStdin(bool enable) {
    int flags = fcntl(STDIN_FILENO, F_GETFL, 0);
    if (enable) fcntl(STDIN_FILENO, F_SETFL, flags | O_NONBLOCK);
    else fcntl(STDIN_FILENO, F_SETFL, flags & ~O_NONBLOCK);
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);

    // maps to hold previous snapshots
    unordered_map<int, ProcSnapshot> prevProc;
    unsigned long long prevTotal = readTotalJiffies();
    auto [totalKB, availKB] = getMemoryKB();
    (void)totalKB;

    const int REFRESH_MS = 2000;
    bool running = true;

    // Prepare pollfd for stdin
    struct pollfd pfd;
    pfd.fd = STDIN_FILENO;
```

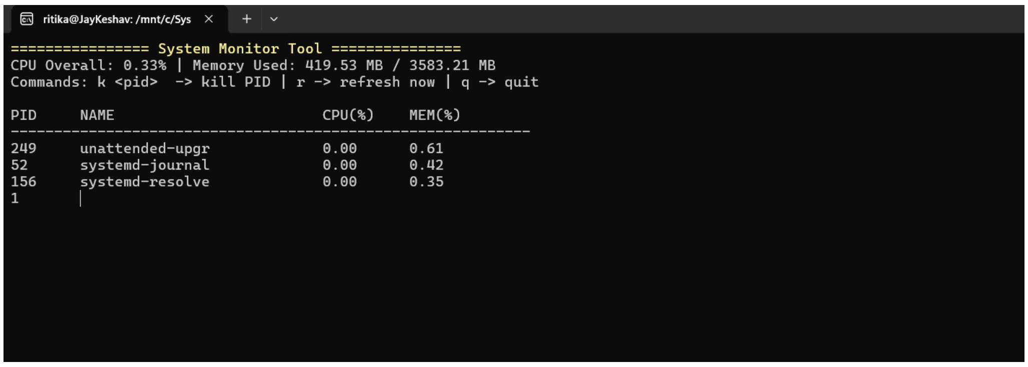
```cpp
    pfd.events = POLLIN;

    while (running) {
        // take current snapshots
        unsigned long long totalJ = readTotalJiffies();
        vector<int> pids = listPids();
        unordered_map<int, ProcSnapshot> curProc;
        vector<ProcInfo> procs;
        for (int pid : pids) {
            ProcSnapshot s = readProcSnapshot(pid);
            curProc[pid] = s;
        }
        // compute deltas
        unsigned long long totalDelta = totalJ - prevTotal;
        if (totalDelta == 0) totalDelta = 1; // avoid div by zero

        // read mem info
        auto [totKB, availKB2] = getMemoryKB();
        unsigned long long usedKB = (totKB > availKB2) ? (totKB - availKB2) : 0;
        double memUsedBytes = usedKB * 1024.0;
        double totalBytes = totKB * 1024.0;

        for (auto &kv : curProc) {
            int pid = kv.first;
            ProcSnapshot cur = kv.second;
            ProcSnapshot prev = {0,0};
            if (prevProc.find(pid) != prevProc.end()) prev = prevProc[pid];
            unsigned long long procDelta = 0;
            if (cur.totalTime >= prev.totalTime) procDelta = cur.totalTime - prev.totalTime;
            double cpuPerc = (100.0 * (double)procDelta / (double)totalDelta);
            double memPerc = 0.0;
            if (cur.rss > 0 && totalBytes>0) {
                double rssBytes = (double)cur.rss * (double)pageSize;
                memPerc = (rssBytes / totalBytes) * 100.0;
            }
            string name = readProcName(pid);
            if (name.empty()) continue;
            procs.push_back({pid, name, cpuPerc, memPerc});
        }

        // sort by cpu desc
        sort(procs.begin(), procs.end(), [](const ProcInfo &a, const ProcInfo &b){
            if (a.cpuPercent == b.cpuPercent) return a.memPercent > b.memPercent;
            return a.cpuPercent > b.cpuPercent;
        });

        // display
    clearScreen();
    cout << "\033[1;33m================ System Monitor Tool ===============\033[0m\n";
    double cpuUsage = 0.0;
    {
        // estimate overall cpu usage as 100 * (totalDelta - idleDelta)/totalDelta isn't available here easily,
        ifstream f("/proc/stat");
        string line;
        if (getline(f,line)) {
            string cpu;
            unsigned long long user=0,nice=0,system=0,idle=0,iowait=0,irq=0,softirq=0,steal=0;
            stringstream ss(line);
            ss >> cpu >> user >> nice >> system >> idle >> iowait >> irq >> softirq >> steal;
            unsigned long long idleAll = idle + iowait;
            unsigned long long nonIdle = user + nice + system + irq + softirq + steal;
            static unsigned long long prevTotal2 = 0, prevIdle2 = 0;
            unsigned long long total2 = idleAll + nonIdle;
            unsigned long long totalDiff2 = total2 - prevTotal2;
            unsigned long long idleDiff2 = idleAll - prevIdle2;
            if (totalDiff2 == 0) cpuUsage = 0.0;
            else cpuUsage = 100.0 * (double)(totalDiff2 - idleDiff2) / (double)totalDiff2;
            prevTotal2 = total2; prevIdle2 = idleAll;
        }
    }
    cout << fixed << setprecision(2);
    cout << "CPU Overall: " << cpuUsage << "% | ";
    cout << "Memory Used: " << (usedKB/1024.0) << " MB / " << (totKB/1024.0) << " MB\n";
    cout << "Commands: k <pid>  -> kill PID | r -> refresh now | q -> quit\n\n";

    cout << left << setw(8) << "PID" << setw(28) << "NAME" << setw(10) << "CPU(%)" << setw(10)
<< "MEM(%)" << "\n";
    cout << string(60,'-') << "\n";
    int show = min((int)procs.size(), 20);
    for (int i=0;i<show;i++) {
        auto &p = procs[i];
        // color high CPU
        if (p.cpuPercent > 50.0) cout << "\033[1;31m";
        else if (p.cpuPercent > 10.0) cout << "\033[1;33m";
        else cout << "\033[0m";
        cout << left << setw(8) << p.pid << setw(28) << p.name
             << setw(10) << p.cpuPercent << setw(10) << p.memPercent << "\033[0m\n";
    }
    cout << "\n";

    // prepare for input polling for REFRESH_MS milliseconds
    int timeout = REFRESH_MS;
    int ret = poll(&pfd, 1, timeout);
    if (ret > 0 && (pfd.revents & POLLIN)) {
            string line;

            if (!getline(cin, line)) {

                char buf[1024];
                ssize_t r = read(STDIN_FILENO, buf, sizeof(buf));
                (void)r;
                continue;
            }

            while (!line.empty() && isspace(line.back())) line.pop_back();
            while (!line.empty() && isspace(line.front())) line.erase(line.begin());
            if (line.empty()) {
                // continue
            } else if (line == "q" || line=="Q") {
                running = false;
                break;
            } else if (line[0]=='k' || line[0]=='K') {

                stringstream ss(line);
                string cmd; int pid;
                ss >> cmd >> pid;
                if (pid>0) {
                    if (kill(pid, SIGTERM) == 0) {
                        cout << "Sent SIGTERM to " << pid << "\n";
                    } else {
                        perror("kill");
                    }

                    this_thread::sleep_for(chrono::milliseconds(300));
                } else {
                    cout << "Invalid PID\n";
                }
            } else if (line=="r" || line=="R") {
                // force refresh immediately (loop continues)
            } else {
                cout << "Unknown command\n";
            }
        } else {
            // timeout, no user input; continue refresh
        }

        // swap prevProc
        prevProc.swap(curProc);
        prevTotal = totalJ;
    }
    cout << "Exiting System Monitor.\n";
    return 0;
}
```

Screenshot



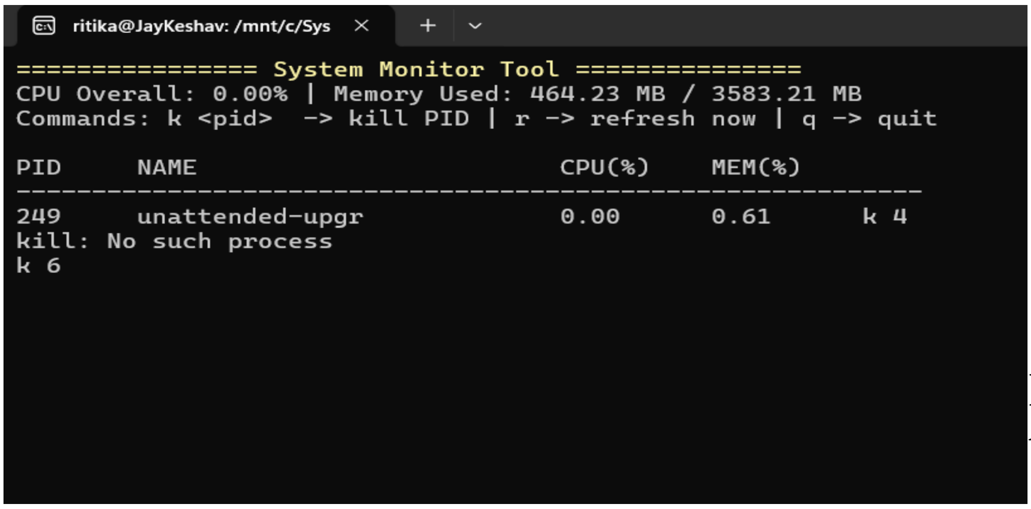
```
ritika@JayKeshav: /mnt/c/Sys
ritika@JayKeshav:/mnt/c/System_monitor_tool$ g++ system_monitor.cpp -o monitor -std=c++17 -pthread
ritika@JayKeshav:/mnt/c/System_monitor_tool$ ./monitor
================ System Monitor Tool ===============
CPU Overall: 0.51% | Memory Used: 420.91 MB / 3583.21 MB
Commands: k <pid>  -> kill PID | r -> refresh now | q -> quit

PID     NAME                            CPU(%)    MEM(%)
------------------------------------------------------------
1       systemd                         8600.00   0.34
52      systemd-journal                 4500.00   0.42
100     systemd-udevd                   3000.00   0.17
249     unattended-upgr                 1400.00   0.61
156     systemd-resolve                 1400.00   0.35
192     wsl-pro-service                 1400.00   0.32
349     bash                            1400.00   0.14
395     systemd                         1200.00   0.31
183     systemd-logind                  1100.00   0.23
157     systemd-timesyn                 1000.00   0.21
218     rsyslogd                        800.00    0.15
167     dbus-daemon                     600.00    0.13
348     Relay(349)                      600.00    0.03
416     bash                            200.00    0.13
202     agetty                          200.00    0.05
350     login                           100.00    0.12
571     monitor                         100.00    0.10
166     cron                            100.00    0.07
2       init-systemd(Ub                 100.00    0.05
225     agetty                          100.00    0.05
================ System Monitor Tool ===============
CPU Overall: 0.04% | Memory Used: 420.86 MB / 3583.21 MB
Commands: k <pid>  -> kill PID | r -> refresh now | q -> quit

PID     NAME                            CPU(%)    MEM(%)
------------------------------------------------------------
249     unattended-upgr                 0.00      0.61
52      systemd-journal                 0.00      0.42
156     systemd-resolve                 0.00      0.35
1       systemd                         0.00      0.34
192     wsl-pro-service                 0.00      0.32
395     systemd                         0.00      0.31
183     systemd-logind                  0.00      0.23
```

Compilation in ubuntu terminal (24.04)



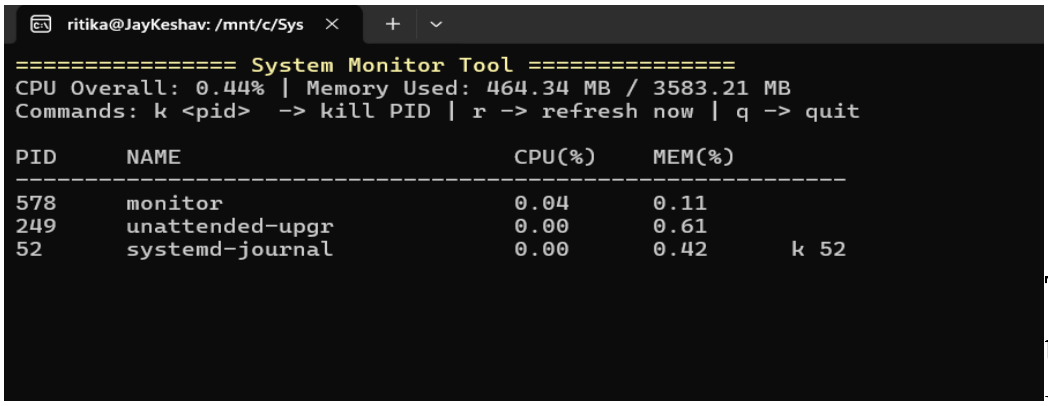
```
ritika@JayKeshav: /mnt/c/Sys
=============== System Monitor Tool ==============
CPU Overall: 0.33% | Memory Used: 419.53 MB / 3583.21 MB
Commands: k <pid>  -> kill PID | r -> refresh now | q -> quit

PID     NAME                            CPU(%)    MEM(%)
------------------------------------------------------------
249     unattended-upgr                 0.00      0.61
52      systemd-journal                 0.00      0.42
156     systemd-resolve                 0.00      0.35
1       |
```

Running the program , getting output as the details of cpu, memory, processes



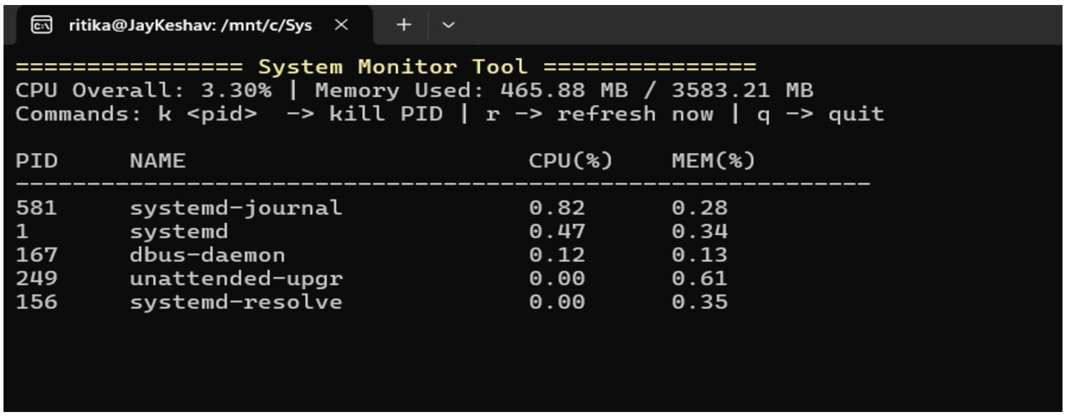
```
ritika@JayKeshav: /mnt/c/Sys

================ System Monitor Tool ===============
CPU Overall: 0.00% | Memory Used: 464.23 MB / 3583.21 MB
Commands: k <pid>  -> kill PID | r -> refresh now | q -> quit

PID     NAME                           CPU(%)    MEM(%)
------------------------------------------------------------
249     unattended-upgr                0.00      0.61      k 4
kill: No such process
k 6
```

Killing the process using pid , getting output , no such process because there is no pid 4



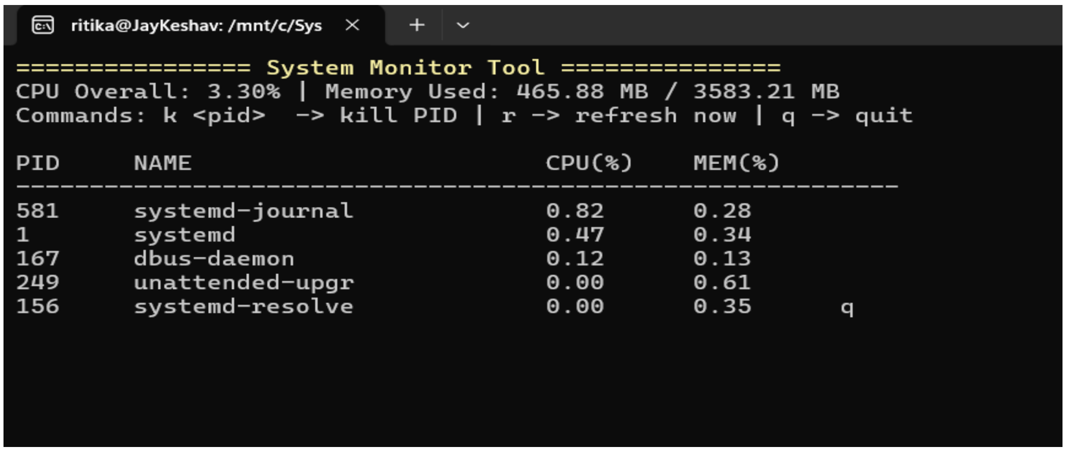
```
ritika@JayKeshav: /mnt/c/Sys

================ System Monitor Tool ===============
CPU Overall: 0.44% | Memory Used: 464.34 MB / 3583.21 MB
Commands: k <pid>  -> kill PID | r -> refresh now | q -> quit

PID     NAME                           CPU(%)    MEM(%)
------------------------------------------------------------
578     monitor                        0.04      0.11
249     unattended-upgr                0.00      0.61
52      systemd-journal                0.00      0.42      k 52
```

Trying to kill pid 52, given command k 52



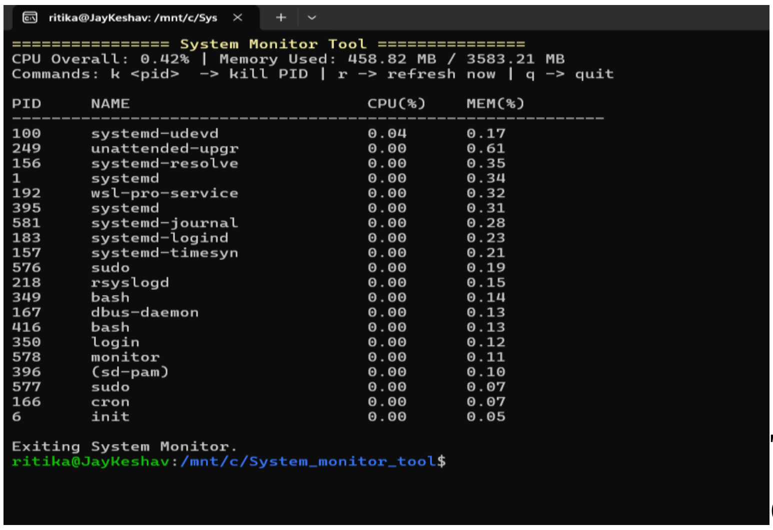
```
ritika@JayKeshav: /mnt/c/Sys

================ System Monitor Tool ===============
CPU Overall: 3.30% | Memory Used: 465.88 MB / 3583.21 MB
Commands: k <pid>  -> kill PID | r -> refresh now | q -> quit

PID     NAME                           CPU(%)    MEM(%)
------------------------------------------------------------
581     systemd-journal                0.82      0.28
1       systemd                        0.47      0.34
167     dbus-daemon                    0.12      0.13
249     unattended-upgr                0.00      0.61
156     systemd-resolve                0.00      0.35
```

the process id 52 killed



```
================ System Monitor Tool ===============
CPU Overall: 3.30% | Memory Used: 465.88 MB / 3583.21 MB
Commands: k <pid>  -> kill PID | r -> refresh now | q -> quit

PID     NAME                         CPU(%)    MEM(%)
------------------------------------------------------------
581     systemd-journal              0.82      0.28
1       systemd                      0.47      0.34
167     dbus-daemon                  0.12      0.13
249     unattended-upgr              0.00      0.61
156     systemd-resolve              0.00      0.35      q
```

command q to quit



```
================ System Monitor Tool ===============
CPU Overall: 0.42% | Memory Used: 458.82 MB / 3583.21 MB
Commands: k <pid>  -> kill PID | r -> refresh now | q -> quit

PID     NAME                         CPU(%)    MEM(%)
------------------------------------------------------------
100     systemd-udevd                0.04      0.17
249     unattended-upgr              0.00      0.61
156     systemd-resolve              0.00      0.35
1       systemd                      0.00      0.34
192     wsl-pro-service              0.00      0.32
395     systemd                      0.00      0.31
581     systemd-journal              0.00      0.28
183     systemd-logind               0.00      0.23
157     systemd-timesyn              0.00      0.21
576     sudo                         0.00      0.19
218     rsyslogd                     0.00      0.15
349     bash                         0.00      0.14
167     dbus-daemon                  0.00      0.13
416     bash                         0.00      0.13
350     login                        0.00      0.12
578     monitor                      0.00      0.11
396     (sd-pam)                     0.00      0.10
577     sudo                         0.00      0.07
166     cron                         0.00      0.07
6       init                         0.00      0.05

Exiting System Monitor.
ritika@JayKeshav:/mnt/c/System_monitor_tool$
```

The program exited or quited

Conclusion

This project successfully implements a Linux-based system monitor using C++. It enhances
understanding of Linux OS functioning and real-time system resource monitoring.

Student details

Name – Ritika Nanda
Regd. – 2241014022
Batch - 10th
Branch – CSE