**Project Proposal:** System Task Manager for Resource Monitoring and Process Management

## Project Summary:

This project aims to develop a comprehensive, cross-platform task manager tool that integrates essential features of Windows Task Manager, Linux Task Manager, and macOS Activity Monitor. Written in Bash, the tool will provide an intuitive command-line interface (CLI) for monitoring system resources, managing processes, and interacting with open applications. The system task manager will offer functionality for resource tracking, process control, and application management, targeting users who need a lightweight yet functional task manager accessible through the command line.

## Project Objectives:

The primary objectives of this project are:

- **Process Management:** Provide a user-friendly interface for viewing active processes, their resource consumption, and controlling processes (e.g., killing, switching focus).
- **System Resource Monitoring:** Continuously monitor and display key system statistics, including CPU, memory, disk, and network usage.
- **Cross-Platform Compatibility:** Design the tool to work seamlessly on major operating systems—Linux, macOS, and limited compatibility on Windows via WSL (Windows Subsystem for Linux).
- **Customizable Interface:** Create an attractive, CLI-based interface with color-coded, well-organized sections for easy readability and quick access to critical information.
- **Modular and Scalable:** Structure the codebase to allow for future extensions, such as real-time graphical monitoring, system alerts, and custom scripts.

## Project Scope:

The project will focus on building a functional task manager prototype with core features as listed below:

### System Resource Overview:

- Real-time tracking of CPU and memory usage.
- Disk usage and network activity monitoring.
- Battery status (for applicable systems) and system uptime.

**Process Management:**

- Display a list of top processes by CPU and memory usage.
- Allow users to kill or restart processes by entering process IDs.
- Integration with open applications management, including the ability to bring specific applications into focus (for Linux environments).

**User Interface:**

- A clean, color-coded command-line interface.
- Separate, well-labeled sections for system stats, processes, and options.
- Interactive menu for refreshing data, killing processes, and exiting the tool.

**Methodology:**

The project will follow a structured approach to achieve its objectives:

**Requirements Analysis:**

Conduct research on user needs and functionalities of popular task managers across platforms.

Identify compatible tools and commands across Linux, macOS, and Windows environments (using WSL for Windows compatibility).

**Design and Planning:**

Design the CLI layout to organize different components: system stats, process list, and open applications.

Plan modular functions for each feature, e.g., system monitoring, process management, and UI display.

**Development:**

Implement individual features in Bash scripts, using tools like \`ps\`, \`top\`, \`df\`, \`ifconfig\`, \`wmctrl\`, and \`xdotool\` for gathering and managing system data.

Test cross-platform compatibility and adjust commands for platform-specific behavior.

**Testing and Optimization:**

Test each module on Linux, macOS, and WSL to ensure compatibility.

Optimize for performance, ensuring the tool runs smoothly with minimal CPU or memory overhead.

**Risk Analysis:**

The main risks include:

- Platform-Specific Limitations: Commands in Bash may not function identically across all platforms, potentially limiting some functionalities (especially on Windows).
- User Learning Curve: Users new to CLI-based interfaces may require guidance in navigating and using the tool.
- Performance Overhead: High resource usage could affect system performance, especially on lower-spec systems.

**Conclusion:** The Cross-Platform System Task Manager project seeks to bridge the functionalities of traditional GUI task managers with a streamlined, CLI-based experience. By leveraging Bash scripting and system commands, this tool will empower users with quick access to system and process information, making it a valuable resource for monitoring and managing resources across different operating systems. This project combines practical utility with minimal overhead, resulting in a lightweight, versatile tool for power users, developers, and system administrators.