# Ultra Task Manager

System Monitoring & Visualization Suite

**Version:** 2.0

## 1. Project Overview

**Ultra Task Manager** is a powerful, hybrid system monitoring solution designed to provide real-time insights into system performance. It seamlessly acts as a bridge between low-level system metrics and high-level data visualization.

The system utilizes a dual-architecture approach:

- **Native Backend (Bash Script):** Directly interacts with the Linux kernel (via `/proc`) and hardware sensors to gather accurate, low-latency data.
- **Modern Frontend (Node.js):** Processes the logs generated by the backend and serves a responsive, interactive web dashboard for easy data consumption.

## 2. Key Features

- **Real-time CPU Monitoring:** Validates load percentages and CPU temperatures.
- **Memory Analytics:** Tracks RAM usage efficiency.
- **Disk Health (S.M.A.R.T.):** Monitors drive status and alerts on potential failures.
- **Network Traffic:** Measures real-time upload and download speeds.
- **Visual Graphing:** Historic data visualization via the web interface.
- **Desktop GUI:** Native Linux interface using `yad` for administration.

# 3. Prerequisites

Ensure the following components are installed on the host system:

- **Operating System:** Linux (Ubuntu/Debian recommended) or Windows with WSL2.
- **Runtime:** Node.js (v14.0.0+) and npm.
- **Utilities:** `yad`, `lm-sensors`, `sysstat`.

# 4. Installation Guide

### Step 1: Get the Code

```
git clone https://github.com/rukaiafawzy/taskmgrr.git
cd taskmgrr
```

### Step 2: Install Dependencies

```
# Navigate to server directory
cd server
npm install
```

### Step 3: Setup Environment

```
# Install required Linux tools
sudo apt update
sudo apt install -y yad lm-sensors nodejs npm
```

# 5. User Manual

### Running the Data Logger (Desktop Mode)

This component is responsible for collecting data. It must be running for the dashboard to show live metrics.

```
chmod +x taskmgrr.sh
./taskmgrr.sh
```

*This will launch the desktop control panel and begin logging data to `~/.taskmgrr_data/`.*

### Running the Web Dashboard

Launch the web server to visualize the collected data.

```
node server.js
```

open your browser and navigate to:
**http://localhost:3000**

## 6. Troubleshooting

- **Missing Hardware Data:** If running on WSL (Windows Subsystem for Linux), direct access to hardware sensors (like GPU temps) is restricted. The system will fall back to safe simulation estimates.
- **Port Conflict:** If port 3000 is in use, modify the `const port = 3000` line in `server.js`.

---