# System Stats & Management

## Overview

- [\*\*DietPi-Dashboard - Official lightweight standalone DietPi web interface\*\*](#dietpi-dashboard)

- [\*\*DietPi-CloudShell - Lightweight system stats for your LCD display or monitor\*\*](#dietpi-cloudshell)

- [\*\*Linux Dash - Web interface system stats\*\*](#linux-dash)

- [\*\*phpSysInfo - Web interface system stats\*\*](#phpsysinfo)

- [\*\*RPi-Monitor - Web interface system stats\*\*](#rpi-monitor)

- [\*\*Netdata - Web interface system stats\*\*](#netdata)

- [\*\*Webmin - Remote system management with web interface\*\*](#webmin)

- [\*\*K3s - Lightweight Kubernetes\*\*](#k3s)

??? info "How do I run \*\*DietPi-Software\*\* and install \*\*optimised software\*\* items?"

To install any of the \*\*DietPi optimised software items\*\* listed below run from the command line:

```sh

dietpi-software

```

Choose \*\*Browse Software\*\* and select one or more items. Finally select `Install`.

DietPi will do all the necessary steps to install and start these software items.

![DietPi-Software menu screenshot](../assets/images/dietpi-software.jpg){: width="643" height="365" loading="lazy"}

To see all the DietPi configurations options, review the [DietPi Tools](../../dietpi\_tools/) section.

[Return to the \*\*Optimised Software list\*\*](../../software/)

## DietPi-Dashboard

DietPi-Dashboard is a very lightweight and standalone web interface for monitoring and managing your DietPi system with your favourite web browser. It is written in Rust. An overview of its features is given by our article [here](https://dietpi.com/blog/?p=1137).

![DietPi-Dashboard screenshot](../assets/images/dietpi-dashboard.jpg){: width="700" height="346" loading="lazy"}

!!! warning "DietPi-Dashboard is still in Beta!"

We hence do not recommend to actively use it on sensitive production systems yet.

=== "Web interface"

DietPi-Dashboard is accessible by default at TCP port \*\*5252\*\*:

- URL: `http://<your.IP>:5252`

- Password: `<your software password>` (default: `dietpi`)

=== "Directories"

The DietPi-Dashboard executable and its configuration file can be found at:

```

/opt/dietpi-dashboard

```

=== "Configuration"

The configuration file is located at:

```

/opt/dietpi-dashboard/config.toml

```

When doing changes, you need to restart the service afterwards:

```sh

systemctl restart dietpi-dashboard

```

=== "Password protection"

Password protection is enabled by default from DietPi v7.9 on. If you installed it before, you'll need to enable it via configuration file. For this, create a SHA512 hash of the `PASSWORD` you want to use for logging into the web interface, and a random 64-character secret which is used to generate a token to transfer and store securely in your browser. Apply those to the configuration file and restart the service for the changes to take effect:

```sh

hash=$(echo -n 'PASSWORD' | sha512sum | mawk '{print $1}')

secret=$(openssl rand -hex 32)

G\_CONFIG\_INJECT 'pass[[:blank:]]' 'pass = true' /opt/dietpi-dashboard/config.toml

GCI\_PASSWORD=1 G\_CONFIG\_INJECT 'hash[[:blank:]]' "hash = \"$hash\"" /opt/dietpi-dashboard/config.toml

GCI\_PASSWORD=1 G\_CONFIG\_INJECT 'secret[[:blank:]]' "secret = \"$secret\"" /opt/dietpi-dashboard/config.toml

unset -v hash secret

systemctl restart dietpi-dashboard

```

To change the password, just replace the hash in the config file and restart the service.

If you want to force a logout of all browsers without changing the password, you can instead change the secret. Generate an apply a new secret to the configuration file and restart the service. Every client and browser will then need to login again to continue using the DietPi-Dashboard, as the stored token that is based on password and secret has been invalidated.

=== "Multiple nodes"

From DietPi v8.0 on, you can install DietPi-Dashboard as backend only node, with does not include an own web interface. Such backend only nodes can then be accessed from another full DietPi-Dashboard frontend/web interface. Additional nodes would need to be added manually into configuration file located at:

```

/opt/dietpi-dashboard/config.toml

```

When doing changes, you need to restart the service afterwards:

```sh

systemctl restart dietpi-dashboard

```

!!! hint "Full DietPi-Dashboard nodes with frontend included can currently not be accessed from other frontends."

=== "Service control"

DietPi-Dashboard by default is started as systemd service and can hence be controlled with the following commands:

```sh

systemctl status dietpi-dashboard

```

```sh

systemctl stop dietpi-dashboard

```

```sh

systemctl start dietpi-dashboard

```

```sh

systemctl restart dietpi-dashboard

```

=== "Logs"

Service logs can be reviewed with the following command:

```sh

journalctl -u dietpi-dashboard

```

=== "Update"

You can easily update DietPi-Dashboard by reinstalling it:

```sh

dietpi-software reinstall 200

```

\*\*\*

Source code: <https://github.com/ravenclaw900/DietPi-Dashboard>

License: [GPLv3](https://github.com/ravenclaw900/DietPi-Dashboard/blob/main/LICENSE)

## DietPi-CloudShell

DietPi-CloudShell turns your console or LCD screen into a lightweight system stats display.

### Sample screenshots

The following screenshots shall give an overview over the displaying features of \*DietPi-CloudShell\*.

=== "CPU usage"

![DietPi-CloudShell CPU usage dialog](../assets/images/dietpi-software-systemstat-cloudshare-cpuusage.jpg){: width="400" height="305" loading="lazy"}

=== "Memory usage"

![DietPi-CloudShell memory usage dialog](../assets/images/dietpi-software-systemstat-cloudshare-memoryusage.jpg){: width="400" height="293" loading="lazy"}

=== "Storage details"

![DietPi-CloudShell storage details dialog](../assets/images/dietpi-software-systemstat-cloudshare-storagedetails.png){: width="400" height="292" loading="lazy"}

=== "Network details"

![DietPi-CloudShell network details dialog](../assets/images/dietpi-software-systemstat-cloudshare-networkstats.jpg){: width="400" height="303" loading="lazy"}

=== "Pi-hole stats"

![DietPi-CloudShell Pi-hole dialog](../assets/images/dietpi-software-systemstat-cloudshare-piholestats.jpg){: width="400" height="305" loading="lazy"}

\*\*\*

YouTube video tutorial: \*DietPi CloudShell (RPi / Odroid XU4)\*

<iframe src="https://www.youtube-nocookie.com/embed/O-W8Z33as\_U?rel=0" frameborder="0" allow="fullscreen" width="560" height="315" loading="lazy"></iframe>

### Configuration

=== "Configuration"

Launch the DietPi-CloudShell control panel:

```sh

dietpi-cloudshell

```

![DietPi-CloudShell main menu](../assets/images/dietpi-software-systemstat-cloudshare-configuration.png){: width="600" height="298" loading="lazy"}

=== "Scenes"

DietPi-CloudShell offers scenes with predefined display outputs resp. layouts.

Scenes are configured in the \*Scenes\* dialog within `dietpi-cloudshell`.

![DietPi-CloudShell scenes menu](../assets/images/dietpi-software-systemstat-cloudshare-scenes.png){: width="600" height="283" loading="lazy"}

=== "Power saving"

This feature will allow you to automatically power down the screen and disable DietPi-CloudShell processing during a specific time.

Please note this feature requires DietPi-CloudShell to be launched with `dietpi-autostart`, or run `dietpi-cloudshell` from the main screen (`tty1`).

If you make changes to DietPi-CloudShell over SSH, once saved, please reboot the system to ensure this feature becomes activated.

Power saving: `Auto screen off`

### Touchscreen support

=== "Waveshare32"

See <https://www.waveshare.com/3.2inch-rpi-lcd-b.htm>.

This is available for all Raspberry Pi and Odroid versions. DietPi will automatically configure your system for the device.

Simply run `dietpi-config`, select `Display Options`, then select `waveshare32`.

After a reboot, your \*Waveshare32\* will become active.

![DietPi-CloudShell on Waveshare32 touchscreen photo](../assets/images/dietpi-software-systemstat-cloudshell-wavesharesupport.png){: width="400" height="258" loading="lazy"}

=== "Odroid 3.5 LCD shield"

See <https://www.hardkernel.com/shop/c1-3-2inch-tfttouchscreen-shield/>.

This is available for all Odroid versions. DietPi will automatically configure your system for the device.

Simply run `dietpi-config`, select `Display Options`, then select `odroid-lcd35`.

After a reboot, your \*Odroid 3.5 LCD\* will become active.

![DietPi-CloudShell on Odroid 3.5 LCD photo](../assets/images/dietpi-software-systemstat-cloudshell-touchscreensupport.jpg){: width="400" height="224" loading="lazy"}

=== "Other touchscreens"

The display feature of \*DietPi-CloudShell\* works generally with any LCD display or monitor with a resolution of at least 320x240 pixels.

## Linux Dash

Linux Dash allows you to monitor your system stats from a web page.

- Also Installs: [LASP webserver stack](../webserver\_stack/)

![Linux Dash web interface screenshot](../assets/images/dietpi-software-systemstat-linuxdash.png){: width="500" height="270" loading="lazy"}

=== "Access to Linux Dash"

The web interface of \*Linux Dash\* can be accessed via:

- URL = `http://<your.IP>/linuxdash/app`

\*\*\*

Official documentation: <https://github.com/afaqurk/linux-dash/wiki>

## phpSysInfo

Allows you to monitor your system stats from a web page. The display output can be customized via an `.ini` file.

- Also Installs: [LASP webserver stack](../webserver\_stack/)

![phpSysInfo web interface screenshot](../assets/images/dietpi-software-systemstat-phpsysinfo.png){: width="500" height="268" loading="lazy"}

=== "Access to phpSysInfo"

The web interface of \*phpSysInfo\* can be accessed via:

- URL = `http://<your.IP>/phpsysinfo`

=== "Customization"

This is done via the file `phpsysinfo.ini` which is located in the phpSysInfo main directory (typical `/var/www/phpsysinfo`). An example file `phpsysinfo.ini.new` is present and gives inline information about all configuration options. Just walk through this file and discover all these bells an whistles.

\*\*\*

Official website: <https://phpsysinfo.github.io/phpsysinfo>

## RPi-Monitor

RPi-Monitor is a slick, lightweight system stats monitor with web interface.

![RPi-Monitor web interface screenshot](../assets/images/dietpi-software-systemstat-rpimonitor.png){: width="500" height="364" loading="lazy"}

=== "Main features"

The main features of \*RPi-Monitor\* are:

- Collecting, storing and presenting metrics

- It's flexibly configurable

- It's user extendable

- Metrics sharing can be done via JSON file or via SNMP

- Alert option

=== "Access to RPi-Monitor"

The web interface is accessible via port \*\*8888\*\*:

- URL = `http://<your.IP>:8888`

=== "Configuration"

The configuration is described there: <https://xavierberger.github.io/RPi-Monitor-docs/20\_index.html>

\*\*\*

Official website: <https://github.com/XavierBerger/RPi-Monitor>.

## Netdata

Netdata is a slick and feature-rich system stats monitor, with web interface.

![Netdata web interface screenshot](../assets/images/dietpi-software-systemstat-netdata.png){: width="500" height="260" loading="lazy"}

=== "Access to Netdata"

The web interface is accessible via port \*\*19999\*\*:

- URL = `http://<your.IP>:19999`

=== "Troubleshooting"

Depending on your system, by default Netdata might be not accessible from remote browsers. If this is the case, open the configuration file

`/etc/netdata/netdata.conf`

and change the line

`bind socket to IP = 127.0.0.1`

to match either the local network IP or static public IP of your DietPi server depending on your needs.

Alternatively comment it e.g. if your server does not have a static public IP but you require remote access. But note that an unprotected publicly accessible Netdata web interface is a potential security risk. Head over to the "Security hardening" tab to learn more about how to restrict access to Netdata.

Once saved, you will need to restart the service to implement the changes by entering the below into the terminal:

```sh

systemctl restart netdata

```

=== "Security hardening"

Note that having Netdata accessible to anyone gives potential attackers a bunch of useful information where to start hacking.

To learn about how to limit access to Netdata, please refer to their documentation regarding [the configuration of access lists](https://learn.netdata.cloud/docs/agent/web/server/#access-lists).

\*\*\*

Official documentation: <https://learn.netdata.cloud/docs/overview/what-is-netdata>

Wikipedia: <https://wikipedia.org/wiki/Netdata>

## Webmin

Webmin is a web-based feature-rich remote system management tool. Many system settings can easily be set using the web interface dialogues.

![Webmin interface screenshot](../assets/images/dietpi-software-systemstat-webmin.png){: width="500" height="276" loading="lazy"}

=== "Access to Webmin"

The web interface is accessible via port \*\*10000\*\*:

- URL = `https://<your.IP>:10000`

- Username = `root`

- Password = `<your software password>` (default: `dietpi`)

???+ hint "Use HTTPS"

Please ensure `https://` URL is typed, `http://` won't work!

=== "System logging"

The Webmin system logging panel still depends on a classical file logger, like Rsyslog. However DietPi will not impose this logging overhead on systems by default. If you need to see system logs via the \*Webmin online panel\*, you can either configure a custom \*syslog\* daemon or install \*Rsyslog\* manually:

```sh

apt install rsyslog

```

DietPi comes with \*systemd\* and the related \*journald\* system logger, which can be accessed via the command `journalctl`.

\*\*\*

Official website: <https://webmin.com/>

<!--Official documentation: <https://doxfer.webmin.com/Webmin/Main\_Page> -->

Wikipedia: <https://wikipedia.org/wiki/Webmin>

## K3s

Lightweight Kubernetes - The certified Kubernetes distribution built for IoT & Edge computing

![K3s logo](../assets/images/logo-k3s.svg){: width="300" height="116" loading="lazy"}

=== "Before installation"

The default installation of K3s creates a single-node cluster.

If you want to have a multi-node setup, you need to configure the nodes to speak to the others.

In `/boot/dietpi.txt`, edit the `SOFTWARE\_K3S\_EXEC` parameter to set command (`server` or `agent`).

You can add other command-line parameters after the command.

Example:

```

SOFTWARE\_K3S\_EXEC=server --disable=local-storage

```

If you need to add many command-line parameters, it is recommended to put them in a file instead,

keeping only the command (`server` or `agent`) in `/boot/dietpi.txt`.

During installation, if `/boot/dietpi-k3s.yaml` exists, it is copied to `/etc/rancher/k3s/config.yaml`, and used by K3s.

The format of this file is documented in the [K3s docs](https://rancher.com/docs/k3s/latest/en/installation/install-options/#configuration-file).

=== "Connecting to your cluster"

When running in `server` mode, K3s generates a `kubeconfig` file at `/etc/rancher/k3s/k3s.yaml`.

Copy this to your client machine, and edit the `server` setting to point to the hostname of the server.

Place the file in the default location (`~/.kube/config`), or point to it using the `KUBECONFIG` environment-variable.

You should now be able to interact with your Kubernetes cluster using `kubectl`:

```sh

kubectl get nodes

kubectl get pods -A

```

=== "View logs"

- Service: `journalctl -u k3s`

\*\*\*

Official website: <https://k3s.io>

Official documentation: <https://rancher.com/docs/k3s/latest/en/>

Source code: <https://github.com/k3s-io/k3s>

License: [Apache 2.0](https://github.com/k3s-io/k3s/blob/master/LICENSE)

[Return to the \*\*Optimised Software list\*\*](../../software/)