# BUILD A DIY SERVER

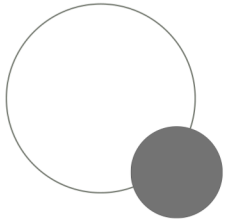
a beginner's guide

by harriethw

If you've picked up a hardcopy of this zine, you can find a digital version at

www.harriethw.com/server-zine.html

# WHY BUILD A SERVER?



*What even is a server?*

A server is simply a computer or software that can talk or be talked to by other computers or software - for example, a computer might serve up a website when queried by a browser

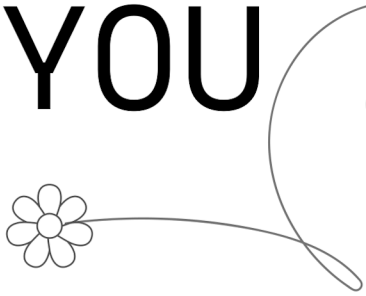
The modern tech landscape has abstracted the web to the ‘cloud’, but the internet is still a network of physical computers, data centres and cables.

Setting up your own server (in this case specifically a web server) is a great opportunity to learn about the reality of how the internet is made, have control over how your data is maintained and made available, and build something cool.

*Who is this for?*

Anyone who has an interest in the topic! No previous knowledge is necessary. If you’re already experienced with networks you probably don’t need a lot of the explanation but it might provide a handy to-do list.

# 1. WHAT YOU NEED



*Tiny computer stuff*

- I'll be using a Raspberry Pi Zero W from 2020 in this zine, but there are tons of others and alternative boards you could use)
- A charging cable / power source for your board
- Micro SD Card (I'm using a SanDisk but again there are alternatives, make sure you get a card that will fit your board)
- (optional) a little protective case

Lots of this stuff can be available second hand, or if you're in the UK Pimoroni is a great indie shop https://shop.pimoroni.com/

*Other necessities*

- WiFi or an internet connection (and if you want to make your server public, access to the modem settings)
- A computer with a screen that you can use the terminal or command line interface on (don't worry if you don't know what this means yet!)
- A way to insert your Micro SD Card into your big computer.
- A cable to connect your micro computer to the computer with a screen.

## Time, patience and google

You could take several hours or days to follow these instructions. Hopefully they are easy to put down and pick up again.

This is just a rough guide from my personal experience, and I'm not an expert so things could be wrong or out of date - if something goes wrong, please remain calm!

Remember the tool that even the most senior engineers use all the time - the internet. Search for key phrases, error messages or acronyms and you'll likely come across the answer or more information than you ever needed to know.

# 2. SETUP YOUR SERVER

you've got your kit and you're ready to gooooo

## *2a. Install DietPi*

DietPi is a really lightweight OS (Operating System) that's designed to need minimal computer processing power, so it's perfect for this low-tech project.

Because we are going for a low tech solution, we aren't going to want to give our micro computer a UI or screen, we're only going to interact with it by talking to it via SSH (aka Secure Shell or Secure Socket Shell).

DietPi already have some amazing and up to date docs on their site, so we're going to direct you over to them with some things to note...

*(We're following instructions in March 2023)*

- For ease of following this doc, you want to look for the ***instructions to setup WiFi***, it might be hidden so we've copied the instructions here.

    - To setup the WiFi, open the SD card folder on your computer, and update next two files using a text editor of your choice:
        - Open the file named **dietpi.txt**. Find **AUTO_SETUP_NET_WIFI_ENABLED** and set to value 1.
        - Open the file **dietpi-wifi.txt** and set **aWIFI_SSID[0]** to the name of your WiFi network.
        - In the same file **dietpi-wifi.txt**, set **aWIFI_KEY[0]** to the password of your WiFi network.
        - Save and close the files

- We're going to ***connect via SSH***, so we want to run a ***headless install***. Make sure you look for the info box about how to do that.
- Make sure you change your password from the default one given to you, for both the **root** user and **dietpi** user.

# *2b. What the hell is headless?*

i

never used the command line before? Don't panic!

Command line interfaces (CLI's) can be a bit intimidating - most people only see them on screens when someone is an elite hacker - but there's nothing mysterious about them.

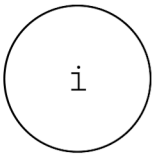
Think of it as a way to talk to your computer, but instead of using your voice, mouse or touchscreen to tell it what to do, you're using magical word spells.

Getting to grips with the CLI is super fun and unlocks a whole host of powerful and speedy tools at your fingertips.

There's a CLI cheatsheet at the end of this booklet to remind you of some commands you might use regularly. *(TBA April 2023)*

If you've never done something like this before, you may want to check out Free Code Camp or another resource on using the command line or bash scripting for beginners.

when you're ready, go to DietPi!




# IP address??

if you're having trouble finding your pi's IP address, you could perhaps look at the connected devices on your network via your internet router (see chapter below!) or download a software called **nmap** to help look at the IPs on your network.

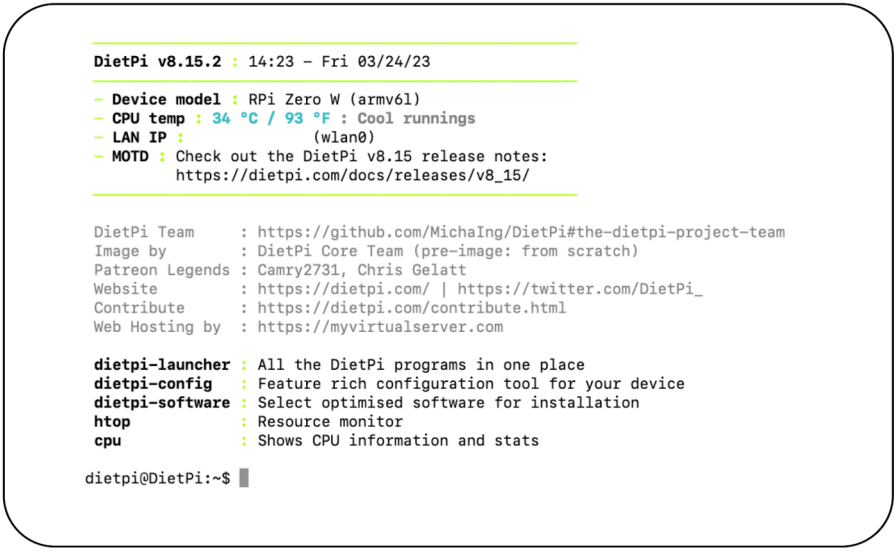
let's get into the command line!

## *2b. ssh*

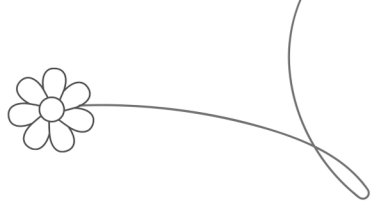
Hopefully should now be able to "**ssh**" into your pi through the terminal. Entering the below command (using your own pi's IP address) should raise a prompt for a password

**ssh dietpi@192.168.1.20**

and you should see something like this



# *2c. more software*



Woohoo, you've got your basic OS running! Now we want to install some programs onto it upfront that will make the rest of the tutorial possible.

Find the following software by running dietpi-software in the command line, choose search software and enter the names of the software.

### *Apache*

This is the server software we're going to use to run your site. As with everything in this zine, there are other options available, but this is a widely used fav and easy to setup.

### *Fail2Ban, LetsEncrypt*

These two will come in handy when we open up the site to the web. Fail2Ban protects your system from brute force attacks, and LetsEncrypt provides free HTTPS certificates

### *ProFTP*

I'll be using a File Transfer Protocol (FTP) to send our content from our regular computer to the pi. I'm using the one recommended by DietPi.

## *2c*

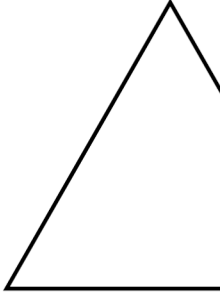
I would then suggest downloading software on your personal computer like **FileZilla** for a UI to connect to the pi.

These are the default credentials you can then use (remember to check DietPi's docs if this doesn't work).

- Username = dietpi
- Password = The same as your root login password. Default is dietpi
- Address = Your IP address (e.g.: 192.168.0.100)
- Port = 21

*Once Apache has installed you should be able to enter your pi's IP address into a browser and see a default Apache website - congrats! you're running a server! There's still a few more things you'll probably want to do...*

# 3. UPLOAD YOUR SITE



Now your Apache server should be serving the file from the default path of **/var/www/index.html**

First we need to setup permissions for the dietpi user to access the web files

**sudo usermod -a -G www-data dietpi**
**sudo chown -R -f www-data:www-data /var/www**

This adds the **dietpi** user to the group of **www-data** and gives that group permission of the files in **/var/www** directory. You should be able to see the contents of the index file by running

**sudo nano /var/www/index.html**

(You can exit by pressing ctrl + x btw)

Don't worry about having to edit your site through the terminal though! In the last step we installed an FTP server so that you can transfer files over from your personal computer to the pi using a handy UI.

First we need to tell the proFTP server to go straight to the www folder when we connect

**sudo dietpi-services stop proftpd**
**sudo sed -i '/DefaultRoot /c\DefaultRoot /var/www' /etc/proftpd/proftpd.conf**
**sudo dietpi-services start proftpd**

Then open FileZilla and create a connection, the default values are:

- Username = dietpi
- Password = The same as your root login password. Default is dietpi
- Address = Your IP address (e.g.: 192.168.0.100)
- Port = 21

You should now see Apache's **index.html** file under the 'Remote site' window in Filezilla!

You should also now be able to delete that file and replace it with one from your own computer's folders along with any other assets you need.

Remember, the server is looking for a default **index.html** file. If you don't have an HTML file ready, no worries, you can do the rest of this process and come back to add your custom site later.

***Low tech web***

As we're running your site on a raspberry pi from your own home and not a fancy cloud server, you'll want to consider the size of your website files (things like images and audio) and the processing power needed (things like javascript).

Googling 'low tech web design' will give you some great resources to make your site eco and pi friendly. Things like using plain HTML/CSS/JS or dithering images are a great way to start.

## what if I want more than one website running?

That's totally possible as well!

It just means you'll need to configure Apache's VirtualHost files. For more infor about this check out Apache's documentation on VirtualHost or find a tutorial for raspberry pi such as...

https://pimylifeup.com/raspberry-pi-apache/#setting-up-an-apache-virtual-host

# 4. OPEN UP

let the world in

## *4a. router config*

Now is probably a good time to get familiar with the router settings provided by your Internet Service Provider (ISP).

We need to make some changes because your router is going to be like a switchboard between your server and the rest of the world wide web, so we need to tell it how to direct that traffic the way we want it to.

For most ISPs you should have a UI to change your browser settings, and many use the same IP or default gateway:

192.168.0.1

if this doesn't work, and the address isn't on a sticker on your router, you might be able to find it by looking at the wifi details in your system settings, or google your ISP name and router address.

Once you have it, popping that IP address in your browser should take you to the settings for it.

*Reserving an IP address*

To keep things easy, let's tell the router to reserve an IP address for our pi, so it always uses the same address to connect to.

For a Virgin Media router, I can go to the menu Advanced Settings -> DHCP and it gives me the option to select connected devices and add a 'rule'

*Port Forwarding*

For Virgin Media routers, this is under the menu for Advanced Settings -> Security -> Port forwarding. You should see a table where you can create a new rule.

Basically you want to tell the router when traffic comes through the port ranges for 80-80 and 443-443 that it knows to direct it to the right ports on your IP address of your PI.

So for the web to access your pi via your IP address you'll want to set the 'external' port ranges of 80-80/443-443 to the local port ranges of 80-80/443 and the IP of your pi.

That's it! Now if you access the IP address of your router (which you can find by googling 'what's my IP') from another internet connection, your website will be served! hurray!

***Handy guide to ports***

Ports have unique numbers and they are a bit like doors for your server, entering through the right door tells the server which corridor to take you down...

80 - the default port used by web servers. Your browser will use this port to try and talk to your router.
443 - the default port for secure connections to servers, so it’s used by https
21 - the default FTP port

You can change these default ports via dietpi as an option for added security (e.g. a bot won't be able to target your FTP because it won't know what port its on.

# *4b. getting a domain name*

*Dynamic DNS*

That's great that we can serve a site on an IP address, but most humans don't use IP addresses to browse the web. We want a nice readable domain name like www.example.com

For this we'll need a DNS service. DNS stands for Domain Name System - it's commonly called the phonebook of the internet. When you put in a url like example.com DNS servers link that domain to an IP address… it's a really fun topic to look into and I recommend googling it if you're interested in finding out more!

The IP address we're using is provided by your ISP, and it might change at any time. This is totally normal and wouldn't usually affect you if you weren't trying to connect a domain name to it. Regular DNS servers might take a while to properly change over when you change the IP address associated with a domain, but using a Dynamic DNS service will make things a lot quicker and easier.

## *4b*

I recommend checking out Dynu, they offer free dynamic DNS and you can buy your domain name from them as well (other suppliers are available of course, you can buy your domain name from elsewhere if it's cheaper and still use it with Dynu).

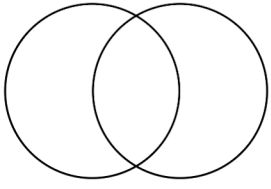
https://www.dynu.com/

*DNS records*

Once you've bought your domain name, you'll want to set up DNS records for it to tell the service to point your domain to your IP address.

As we're pointing to an IP, we'll be using *A records*.

For my domain www.poem.garden I've got A records for "poem.garden" and "www.poem.garden"



## *4c. SSL Certificate*

You may have noticed when visiting your site that it does not have the little padlock next to it's address in the browser. That's because it doesn't have an "SSL certificate" to access it via https. This is like passing an ID card between your server and the browser that says it's trustworthy.

Way back a few chapters ago we installed the software Let's Encrypt, and that will allow us to create our own certificate for free. Run:

sudo dietpi-letsencrypt

And follow the setup instructions for your domain. Or check out the info on dietpi here

https://dietpi.com/docs/software/system_security/#lets-encrypt

# THANK YOU

This is a living document and still a work in progress.

Follow me for updates or send questions to

https://mas.to/@harriethw