First, get a fresh installation of debian. You can do this on a new computer, e.g. a raspberry pi, or by following the steps in my video, "How to Setup a Remote Desktop in Bitclouds."

Log into your debian machine as root and update it.

```
ssh root@ipaddress
apt update -y && apt upgrade -y && apt autoremove -y
```

Install php (the language wordpress is written in), the php-fpm plugin (which lets php run in the background), the php-mysql plugin (which lets php work with some database software that you will probably install later if you decide to build a wordpress site), and nginx, which we will use as our webserver.

```
apt install php php-fpm php-mysql nginx -y
```

Configure nginx. (If you are doing this on debian 9, see this document.) First remove the default configuration file, because by default nginx isn't set up to work with php, and then replace it by running the subsequent "echo" commands, which will create a configuration file that is set up to work with php.

```
rm /etc/nginx/sites-enabled/default
echo "server {" >> /etc/nginx/sites-enabled/default
echo "        listen 80 default_server;" >> /etc/nginx/sites-enabled/default
echo "        listen [::]:80 default_server;" >> /etc/nginx/sites-enabled/default
echo "" >> /etc/nginx/sites-enabled/default
echo "        root /var/www/html;" >> /etc/nginx/sites-enabled/default
echo "        index index.php index.html index.htm index.nginx-debian.html;" >>
/etc/nginx/sites-enabled/default
echo "" >> /etc/nginx/sites-enabled/default
echo "        server_name _;" >> /etc/nginx/sites-enabled/default
echo "" >> /etc/nginx/sites-enabled/default
echo "        location / {" >> /etc/nginx/sites-enabled/default
echo "                try_files \$uri \$uri/ /index.php?q=\$uri&\$args;" >>
/etc/nginx/sites-enabled/default
echo "        }" >> /etc/nginx/sites-enabled/default
echo "" >> /etc/nginx/sites-enabled/default
echo "        location ~\.php$ {" >> /etc/nginx/sites-enabled/default
echo "                include snippets/fastcgi-php.conf;" >> /etc/nginx/sites-enabled/default
```

In the next line, "php7.3-fpm.sock;" appears. It might need to be php7.2-fpm.sock; at least that's what worked in one of my tests on ubuntu. Initially php was not working, so I found out which version of php I was running, using a command I will share in a moment. It said php7.2 on my ubuntu machine so that's what I put here and then everything worked fine. This part seems to

me to be the part of this tutorial that is most likely to change in the future. If/when php upgrades to php8 and then php9, etc., you may have to search for those instead of php7. After finding what version of php you have by running `sudo ls /run/php/`, put the version number that shows up when you run that command in place of 7.3 below.

```
echo "                fastcgi_pass unix:/run/php/php7.3-fpm.sock;" >>
/etc/nginx/sites-enabled/default
echo "        }" >> /etc/nginx/sites-enabled/default
echo "" >> /etc/nginx/sites-enabled/default
echo "}" >> /etc/nginx/sites-enabled/default
```

In case it's needed, here is the plaintext version:

```
server {
    listen 80 default_server;
    listen [::]:80 default_server;

    root /var/www/html;
    index index.php index.html index.htm index.nginx-debian.html;

    server_name _;

    location / {
        try_files $uri $uri/ /index.php?q=$uri&$args;
    }

    location ~\.php {
        include snippets/fastcgi-php.conf;
        fastcgi_pass unix:/run/php/php7.4-fpm.sock;
    }
}
```

Next, start the nginx server.

```
systemctl enable nginx && systemctl start nginx
```

Install tor, which lets you publish your website on the world wide web so that any onion-enabled browser can visit it.

```
apt install tor -y
```

Configure tor.

```
echo "HiddenServiceDir /var/lib/tor/tor_website/" >> /etc/tor/torrc
echo "HiddenServicePort 80 localhost:80" >> /etc/tor/torrc
chmod -R 700 /var/lib/tor
systemctl enable tor && systemctl start tor
systemctl restart tor
```

Technically you have a website now, though all it shows right now is the nginx welcome page. We'll change what your website shows in a minute, but you can find the onion address of your website with the following command.

```
cat /var/lib/tor/tor_website/hostname
```

The result should be something like 44jfvjqw93jflk3.onion. Assuming you see such an address, use any device that has an onion-compatible browser, such as the tor browser, to visit your onion address and view your website.

You can add whatever files you want to serve (e.g. webpages) to /var/www/html/ For this example I will download wordpress and create a basic wordpress site.

In order to run a wordpress site, we first need database software because wordpress is heavily reliant on databases. I recommend mariadb because it is free, open source, and based on the popular mysql program.

```
apt install mariadb-server mariadb-client -y
```

Run mariadb.

```
mariadb
```

Your terminal will change slightly when you run mariadb.

```
CREATE DATABASE wordpress;
```

On the next line, replace the word password with an actual, secure password.

```
CREATE USER 'admin'@'localhost' IDENTIFIED BY 'password';
```

You can enter the following commands all at once:

```
GRANT ALL ON wordpress.* TO 'admin'@'localhost';
FLUSH PRIVILEGES;
exit;
```

Great! Now your database is installed and configured. The last thing we need to do is install wordpress. First download it.

```
cd
wget https://wordpress.org/latest.tar.gz
```

Next unzip it and replace everything in your current html directory with wordpress.

```
tar xpf latest.tar.gz
rm latest.tar.gz
rm -rf /var/www/html
cp -r wordpress /var/www/html/
rm -rf wordpress
chown -R www-data:www-data /var/www/html
find /var/www/html -type d -exec chmod 755 {} \;
find /var/www/html -type f -exec chmod 644 {} \;
systemctl reload nginx
```

Now you only need to visit your website in your onion-compatible browser to finish setup. Once you're there, select English as your language and click Continue, then click Let's go! In the form that follows, keep the database name as wordpress, change the username to admin, and change the password to the one you set earlier when we were configuring mariadb. Keep the database host as localhost and the table prefix as wp_. Then hit Submit. Now click Run the Installation.

Give your site a title, e.g. Test Site, use admin as your username, and set a secure password. Type your email and then decide if you want your site to be accessible to search engines. If you don't, check the box, otherwise keep it unchecked. Click Install WordPress. You're done! You can now login and manage your wordpress site at [your onion address].onion/wp-login.php, or you can visit your onion address to view your website.