comment installer-fathom-privacy-focused-web-analytics-on-debian-12

Fathom is privacy-focused web analytics that delivers clean and concise data about your websites. It's a simple website analytics program that provides insightful reporting and metrics about your sites. Fathom is created as an alternative to Google Analytics, but it doesn't invade users' privacy and doesn’t compromise visitor data. It's GDPR compliant with no cookie banners required.

Fathom is available in two versions, the open-source version that can be installed on your server, and the pro version that you can purchase from the official Fathom website. Notable Fathom website analytics users are IBM, Laravel, GitHub, Hoshicord, BOSCH, McLaren, VueJS, and many more.

In this guide, we'll go over the Fathom Privacy Focused Web Analytics installation on the Debian 12 server step-by-step. We'll over the Fathom installation with PostgreSQL as the database and Nginx as the reverse proxy. Furthermore, we'll also secure Fathom with SSL/TLS certificates from Letsencrypt.

Prerequisites

Before proceeding, ensure you have the following:

- A Debian 12 server.
- A non-root user with administrator privileges.
- A domain name pointed to a server IP address.

Installing Dependencies

Fathom is a privacy-focused web analytics built on top of Golang and Preact. It supports databases such as PostgreSQL, MySQL, and SQLite, and can be run with Nginx reverse proxy. Now you will install package dependencies for Fathom, such as PostgreSQL, Nginx, Certbot, and UFW (Uncomplicated Firewall).

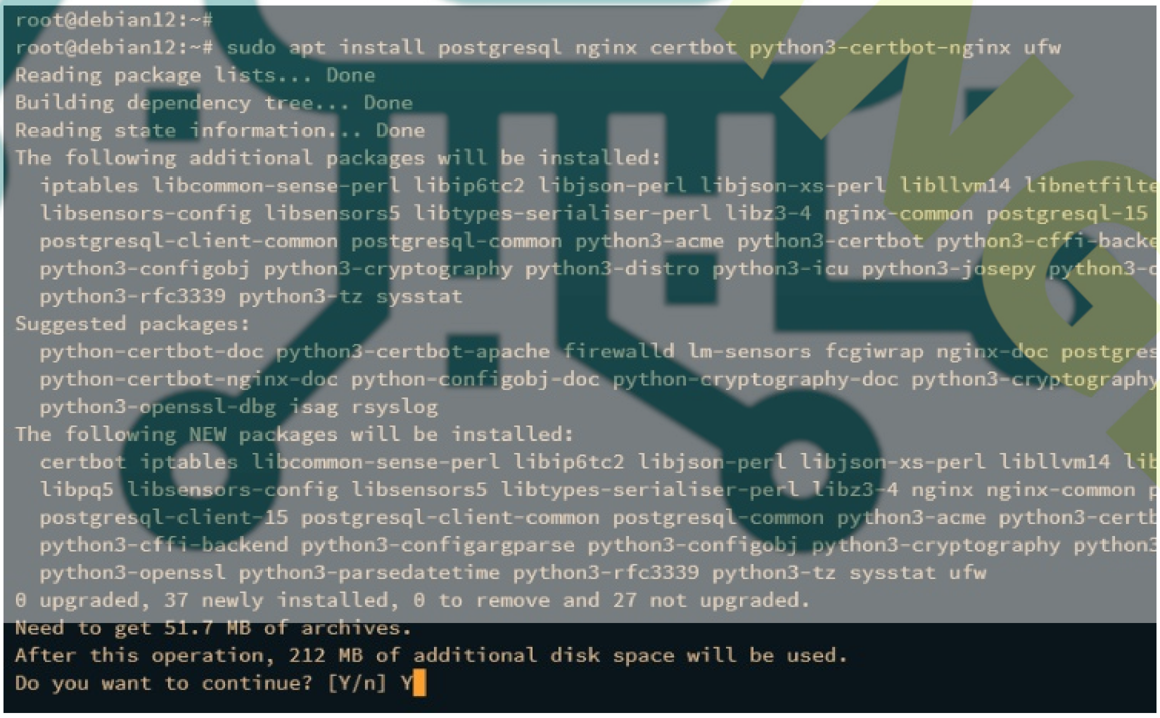
First, execute the apt command below to update your Debian repository.

```
sudo apt update
```

Now install dependencies by executing the following command. With this, you will install the PostgreSQL server, Nginx web server, Certbot, Nginx Certbot plugin, and UFW (Uncomplicated Firewall).

```
sudo apt install postgresql nginx certbot python3-certbot-nginx ufw
```

Type y to proceed with the installation.



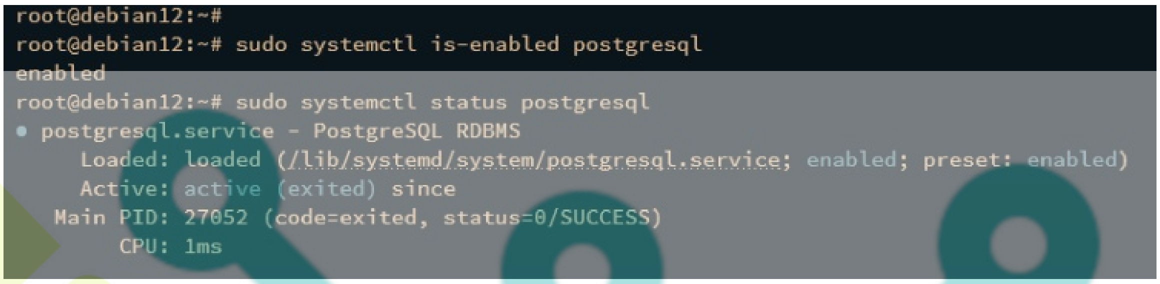
```
root@debian12:~#
root@debian12:~# sudo apt install postgresql nginx certbot python3-certbot-nginx ufw
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  iptables libcommon-sense-perl libip6tc2 libjson-perl libjson-xs-perl libllvm14 libnetfilte
  libsensors-config libsensors5 libtypes-serialiser-perl libz3-4 nginx-common postgresql-15
  postgresql-client-common postgresql-common python3-acme python3-certbot python3-cffi-backe
  python3-configobj python3-cryptography python3-distro python3-icu python3-josepy python3-c
  python3-rfc3339 python3-tz sysstat
Suggested packages:
  python-certbot-doc python3-certbot-apache firewalld lm-sensors fcgiwrap nginx-doc postgres
  python-certbot-nginx-doc python-configobj-doc python-cryptography-doc python3-cryptography
  python3-openssl-dbg isag rsyslog
The following NEW packages will be installed:
  certbot iptables libcommon-sense-perl libip6tc2 libjson-perl libjson-xs-perl libllvm14 lib
  libpq5 libsensors-config libsensors5 libtypes-serialiser-perl libz3-4 nginx nginx-common p
  postgresql-client-15 postgresql-client-common postgresql-common python3-acme python3-certb
  python3-cffi-backend python3-configargparse python3-configobj python3-cryptography python3
  python3-openssl python3-parsedatetime python3-rfc3339 python3-tz sysstat ufw
0 upgraded, 37 newly installed, 0 to remove and 27 not upgraded.
Need to get 51.7 MB of archives.
After this operation, 212 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
```

After dependencies are installed, you will ensure that every dependency is available on your system.

Verify the postgresql service using the command below to ensure that the service is running and enabled.

```
sudo systemctl is-enabled postgresql
sudo systemctl status postgresql
```

The following result will be shown on your terminal, which indicates that postgresql is running and enabled.

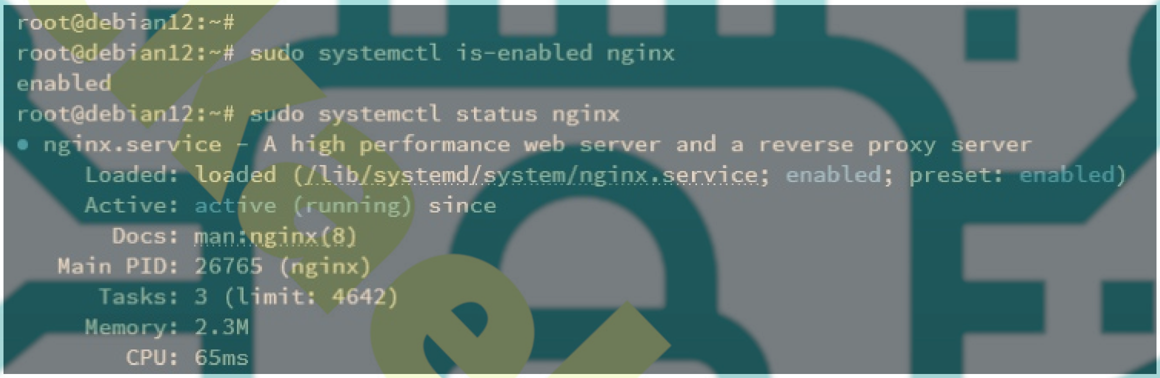


```
root@debian12:~#
root@debian12:~# sudo systemctl is-enabled postgresql
enabled
root@debian12:~# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
     Loaded: loaded (/lib/systemd/system/postgresql.service; enabled; preset: enabled)
     Active: active (exited) since
   Main PID: 27052 (code=exited, status=0/SUCCESS)
        CPU: 1ms
```

Next, verify the nginx service by executing the following command.

```
sudo systemctl is-enabled nginx
sudo systemctl status nginx
```

A similar output will be printed on your terminal that indicates the nginx service is running and enabled.

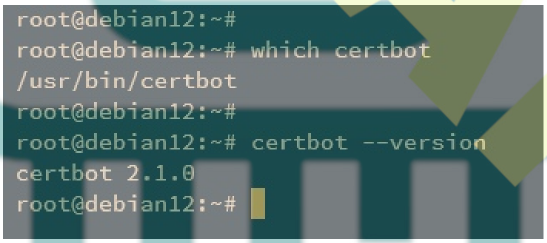


```
root@debian12:~#
root@debian12:~# sudo systemctl is-enabled nginx
enabled
root@debian12:~# sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/lib/systemd/system/nginx.service; enabled; preset: enabled)
     Active: active (running) since
       Docs: man:nginx(8)
   Main PID: 26765 (nginx)
      Tasks: 3 (limit: 4642)
     Memory: 2.3M
        CPU: 65ms
```

Lastly, verify the certbot by executing the following command. This will locate the binary executable file of certbot and check the current certbot version that is installed on your system.

```
which certbot
certbot --version
```

The displayed output will show you that Certbot **2.1** is installed at */usr/bin/certbot*.



```
root@debian12:~#
root@debian12:~# which certbot
/usr/bin/certbot
root@debian12:~#
root@debian12:~# certbot --version
certbot 2.1.0
root@debian12:~#
```

Configuring Firewall

After installing dependencies, you will configure UFW to secure your Debian machine. You will open ports for SSH, HTTP, and HTTPS on your system.

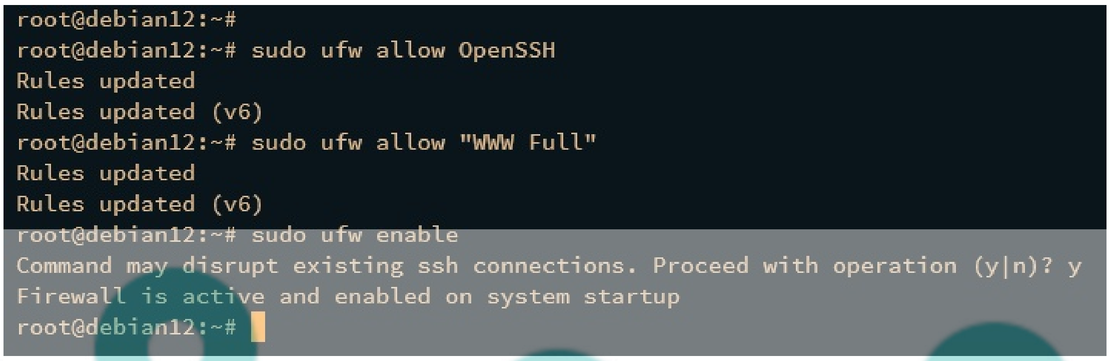
Execute the following command to open ports for SSH, HTTP, and HTTPS protocols. The OpenSSH profile will open port 22/tcp, and the WWW Full profile will open both HTTP and HTTPS - ports 80 and 443.

```
sudo ufw allow OpenSSH
sudo ufw allow "WWW Full"
```

Now run the below command to start and enable UFW on your Debian system.

```
sudo ufw enable
```

Type y to proceed with the confirmations. When successful, you should get the message "**Firewall is active and enabled on system startup**".

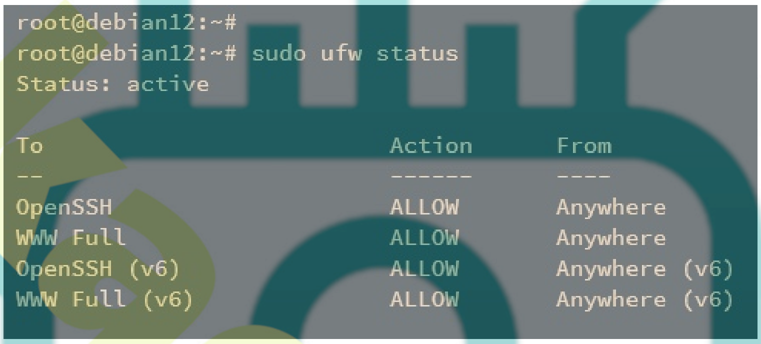


```
root@debian12:~#
root@debian12:~# sudo ufw allow OpenSSH
Rules updated
Rules updated (v6)
root@debian12:~# sudo ufw allow "WWW Full"
Rules updated
Rules updated (v6)
root@debian12:~# sudo ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
root@debian12:~#
```

Lastly, verify the UFW status using the command below.

```
sudo ufw status
```

The output active indicates that UFW is running and enabled, you should also see the **OpenSSH** and **WWW Full** profiles added to UFW.



```
root@debian12:~#
root@debian12:~# sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
OpenSSH                    ALLOW       Anywhere
WWW Full                   ALLOW       Anywhere
OpenSSH (v6)               ALLOW       Anywhere (v6)
WWW Full (v6)              ALLOW       Anywhere (v6)
```

Creating PostgreSQL Database and User

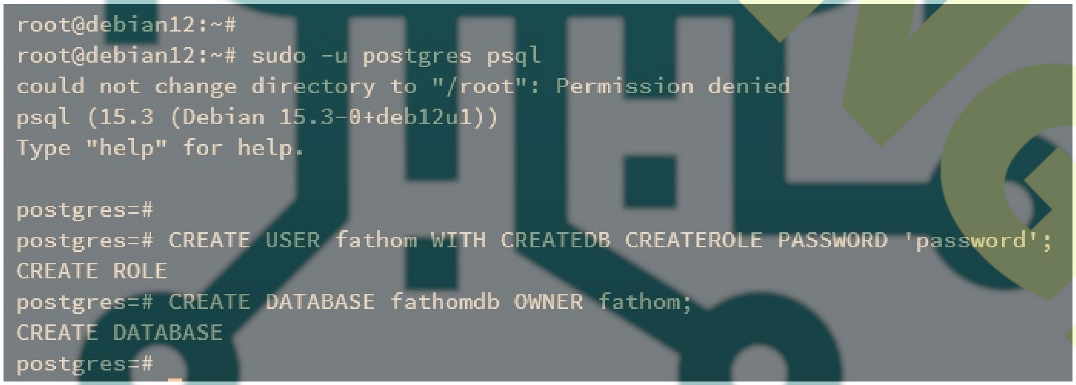
In the following step, you will create a new PostgreSQL database and user that Fathom will use. To do that, you must log in to the PostgreSQL server via the *psql* command line.

Execute the following command to log in to the PostgreSQL server.

```
sudo -u postgres psql
```

Create a new database **fathomdb** and user **fathom** using the following PostgreSQL queries. be sure to change the password in the following query.

```
CREATE USER fathom WITH CREATEDB CREATEROLE PASSWORD 'password';
CREATE DATABASE fathomdb OWNER fathom;
```

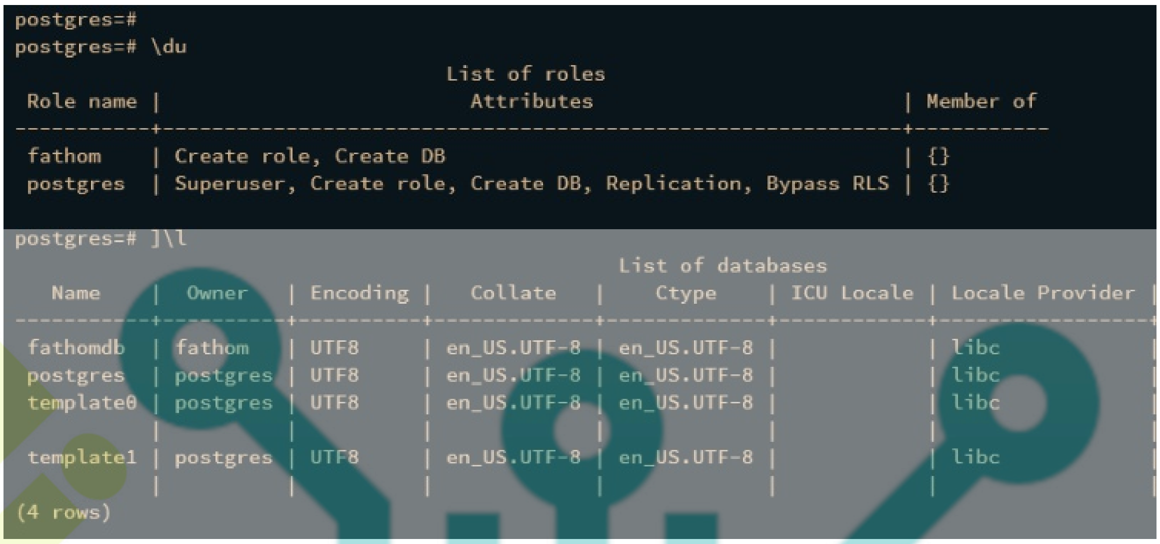


```
root@debian12:~#
root@debian12:~# sudo -u postgres psql
could not change directory to "/root": Permission denied
psql (15.3 (Debian 15.3-0+deb12u1))
Type "help" for help.

postgres=#
postgres=# CREATE USER fathom WITH CREATEDB CREATEROLE PASSWORD 'password';
CREATE ROLE
postgres=# CREATE DATABASE fathomdb OWNER fathom;
CREATE DATABASE
postgres=#
```

Now execute the following queries to verify the list of available databases and users on the PostgreSQL server.

```
\du
\l
```

If the database and user are created, the following output will be presented:



```
postgres=#
postgres=# \du
                                   List of roles
 Role name |                         Attributes                         | Member of
-----------+------------------------------------------------------------+-----------
 fathom    | Create role, Create DB                                     | {}
 postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS | {}

postgres=# ]\l
                                                  List of databases
   Name    |  Owner   | Encoding |   Collate   |    Ctype    | ICU Locale | Locale Provider |
-----------+----------+----------+-------------+-------------+------------+-----------------+
 fathomdb  | fathom   | UTF8     | en_US.UTF-8 | en_US.UTF-8 |            | libc            |
 postgres  | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |            | libc            |
 template0 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |            | libc            |
           |          |          |             |             |            |                 |
 template1 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |            | libc            |
           |          |          |             |             |            |                 |
(4 rows)
```

Type quit to exit from the PostgreSQL server.

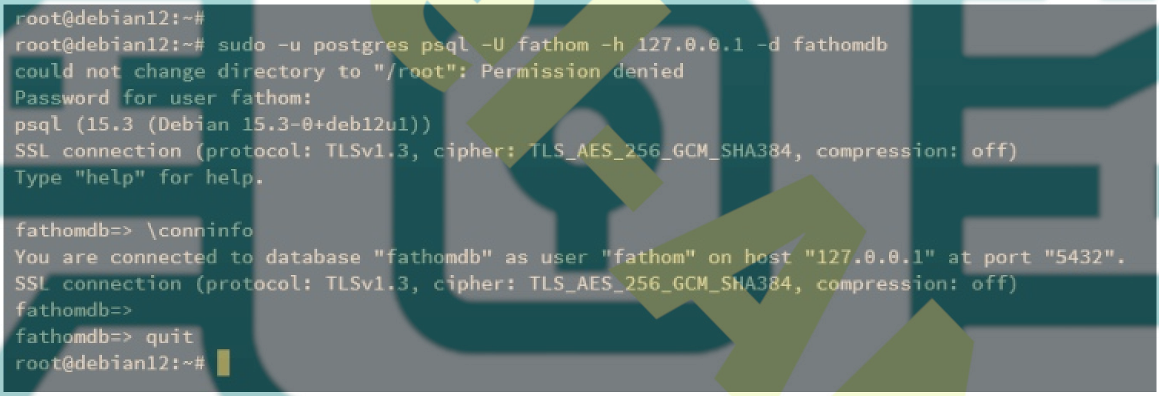
After creating the database and user, execute the following command to log in to PostgreSQL with user fathom and database fathomdb. Input the password for your database user when prompted.

```
sudo -u postgres psql -U fathom -h 127.0.0.1 -d fathomdb
```

Once connected to the PostgreSQL server, execute the following query to verify your connection.

```
\conninfo
```

After executing the command, the following output will be shown, which indicates that you're connected to the database **fathomdb** with user **fathom**.



```
root@debian12:~#
root@debian12:~# sudo -u postgres psql -U fathom -h 127.0.0.1 -d fathomdb
could not change directory to "/root": Permission denied
Password for user fathom:
psql (15.3 (Debian 15.3-0+deb12u1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression: off)
Type "help" for help.

fathomdb=> \conninfo
You are connected to database "fathomdb" as user "fathom" on host "127.0.0.1" at port "5432".
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression: off)
fathomdb=>
fathomdb=> quit
root@debian12:~#
```

Type quit again to exit from the PostgreSQL server.

Downloading Fathom Binary File

Fathom is written in Golang and Preact, and it's available as a single binary file that you can easily download and install on your system.

Visit the Fathom GitHub page and grab the download URL for the Fathom binary package. Then, download it via the wget command below. In this example, you will download Fathom 1.3.1.

```
wget https://github.com/usefathom/fathom/releases/download/v1.3.1/fathom_1.3.1_linux_amd64.tar.gz
```

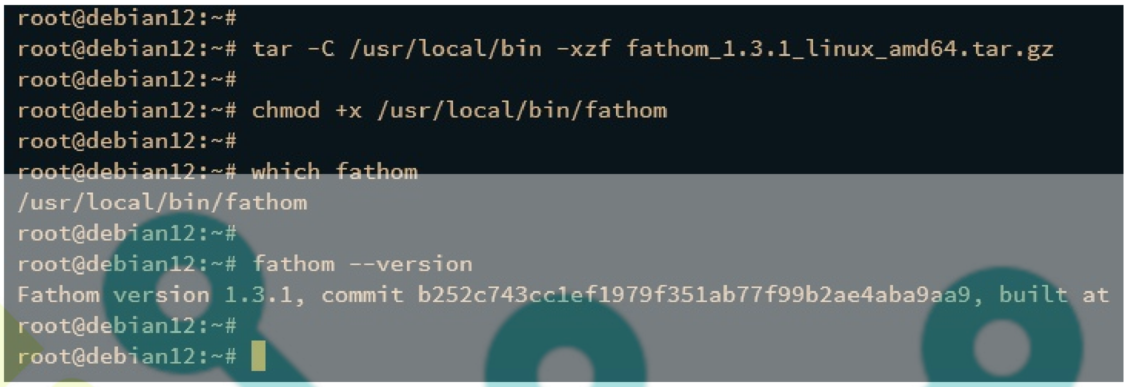
Once downloaded, extract the Fathom binary package to */usr/local/bin/fathom* and make it executable via the following command.

```
tar -C /usr/local/bin -xzf fathom_1.3.1_linux_amd64.tar.gz
chmod +x /usr/local/bin/fathom
```

Lastly, run the following command to verify the fathom binary file location and the current fathom version that you've downloaded.

```
which fathom
fathom --version
```

The following output shows you Fathom **1.3.1** is installed on */usr/local/bin/fathom*.



```
root@debian12:~#
root@debian12:~# tar -C /usr/local/bin -xzf fathom_1.3.1_linux_amd64.tar.gz
root@debian12:~#
root@debian12:~# chmod +x /usr/local/bin/fathom
root@debian12:~#
root@debian12:~# which fathom
/usr/local/bin/fathom
root@debian12:~#
root@debian12:~# fathom --version
Fathom version 1.3.1, commit b252c743cc1ef1979f351ab77f99b2ae4aba9aa9, built at
root@debian12:~#
root@debian12:~#
```

Configuring Fathom

In the following step, you will configure Fathom by:

- Integrating Fathom with PostgreSQL database.
- Running Fathom in the background as a systemd service.
- Adding administrator user for your Fathom installation.

Integration with PostgreSQL as Database

Fathom supports multiple databases, such as SQLite (default), MySQL, and MariaDB. In this section, you will set up the Fathom installation directory and integrate Fathom with PostgreSQL database server.

Execute the following command to create a new system user fathom that will be used to run your Fathom installation.

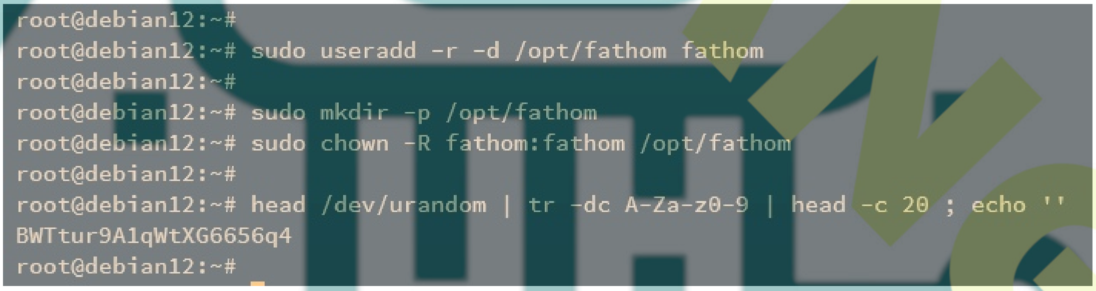
```
sudo useradd -r -d /opt/fathom fathom
```

Now create a new home directory */opt/fathom* and change the ownership to user fathom. The directory */opt/fathom* will be used for storing Fathom installation data.

```
sudo mkdir -p /opt/fathom
sudo chown -R fathom:fathom /opt/fathom
```

After that, run the command below to generate a random secret for Fathom. Be sure to copy the output because you will need it for securing fathom.

```
head /dev/urandom | tr -dc A-Za-z0-9 | head -c 20 ; echo ''
```



```
root@debian12:~#
root@debian12:~# sudo useradd -r -d /opt/fathom fathom
root@debian12:~#
root@debian12:~# sudo mkdir -p /opt/fathom
root@debian12:~# sudo chown -R fathom:fathom /opt/fathom
root@debian12:~#
root@debian12:~# head /dev/urandom | tr -dc A-Za-z0-9 | head -c 20 ; echo ''
BWTtur9A1qWtXG6656q4
root@debian12:~#
```

Next, move to the */opt/fathom* directory.

```
cd /opt/fathom
```

Then, create a new */opt/fathom/data* directory and a new file */opt/fathom/data/.env* using the following command.

```
sudo -u fathom mkdir -p /opt/fathom/data
sudo -u fathom nano /opt/fathom/data/.env
```

Insert the following configuration and be sure to change the details of the PostgreSQL database name, user, and password.

```
FATHOM_GZIP=true
FATHOM_DEBUG=true
FATHOM_DATABASE_DRIVER="postgres"
FATHOM_DATABASE_NAME="fathomdb"
FATHOM_DATABASE_USER="fathom"
FATHOM_DATABASE_PASSWORD="password"
```

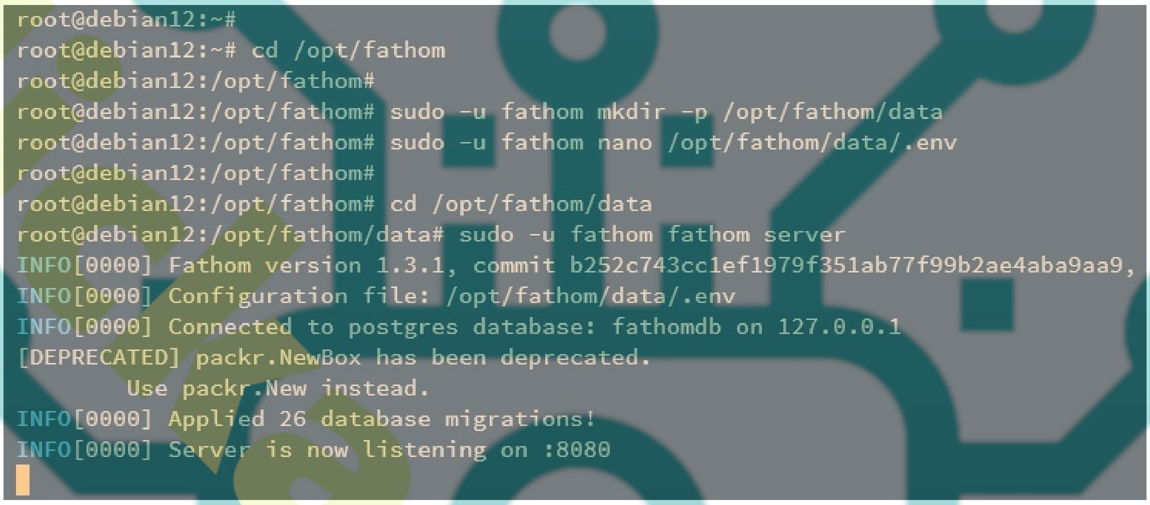
```
FATHOM_DATABASE_HOST="127.0.0.1"
FATHOM_DATABASE_SSLMODE="disable"
FATHOM_SECRET="BWTtur9A1qWtXG6656q4"
```

Save and exit the file when finished.

Lastly, run the following command to ensure that your Fathom configuration is successful.

```
cd /opt/fathom/data
sudo -u fathom fathom server
```

After executing the command, Fathom should be running on localhost with default port 8080, and the following output will be printed to your terminal:



```
root@debian12:~#
root@debian12:~# cd /opt/fathom
root@debian12:/opt/fathom#
root@debian12:/opt/fathom# sudo -u fathom mkdir -p /opt/fathom/data
root@debian12:/opt/fathom# sudo -u fathom nano /opt/fathom/data/.env
root@debian12:/opt/fathom#
root@debian12:/opt/fathom# cd /opt/fathom/data
root@debian12:/opt/fathom/data# sudo -u fathom fathom server
INFO[0000] Fathom version 1.3.1, commit b252c743cc1ef1979f351ab77f99b2ae4aba9aa9,
INFO[0000] Configuration file: /opt/fathom/data/.env
INFO[0000] Connected to postgres database: fathomdb on 127.0.0.1
[DEPRECATED] packr.NewBox has been deprecated.
         Use packr.New instead.
INFO[0000] Applied 26 database migrations!
INFO[0000] Server is now listening on :8080
```

Press Ctrl+c to terminate the process.

Running Fathom in the Background as Systemd Service

In the following section, you will create a new systemd service file that will be used to run Fathom in the background. With this, you can easily manage Fathom via the systemctl utility.

Create a new systemd service file */etc/systemd/system/fathom.service* using the following nano editor command.

```
sudo nano /etc/systemd/system/fathom.service
```

Insert the following configuration into the file.

```
[Unit]
Description=Starts the fathom server
Requires=network.target
After=network.target

[Service]
Type=simple
User=fathom
Restart=always
RestartSec=3
WorkingDirectory=/opt/fathom/data
ExecStart=/usr/local/bin/fathom server

[Install]
WantedBy=multi-user.target
```

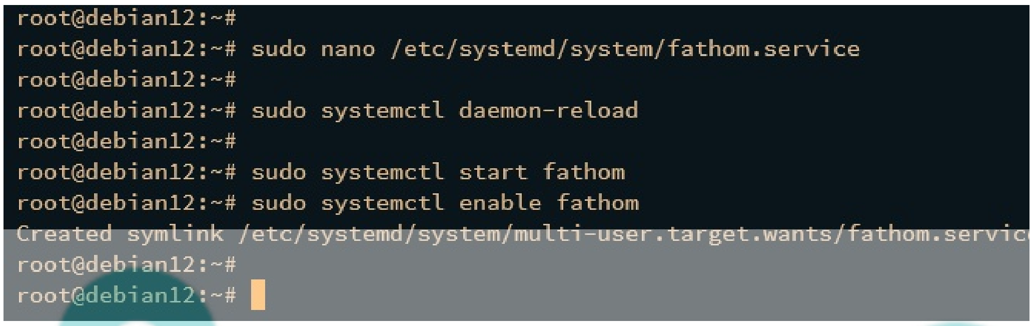
Save and close the file when you're done.

Next, run the following systemctl command to reload the systemd manager and apply the changes that you've made.

```
sudo systemctl daemon-reload
```

Once the systemd manage is reloaded, execute the *systemctl* command below to start and enable the *fathom* service. This will run Fathom in the background on localhost with default port 8080.

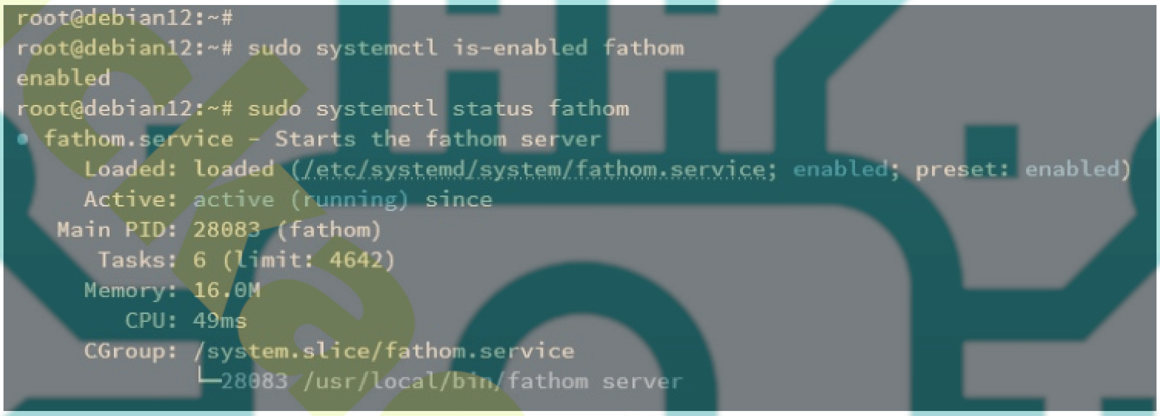
```
sudo systemctl start fathom
sudo systemctl enable fathom
```


```
root@debian12:~#
root@debian12:~# sudo nano /etc/systemd/system/fathom.service
root@debian12:~#
root@debian12:~# sudo systemctl daemon-reload
root@debian12:~#
root@debian12:~# sudo systemctl start fathom
root@debian12:~# sudo systemctl enable fathom
Created symlink /etc/systemd/system/multi-user.target.wants/fathom.servic
root@debian12:~#
root@debian12:~#
```

Verify the *fathom* service using the below command to ensure that the service is running and enabled.

```
sudo systemctl is-enabled fathom
sudo systemctl status fathom
```

The following output will be presented if fathom is running and enabled.


```
root@debian12:~#
root@debian12:~# sudo systemctl is-enabled fathom
enabled
root@debian12:~# sudo systemctl status fathom
● fathom.service - Starts the fathom server
     Loaded: loaded (/etc/systemd/system/fathom.service; enabled; preset: enabled)
     Active: active (running) since
   Main PID: 28083 (fathom)
      Tasks: 6 (limit: 4642)
     Memory: 16.0M
        CPU: 49ms
     CGroup: /system.slice/fathom.service
             └─28083 /usr/local/bin/fathom server
```

Adding Fathom Administrator User

Now that Fathom is running in the background as a service, the next section is to create an administrator user for your Fathom installation. This can be done via the fathom command line.

To create a Fathom user, you can run the fathom command line from the fathom data directory.

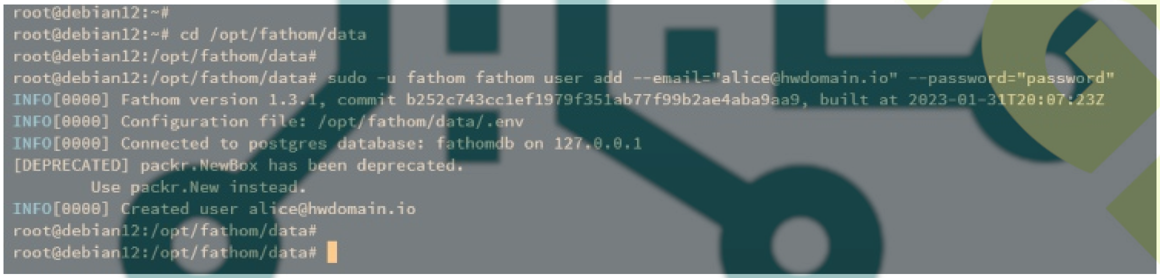
Move your current working directory to */opt/fathom/data*.

```
cd /opt/fathom/data
```

Execute the following command to create an administrator user for your Fathom installation. Be sure to change the email address and password with the following command.

```
sudo -u fathom fathom user add --email="alice@hwdomain.io" --password="password"
```

The command will use the .env file to connect to the PostgreSQL server. Once the user is created, you should get the confirmation like the following:


```
root@debian12:~#
root@debian12:~# cd /opt/fathom/data
root@debian12:/opt/fathom/data#
root@debian12:/opt/fathom/data# sudo -u fathom fathom user add --email="alice@hwdomain.io" --password="password"
INFO[0000] Fathom version 1.3.1, commit b252c743cc1ef1979f351ab77f99b2ae4aba9aa9, built at 2023-01-31T20:07:23Z
INFO[0000] Configuration file: /opt/fathom/data/.env
INFO[0000] Connected to postgres database: fathomdb on 127.0.0.1
[DEPRECATED] packr.NewBox has been deprecated.
        Use packr.New instead.
INFO[0000] Created user alice@hwdomain.io
root@debian12:/opt/fathom/data#
root@debian12:/opt/fathom/data#
```

Configuring Nginx as a Reverse Proxy

At this point, you've finished the configuration of Fathom. You will configure Nginx as a reverse proxy for your Fathom installation in the following step. Before going further, ensure that you have a domain name pointed to the server IP address.

Create a new Nginx server block configuration */etc/nginx/sites-available/fathom* using the nan editor command below.

```
sudo nano /etc/nginx/sites-available/fathom
```

Insert the following configuration and be sure to change the domain name within the **server_name** option.

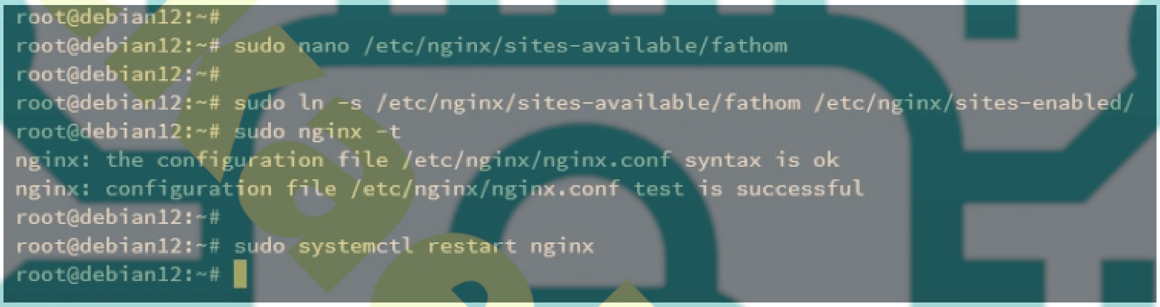
```
server {
    listen 80;
    server_name analytics.hwdomain.io;

    location / {
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $remote_addr;
        proxy_set_header Host $host;
        proxy_pass http://127.0.0.1:8080;
    }
}
```

When you're done, save and exit the file.

Now run the following command to activate the server block file /etc/nginx/sites-available/fathom and verify Nginx syntax.

```
sudo ln -s /etc/nginx/sites-available/fathom /etc/nginx/sites-enabled/
sudo nginx -t
```

If you've proper syntax, the output "**syntax is ok - test is successful**" should be printed to your terminal.



```
root@debian12:~#
root@debian12:~# sudo nano /etc/nginx/sites-available/fathom
root@debian12:~#
root@debian12:~# sudo ln -s /etc/nginx/sites-available/fathom /etc/nginx/sites-enabled/
root@debian12:~# sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
root@debian12:~#
root@debian12:~# sudo systemctl restart nginx
root@debian12:~#
```

Next, run the following command to restart the Nginx service and apply the changes that you've made. After executing the command, your Fathom installation should be accessible from your domain name.

```
sudo systemctl restart nginx
```

Lastly, run the following certbot command to secure Fathom installation with SSL/TLS certificates from Letsencrypt. Be sure to change the domain name and email address details with your information.

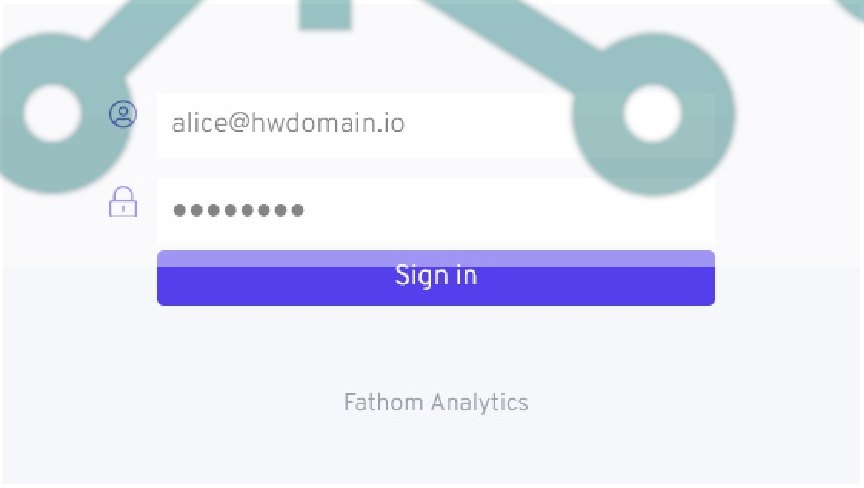
```
sudo certbot --nginx --agree-tos --redirect --hsts --staple-ocsp --email admin@hwdomain.io -d analytics.hwdomain.io
```

Once the process is finished, your SSL/TLS certificates will be available in the */etc/letsencrypt/live/domain.com* directory. Also, the Nginx server block file */etc/nginx/sites-available/fathom* is configured with HTTPS automatically via the Certbot Nginx plugin.

Accessing Fathom

Launch your preferred web browser and visit the domain name of your Fathom installation, such as http://analytics.hwdomain.io/. You should be redirected automatically to an HTTPS connection and you should get the Fathom login page.

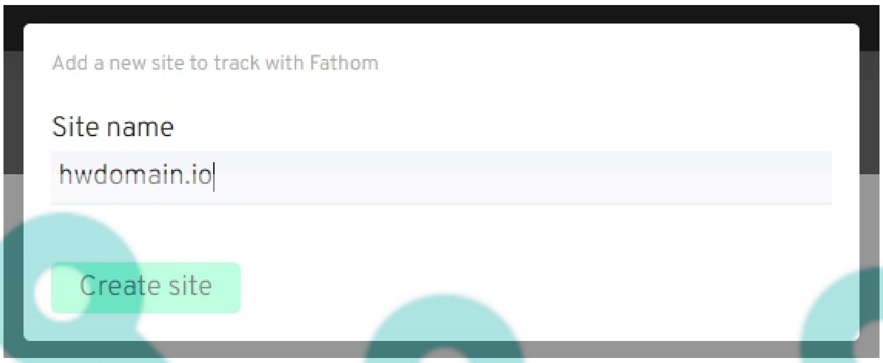
Input your admin email address and password, then click **Sign in**.



alice@hwdomain.io

••••••••

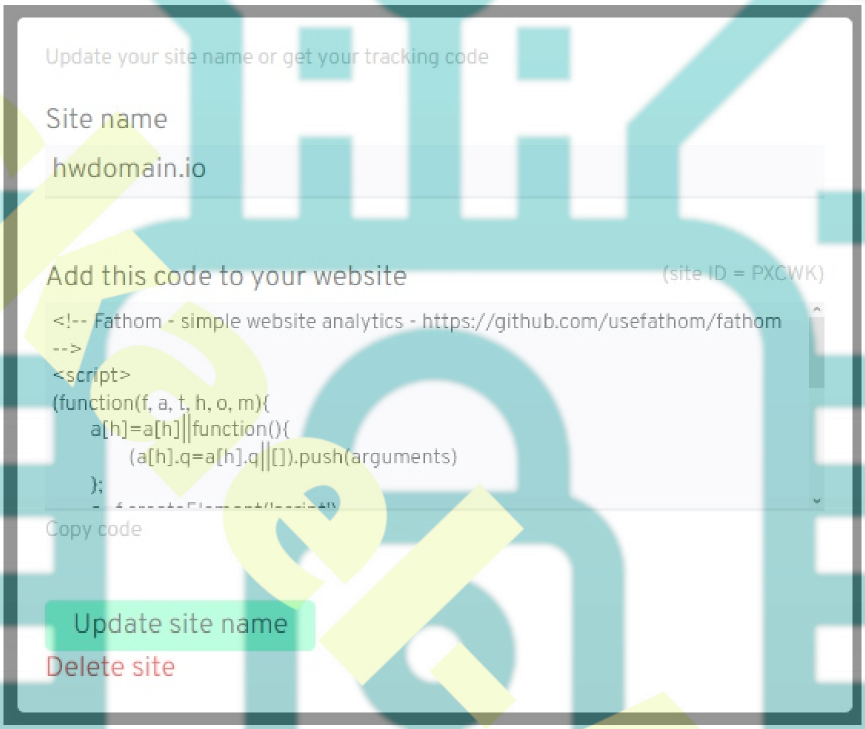
Click **Create site** to create first tracker.


Add a new site to track with Fathom

Site name

hwdomain.io

You should see the tracker code generated by Fathom.


Update your site name or get your tracking code

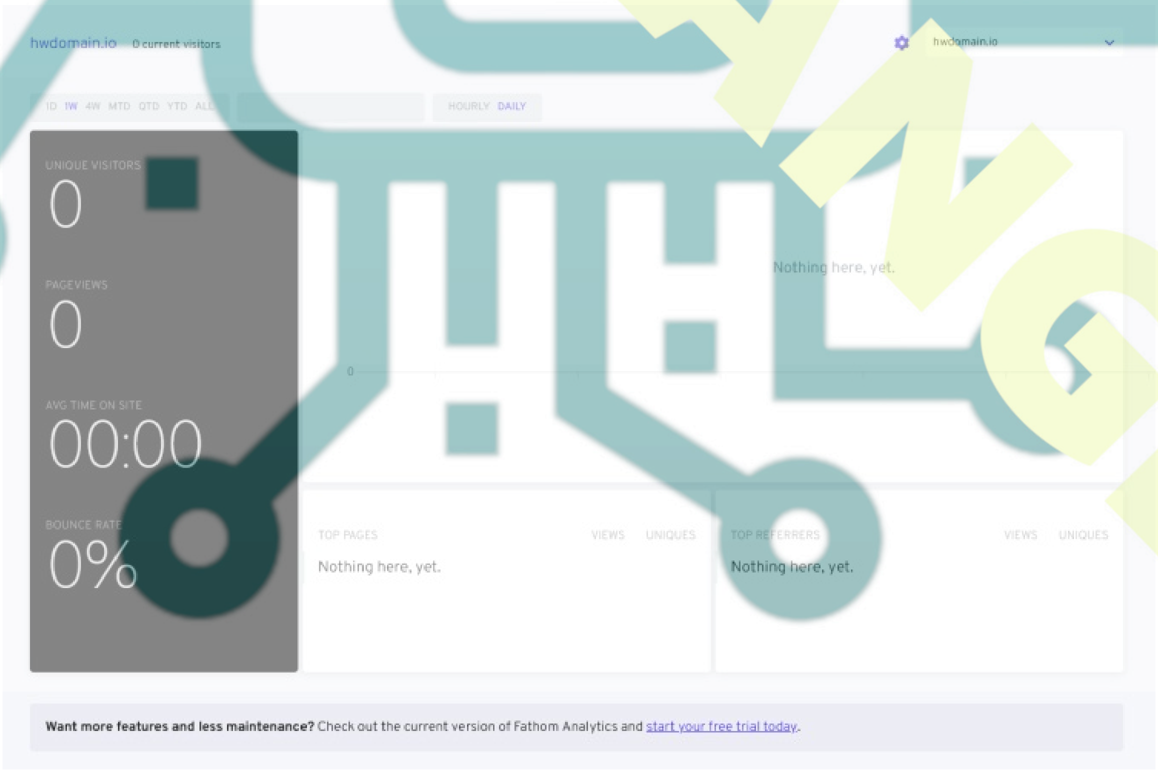
Site name

hwdomain.io

Add this code to your website (site ID = PXCWK)

```
<!-- Fathom - simple website analytics - https://github.com/usefathom/fathom
-->
<script>
(function(f, a, t, h, o, m){
    a[h]=a[h]||function(){
        (a[h].q=a[h].q||[]).push(arguments)
    };
```

Below details Fathom site analytics.


hwdomain.io 0 current visitors

UNIQUE VISITORS 0

PAGEVIEWS 0

AVG TIME ON SITE 00:00

BOUNCE RATE 0%

Nothing here, yet.

TOP PAGES VIEWS UNIQUES

Nothing here, yet.

TOP REFERRERS VIEWS UNIQUES

Nothing here, yet.

Want more features and less maintenance? Check out the current version of Fathom Analytics and start your free trial today.

From here, you can add and set new trackers for your websites.

Conclusion

To conclude, you've completed the installation of Fathom privacy-focused web analytics on the Debian 12 server with PostgreSQL database server and Nginx as a reverse proxy. You've also secured your installation with SSl/TLS certificates from Letsencrypt and configured UFW (Uncomplicated Firewalld on your Debian server. You can add a new tracker and implement it on your websites.
