How to Install Snipe-IT on Ubuntu 16.04

Posted December 12, 2017 28.4k APPLICATIONS UBUNTU 16.04

By: Namo

Introduction

In the IT industry, *asset management* is the process of tracking assets throughout their entire life cycle, including acquisition, maintenance, storage, and disposal. Although the specific assets vary, the focus is generally on individual pieces of hardware or software, licenses, and file-based resources, like digital artwork.

Snipe-IT — a free and open-source application designed specifically for IT asset management — provides a web-based interface for tracking licenses, accessories, consumables, and components. Snipe-IT includes user-based accounts with configurable group-level permissions, customizable reporting capabilities, and a JSON REST API for connecting to, managing, and extending Snipe-IT from the command line or third-party applications.

In this tutorial, you will download, install, and configure Snipe-IT and then you will create an admin user account to log into Snipe-IT for the first time.

Prerequisites

To complete this tutorial, you will need:

- One Ubuntu 16.04 server set up by following this Ubuntu 16.04 initial server setup tutorial, including a sudo non-root user and a firewall.
- The LEMP stack configured by following this LEMP installation guide.
- The following DNS records set up for your server. You can follow this hostname tutorial for details on how to add them.
    - An A record with `example.com` pointing to your server's public IP address.
    - An A record with `www.example.com` pointing to your server's public IP address.
- Nginx secured with an SSL certificate by following this setting up Let's Encrypt with Nginx server blocks on Ubuntu 16.04 tutorial. Be sure to choose option 2, `Redirect`, in Step 4 of the Nginx setup tutorial, as this

will provide automatic redirects to HTTPS on your Snipe-IT installation.

Step 1 — Preparing the Server

Before downloading Snipe-IT, prepare the server by installing some additional PHP libraries and creating the MySQL database and database user Snipe-IT will use to store its data.

Snipe-IT is built on the Laravel PHP framework and, thus, requires the Composer dependency manager for the installation and management of additional PHP libraries.

Use `apt-get` to install `composer` and `unzip`, a utility that's needed to extract files from Zip archives.

```
$ sudo apt-get install composer unzip
```

Next, install the additional PHP modules that Snipe-IT relies on.

```
$ sudo apt-get install php7.0-mbstring php7.0-xml php7.0-mcrypt php7.0-gd php7.0-zip php7.0-curl php
```

The extra packages provide PHP with:

- php7.0-mbstring - the Multibyte String module for handling languages that can't be expressed in 256 characters
- php7.0-xml - the DOM module for working with XML documents through the document object model (DOM) API, the SimpleXML module for converting XML to an object that you can manipulate with property selectors and array iterators, the WDDX module for exchanging data in the Web Distributed Data Exchange (WDDX) format, the XML Parser module for parsing XML documents, and the XSL module for performing XSLT transformations
- php7.0-mcrypt - the Mcrypt module for working with block cipher algorithms
- php7.0-gd - the GD module for image processing
- php7.0-zip - the Zip module for manipulating Zip-compressed archives
- php7.0-curl - the Client URL Library module for connecting to and communicating with servers over a variety of protocols
- php7.0-bcmath - the BCMath Arbitrary Precision Mathematics module for working with numbers of any size and precision up to 2147483647 decimals

Now, use the command-line `mysql` utility to log into MySQL as your **root** database user.

```
$ mysql -u root -p
```

Create a new MySQL user named **snipeit** on the localhost, `127.0.0.1`, and assign the user a password.

```
mysql> create user snipeit@127.0.0.1 identified by 'snipeit_user_password';
```

Next, create a database named `snipeitdb` where Snipe-IT will store its data.

```
mysql> create database snipeitdb;
```

Grant all privileges on all tables in the `snipeitdb` database to the **snipeit** user, so that Snipe-IT has permission to perform any action it needs on the database.

```
mysql> grant all on snipeitdb.* to snipeit@127.0.0.1;
```

Finally, activate the changes by reloading the grant tables with the `flush privileges` command and exit the utility.

```
mysql> flush privileges;
mysql> exit;
```

Your server now has the additional PHP libraries and MySQL database that Snipe-IT needs to function properly, so let's download and configure Snipe-IT itself.

Step 2 — Downloading and Configuring Snipe-IT

Per the official installation instructions, you'll use Git to download the latest version of Snipe-IT. Since Git only clones into existing directories if they're empty, use `ls` to view the contents of the directory you configured for Snipe-IT's Nginx server block in the Prerequisites.

```
$ ls /var/www/example.com/html/
```

If the directory isn't empty, use basic Linux navigation and file management commands to clear it out now. `mv` moves the contents to a different location, and `rm` deletes them altogether.

Once the directory is empty, download Snipe-IT from its official repository on GitHub.

```
$ git clone https://github.com/snipe/snipe-it /var/www/example.com/html/
```

The output confirms the location you're cloning into and then provides a real-time report of the process, including a count of the objects Git expected to copy as well as the number it actually did copy.

```
Output from git clone
Cloning into '/var/www/example.com/html/'...
remote: Counting objects: 70975, done.
```

```
remote: Compressing objects: 100% (62/62), done.
remote: Total 70975 (delta 20), reused 37 (delta 15), pack-reused 70897
Receiving objects: 100% (70975/70975), 67.04 MiB | 14.35 MiB/s, done.
Resolving deltas: 100% (44264/44264), done.
Checking connectivity... done.
```

You now have a complete copy of Snipe-IT, but before you begin installation, you need to enable Nginx to access the `storage`, `public/uploads`, and `bootstrap/cache` directories, as this is where Snipe-IT writes its caches, logs, and uploaded files.

Change to the installation directory.

```
$ cd /var/www/example.com/html/
```

Use `chown` with the `-R` option to recursively change the user and group ownership to `www-data` — Nginx's user and group — on all three directories.

```
$ sudo chown -R www-data:www-data storage
$ sudo chown -R www-data:www-data public/uploads
$ sudo chown -R www-data:www-data bootstrap/cache
```

Then, use `chmod` with the `-R` flag to recursively set permissions on these directories, making them read-, write-, and executable by their owner, read- and executable by their group, and read- and executable by the world.

```
$ sudo chmod -R 755 storage
$ sudo chmod -R 755 public/uploads
$ sudo chmod -R 755 bootstrap/cache
```

With the file and directory permissions correctly set for Nginx, you're ready to run `composer install`, which reads the list of additional dependencies in Snipe-IT's `composer.json` file and then resolves and installs them into `/var/www/example.com/html/vendor`.

The `--no-dev` option tells `composer` to ignore dependencies that are not necessary for running Snipe-IT but are useful when doing development on Snipe-IT.

The `--prefer-source` option tells `composer` to download the dependencies from their version control repositories, if they exist.

```
$ composer install --no-dev --prefer-source
```

The output reports each dependency that `composer` attempts to install, indicates whether the dependency was successfully cloned, and finishes by creating optimized autoload files which improve the performance of class loading in Composer-backed PHP applications.

```
Output from composer install --no-dev --prefer-source
Loading composer repositories with package information
Installing dependencies from lock file
  - Installing symfony/finder (v3.3.10)
    Cloning 773e19a491d97926f236942484cb541560ce862d
...
Generating optimized autoload files
```

You can now begin configuring your installation. Start by making a copy of the `.env.example` file that ships with Snipe-IT; this is where Snipe-IT stores environment variables and settings like timezone, base URL, and log size. Then, open `.env` for editing.

```
$ cp .env.example .env
$ nano .env
```

Look for the following:

.env

```
# ------------------------------------------
# REQUIRED: BASIC APP SETTINGS
# ------------------------------------------
...
APP_URL=null
...
```

`APP_URL` tells Snipe-IT the base URL for your installation. Replace `null` with your domain name.

.env

```
# ------------------------------------------
# REQUIRED: BASIC APP SETTINGS
# ------------------------------------------
...
APP_URL=https://example.com
...
```

Next, find the following lines:

.env

```
...
# ------------------------------------------
# REQUIRED: DATABASE SETTINGS
# ------------------------------------------
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
```

```
DB_DATABASE=null
DB_USERNAME=null
DB_PASSWORD=null
DB_PREFIX=null
DB_DUMP_PATH='/usr/bin'
DB_CHARSET=utf8mb4
DB_COLLATION=utf8mb4_unicode_ci
...
```

This is where you tell Snipe-IT how to connect to the MySQL database you created in Step 1.

Because Snipe-IT is configured by default to connect to a MySQL database running on the localhost, you don't need to modify the first two lines.

Replace `DB_DATABASE` and `DB_USERNAME` with the name of the MySQL database and database user you created in Step 1, and replace `DB_PASSWORD` with the password you assigned that database user.

`DB_PREFIX` adds custom prefixes to the table names in Snipe-IT's database. This setting is not required but may stop some automated attacks that rely on default database tables names. Leave this set to the default `null` value unless you want to add a custom prefix.

.env

```
# ------------------------------------------
# REQUIRED: DATABASE SETTINGS
# ------------------------------------------
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_DATABASE=snipeitdb
DB_USERNAME=snipeit
DB_PASSWORD=snipeit_user_password
DB_PREFIX=null
```

Close and save the file.

Now, use `artisan migrate` to populate MySQL with Snipe-IT's default database schema. This command will tell Laravel to perform a database migration using the files found in `/var/www/example.com/html/database/migrations/` .

```
$ sudo php artisan migrate
```

When prompted, enter `yes` to confirm that you want to perform the migration.

The output reports the name of each migration it completes in real time.

```
Output from php artisan migrate
```

```
**************************************
*     Application In Production!     *
**************************************

 Do you really wish to run this command? (yes/no) [no]:
 > yes

 Migration table created successfully.
 ...
 Migrated:  2017_11_08_123942_labels_display_company_name
```

Finally, use `artisan key:generate` to create an application key for your installation. Laravel will write the key's value to the `APP_KEY` line in the `.env` file, and Snipe-IT will use the key when encrypting and decrypting data like session tokens.

```
$ php artisan key:generate
```

Once again, when prompted, enter `yes` to confirm that you want to generate the application key.

When finished, the output will show you the key that was generated and tell you that the value was written to the `.env` file.

```
Output from php artisan key:generate
**************************************
*     Application In Production!     *
**************************************

 Do you really wish to run this command? (yes/no) [no]:
 > yes

Application key [base64:rxP+jS3Q8qtM9eBktXtS/zqrrXVY1LEMxoZkbV35Al0=] set successfully.
```

With installation and configuration complete, it's time to modify Nginx to serve Snipe-IT.

Step 3 — Configuring Nginx

Before you can bring Snipe-IT up in your web browser, you first need to point Nginx to Snipe-IT's root web application directory, and you need to redirect incoming requests to Snipe-IT's request handler.

Start by opening the configuration file you created for Snipe-IT's Nginx server block.

```
$ sudo nano /etc/nginx/sites-available/example.com
```

Look for the directive that sets the server block's root directory.

/etc/nginx/sites-available/example.com

```
server {
    ...
    root /var/www/example.com/html;
    ...
}
```

Snipe-IT's web application files are located in the `public` directory that was automatically created when you cloned the project from GitHub. Modify Nginx to use `public` as this server block's root directory.

/etc/nginx/sites-available/example.com

```
server {
    ...
    root /var/www/example.com/html/public;
    ...
}
```

Next, find the default location block:

/etc/nginx/sites-enabled/snipe-it

```
server {
    ...
    location / {
        try_files $uri $uri/ =404;
    }
    ...
}
```

Modify this block to pass all requests to Snipe-IT's request handler for processing.

/etc/nginx/sites-enabled/snipe-it

```
server {
    ...
    location / {
        try_files $uri $uri/ /index.php$is_args$args;
    }
    ...
}
```

Save and close the file.

Before restarting Nginx, test your new configuration.

```
$ sudo nginx -t
```

The output should report that your `syntax is ok`. If it doesn't, follow the on-screen messages for additional help.

Now, restart Nginx to apply the changes.

```
$ sudo systemctl reload nginx
```

Finally, verify that Nginx is back up and running.

```
$ sudo systemctl status nginx
```

The output should indicate that the service is `active (running)`. If it doesn't, retrace the previous steps to resolve the problem before continuing

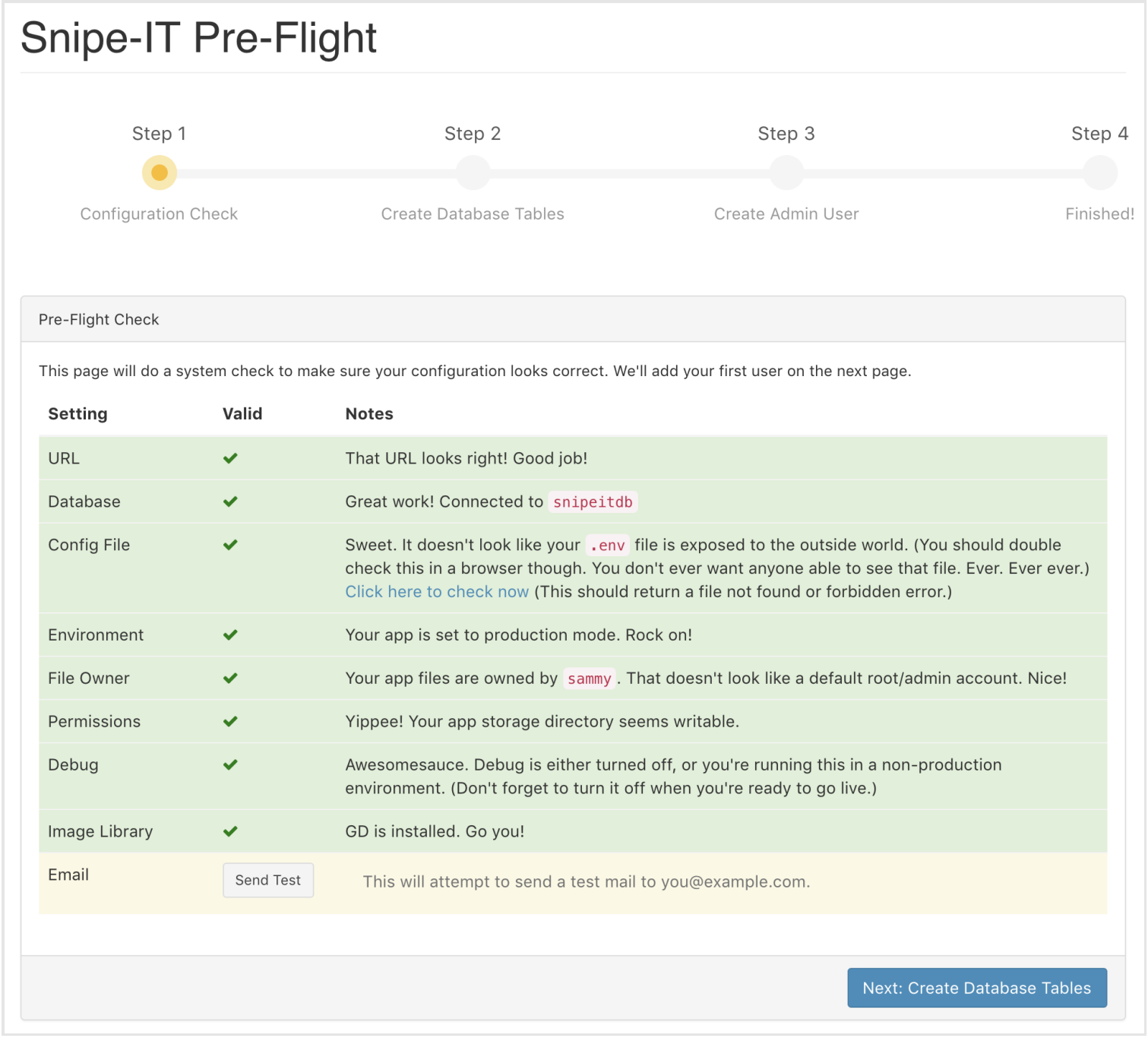
Now that Nginx is fully configured, log into Snipe-IT's web setup utility to complete the installation.

Step 4 — Setting up Snipe-IT with the Pre-Flight Utility

To finish the installation, point your web browser to `https://example.com`. This will take you to **Step 1** of Snipe-IT's **Pre-Flight Utility**, where Snipe-IT will do a quick test of your installation to make sure that everything is correctly configured.

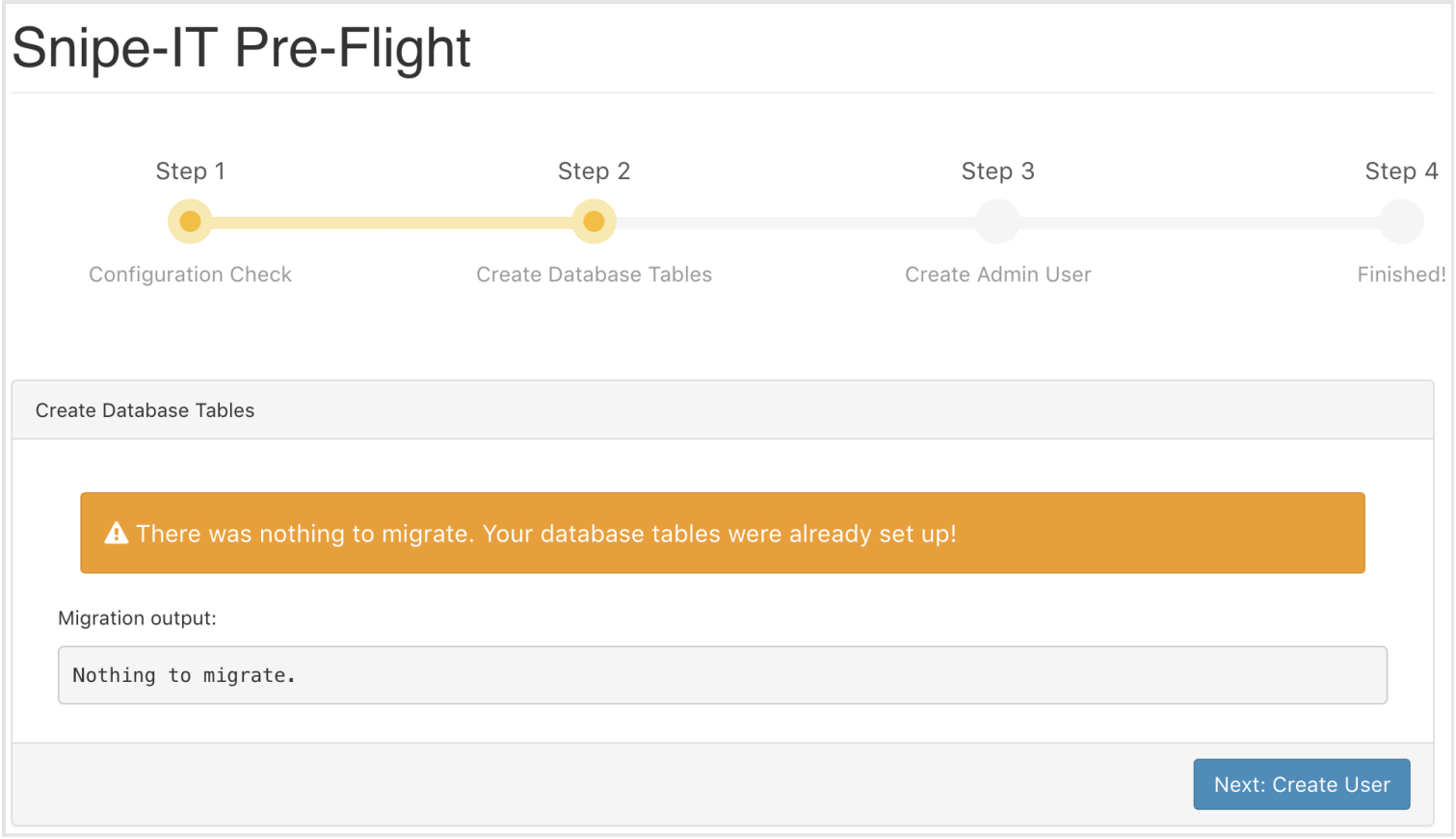
On this screen, you'll see a table showing you each setting that **Pre-Flight** tested, the setting's test result, and a short note describing the setting. A green checkmark in the **Valid** column indicates the setting was correct. If any setting is highlighted in pink and marked with a red **X** in the **Valid** column, that indicates there's a problem with that setting. Follow Snipe-IT's instructions for resolving the problem before continuing.

As we haven't configured Snipe-IT for email, you can click the blue **Next: Create Database Tables** button in the bottom, right-hand corner of the screen to continue the installation now.



In **Step 2** of **Pre-Flight**, Snipe-IT checks your database and performs a migration if necessary. Since you already did a manual database migration with `artisan` in Step 3 of this tutorial, **Pre-Flight** will tell you that the database is **already set up** and that there is **Nothing to migrate**.

Press the blue **Next: Create User** button in the bottom, right-hand corner of the screen.



In **Step 3** of **Pre-Flight**, Snipe-IT asks you to enter some general application settings and create your first administrative user account.

In the **Site Name** field, enter the label you want Snipe-IT to display at the top of every screen. This could be your company's name or it could even be something more descriptive like, **Sammy's Asset Management**.

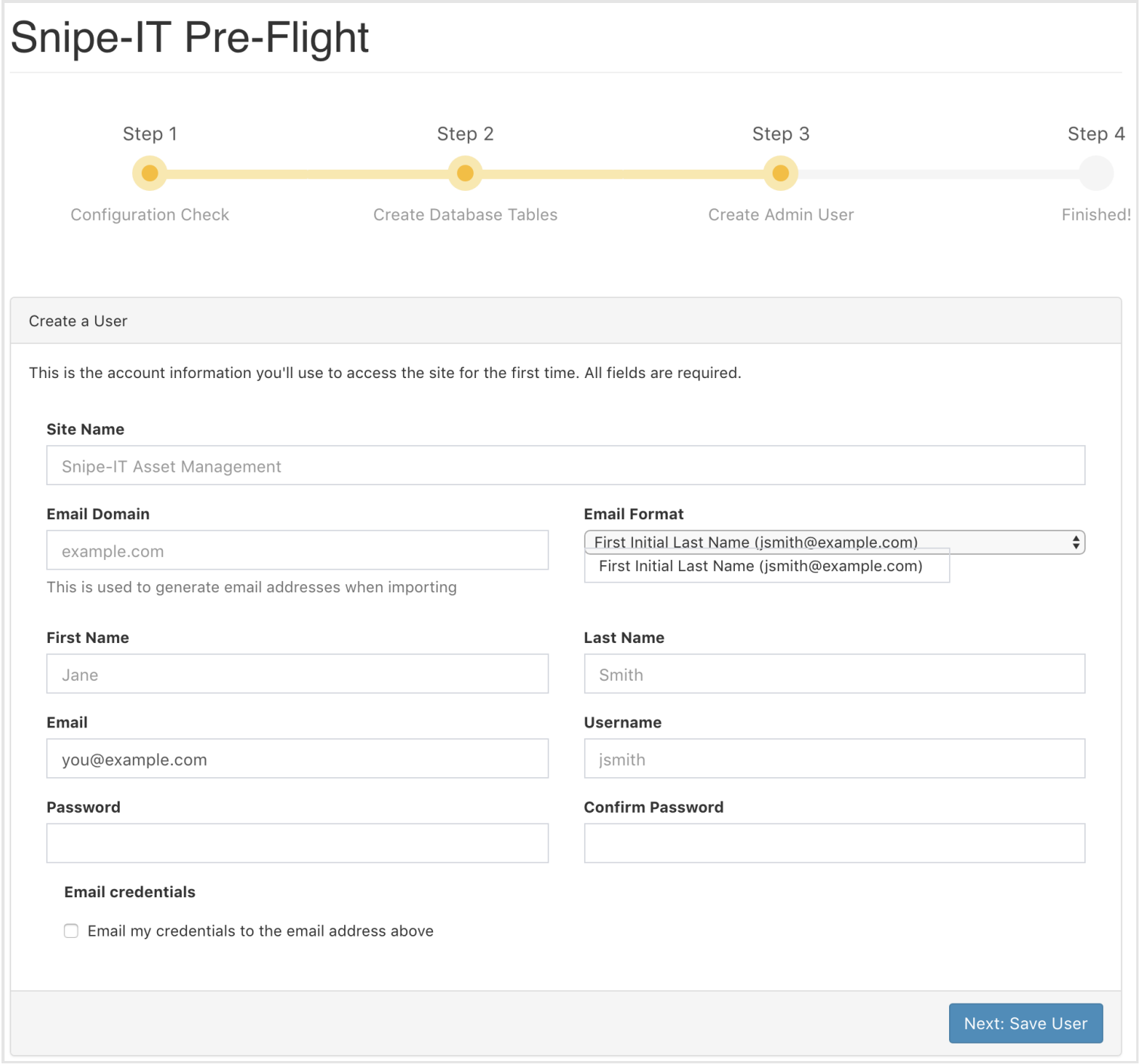
In the **Email Domain** field, enter the domain you want Snipe-IT to use for outgoing mail, and in the **Email Format** field, select the way you want Snipe-IT to format the **To:** header in outgoing messages.

Enter your name in the **First Name** and **Last Name** fields and your email address in the **Email** field.

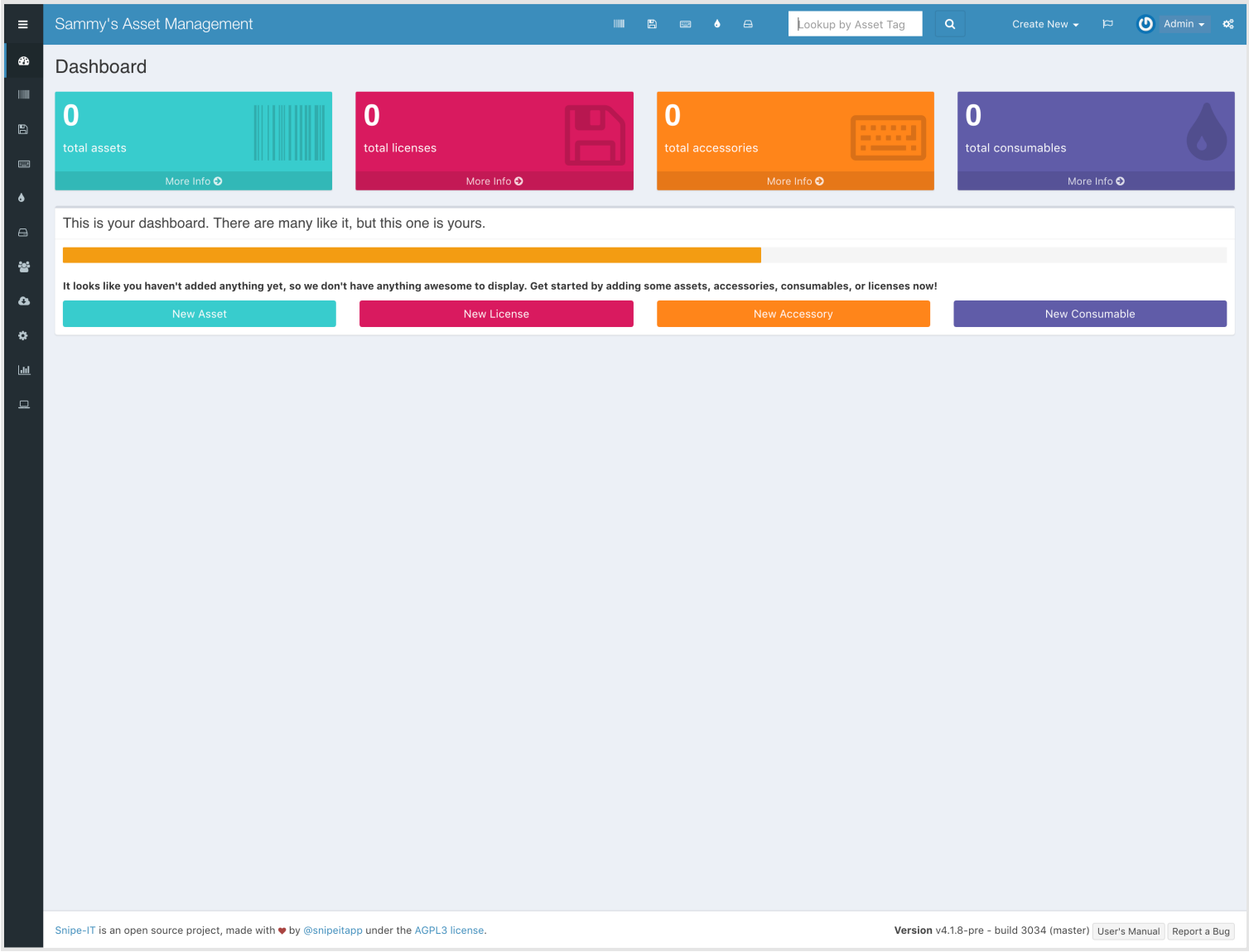
Finally, enter the username you'd like associated with your account in the **Username** field, and enter the password you'd like to use in the **Password** field. Be sure to enter the same password in the **Confirm Password** field and make a note of your credentials before continuing. You'll need them both to log into Snipe-IT.

Because you're creating this account for yourself, you can leave **Email my credentials to the email address above** unchecked.

Click the blue **Next: Save User** button in the bottom, right-hand corner of the screen once you've filled out all of the information.



In **Step 4** of **Pre-Flight**, Snipe-IT saves the general application settings you just entered, creates the new administrative user, and logs you into the main dashboard.



At this point, your installation is complete and you can start using Snipe-IT to manage your or your clients' IT assets.

Conclusion

In this article you set up the LEMP stack, secured Nginx with a Let's Encrypt TLS/SSL certificate, installed and configured Snipe-IT, created an administrative user account, and logged into the main Snipe-IT dashboard.

To learn about adding and editing assets to Snipe-IT, see the official guide to managing assets.

To learn about working with user accounts in Snipe-IT, see the official documentation on managing users.

Or, for other questions, check out the official Snipe-IT User's Manual.

Editor:
Dave Rankin

Related Tutorials

How to Install and Configure Magento on Ubuntu 14.04

How To Install YunoHost on Debian 9

How To Display Data from the DigitalOcean API with React

How To Build a Node.js Application with Docker

How To Install and Set Up Manifold Scholar

1 Comment

Leave a comment...

Log In to Comment

dsangvikar *January 31, 2018*

0 I completed step 3. But on going to the subdomain I configured for snipeit, I get a 404 everytime. Has something changed?