Getting Started - AlmaLinux

Requirements

Ticaga requires the latest version of PHP, this is because we pride ourselves on being up-to date with security updates, functions and cool new features.

Minimum requirements:

- PHP: **8.4**
- MariaDB: **10.5.22** // MYSQL: **8.0**
- Ioncube loaders: **13.3.1**
- Composer
- Laravel 11.x
- Lets Encrypt for SSL

Environment:

We recommend AlmaLinux 9 on a Virtual Private server with at least 2GB RAM.

Update the server & install Apache

To run Ticaga, we need to update the server:

```
sudo dnf update -y && sudo dnf upgrade -y
```

After we've updated and got the latest versions of the repositories, we can install Apache:

```
dnf install httpd nano zip unzip wget php-pdo_mysql php-pdo npm -y
```

Now we've installed Apache, we need to start it on server boot:

```
sudo systemctl enable httpd && sudo systemctl start httpd
```

If you have a firewall built in on the server, you'll need to allow Apache threw the firewall (optional):

```
sudo firewall-cmd --zone=public --add-service=http --permanent
sudo firewall-cmd --zone=public --add-service=https --permanent
sudo firewall-cmd --reload
```

If you don't have an firewall installed the commands will fail, you can ignore them and continue the documentation or install a firewall.

Install PHP

Now we've got Apache sorted, we can install PHP, we recommend the Remi Repo for this, however you can use any repository you prefer:

```
dnf install https://rpms.remirepo.net/enterprise/remi-release-9.rpm -y
dnf module switch-to php:remi-8.4/common -y
```

Finally to finish PHP off we need to install the PHP modules required for Ticaga:

```
sudo dnf install php php-cli php-fpm php-curl php-mysqlnd php-gd php-opcache php-zip php-intl php-common
php-bcmath php-json php-readline php-mbstring php-apcu php-xml php-dom php-mcrypt php-mailparse -y
```

Database

Now we've got PHP sorted, we can now sort out the database, this is where all your Ticaga tickets, departments, customers, agents will be created, edited and removed.

Let's get started:

```
yum install mariadb-server -y
```

Now we need to enable MariaDB on the server boot, just like Apache:

```
sudo systemctl enable mariadb && sudo systemctl start mariadb
```

Now we need to get everything sorted, so let's do:

```
mysql_secure_installation
```

This loads the installation for MariaDB, find out more and the questions here.

Time to create the database and user for Ticaga:

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

Enter your root password, or the new password you created in the last step.

```
CREATE DATABASE ticaga;
CREATE USER 'ticaga_user'@'localhost' IDENTIFIED BY 'p@s3w0r$2024!';
GRANT ALL PRIVILEGES ON ticaga.* TO 'ticaga_user'@'localhost';
FLUSH PRIVILEGES;
exit;
```

ticaga = MySQL Database Name

ticaga_user = MySQL Username

p@s3w0r$2024! = MySQL Password (Please **don't use** this)

ticaga.* = Database name and wildcard. (You **need** the **.***)

ticaga_user = MySQL Username

Congratulations you've now got the database ready.

Virtual Host

Okay, I'm sorry we've just called this Virtual Host, however that is exactly what it is, however to normal humans like ourselves, this is the configuration file which allows our domain (hostname) to show Ticaga.

You need to create a DNS Record for the subdomain (**support.ticaga.com**) if you aren't using your own domain for Ticaga (**ticaga.com**). We recommend a Subdomain for Ticaga to keep it away from your main website just for security, we recommend the same for a billing system like Blesta, WHMCS or ClientExec, keep them separate from Ticaga and your website.

These folders don't come by default in an operating system but we like them because of usability:

```
sudo mkdir /etc/httpd/sites-available /etc/httpd/sites-enabled
```

Now we've made the folders, lets include the configurations in the sites-enabled folder:

```
echo "IncludeOptional sites-enabled/*.conf" >> /etc/httpd/conf/httpd.conf
```

What this does is allows Apache to read our configuration file in sites-enabled, we don't read the available as they aren't all enabled.

Let's make our ticaga configuration, in this example we'll use **demo.ticaga.com**:

```
sudo nano /etc/httpd/sites-available/demo.ticaga.com.conf
```

Time to paste in the configuration and edit as you require:

```
<VirtualHost *:80>
    ServerName demo.ticaga.com
    DocumentRoot /var/www/ticaga/public
    <Directory /var/www/ticaga/public>
        Options FollowSymlinks
        AllowOverride All
        Require all granted
        <Files .env>
            Order allow,deny
            Deny from all
```

```
        </Files>
    </Directory>
    ErrorLog /var/log/httpd/demo.ticaga.com_error.log
    CustomLog /var/log/httpd/demo.ticaga.com.log combined
</VirtualHost>
```

You **need** to keep the **/public** folder because if you don't you'll **expose** all the Laravel files in the Ticaga installation.

Now we can symlink the new configuration to the enabled folder to enable your configuration.

```
sudo ln -s /etc/httpd/sites-available/demo.ticaga.com.conf /etc/httpd/sites-enabled/demo.ticaga.com.conf
```

Install Composer

We need to install Composer as it's a dependency manager for PHP and we use it for Laravel.

```
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"
php -r "if (hash_file('sha384', 'composer-setup.php') ===
'dac665fdc30fdd8ec78b38b9800061b4150413ff2e3b6f88543c636f7cd84f6db9189d43a81e5503cda447da73c7e
5b6') { echo 'Installer verified'; } else { echo 'Installer corrupt'; unlink('composer-setup.php'); } echo PHP_EOL;"
php composer-setup.php
php -r "unlink('composer-setup.php');"

sudo mv composer.phar /usr/local/bin/composer
```

Then you need to run:

```
dnf install composer -y
```

This will install composer to your server.

Let's Encrypt SSL

Now we need to install Let's Encrypt, this is a way to get a free SSL Certificate for our Ticaga installation.

This allows you to use **https://demo.ticaga.com** instead of **http://demo.ticaga.com**.

```
dnf install certbot python3-certbot-apache mod_ssl -y
certbot --apache
```

If you're using **Nginx** you will need to change **--apache** to **--nginx**. We recommend Apache as it's easier to maintain.

Ioncube Loaders

Ioncube loaders are required for Ticaga as we encode **8** files for licensing purposes.

Without this your Ticaga installation will not work.

```
sudo wget -N https://downloads.ioncube.com/loader_downloads/ioncube_loaders_lin_x86-64.zip
sudo unzip ioncube_loaders_lin_x86-64.zip
sudo mv ./ioncube/ioncube_loader_lin_8.4.so /usr/lib64/php/modules/ioncube_loader_lin_8.4.so
sudo nano /etc/php.ini
```

Paste this near the top:

```
zend_extension = /usr/lib64/php/modules/ioncube_loader_lin_8.4.so
```

Remove the un-needed files for Ioncube:

```
sudo rm -rf ./ioncube
sudo rm -rf ./ioncube_loaders_lin_x86-64.zip
```

Add a new SSH User

Laravel doesn't like it when you run it in root, this is for security purposes. We agree with them it's safer, however you can use root.

```
sudo useradd -m -d /home/ticaga demo
```

This creates a user called "demo".

```
sudo passwd demo
```

This allows you to create a password for the username "demo".

```
sudo usermod -aG wheel demo
```

This allows you to add the user "demo" to the sudo group, **this allows you to use sudo**.

Install Ticaga

Let's log into the new user we created earlier:

```
su demo
```

Let's give our **www** folder the correct permissions:

```
sudo chown -R $USER /var/www/
```

Before we continue let's disable **SELinux**:

```
sudo sed -c -i "s/\SELINUX=.*/SELINUX=disabled/" /etc/selinux/config
```

And Selinux rules incase it's playing up:

```
semanage fcontext -a -t httpd_sys_rw_content_t "/var/www/ticaga/storage(/.*)?"
semanage fcontext -a -t httpd_sys_rw_content_t "/var/www/ticaga/bootstrap/cache(/.*)?"
restorecon -Rv /var/www/ticaga
sudo setsebool -P httpd_can_network_connect_db 1
```

Let's go to the www folder:

```
cd /var/www/
```

Let's create the Laravel project which we will use for Ticaga:

```
composer create-project --prefer-dist laravel/laravel ticaga
```

Now let's go into the ticaga folder:

```
cd ticaga
```

Let's remove all the default files from Laravel:

```
sudo rm -rf ./*
```

THIS IS AN IMPORTANT STEP AND DANGEROUS, ENSURE THE ./ IS BEFORE THE *, ELSE YOU WILL DELETE THE WHOLE SERVER!

If you are worried about breaking your server, please contact our support team or open a sales ticket if you have a trial license.

Let's get the latest version of Ticaga:

```
wget -N https://ticaga.com/latest.zip
```

Unzip the files:

```
unzip latest.zip
```

Let's provide the correct permissions for Ticaga:

```
sudo chown -R apache: ./*
```

Now we've done everything we need, it's time to edit the .env file for Ticaga.

```
sudo mv .env.example .env
```

and let's edit the newly created .env:

```
sudo nano .env
```

or you can use vi to edit:

```
sudo vi .env
```

You can now edit the main settings:

```
APP_NAME="Ticaga Demo"
```

This is your company name.

```
APP_URL=https://demo.ticaga.com
```

This is your URL to your Ticaga, sub-domain or main domain if you prefer.

```
DEFAULT_DEPARTMENT=support
```

This is your default department for tickets if the customer selects "Create ticket".

If you leave this blank it will disable the "Create ticket" button.

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=ticaga
DB_USERNAME=ticaga_user
DB_PASSWORD=p@s3w0r$2024!
```

These are the database name, username and password you created when doing the Database.

Let's provide the permissions for the bootstrap and storage folders:

```
sudo chmod -R ugo+rw bootstrap storage
```

Now we need to run the composer command to create the vendors folder for Ticaga, without this Ticaga won't work. We don't ship the vendors folder as it is huge:

```
composer install
```

Now we need to generate a secure App Key:

```
sudo php artisan key:generate
```

Let's sort the storage link out so you can upload avatars:

```
sudo php artisan storage:link
```

Let's create the database tables for Ticaga:

```
sudo php artisan migrate
```

Now we need to Seed the database with content pre-filled for Ticaga:

```
php artisan db:seed
```

Now let's install Vite (this ensures nothing goes wrong during set-up):

```
sudo npm install
sudo npm install -g vite
```

If you have any problems run:

```
sudo rm -rf node_modules/
```

If you see this:

```
added 153 packages, and audited 154 packages in 8s

37 packages are looking for funding
  run `npm fund` for details

6 vulnerabilities (3 moderate, 3 high)

To address all issues, run:
  npm audit fix

Run `npm audit` for details.
```

Run the following:

```
sudo dnf module install nodejs:22 -y
sudo npm install -g npm@11.2.0
sudo npm audit fix
```

Publish Livewire Assets:

```
php artisan livewire:publish --assets
```

Need to replace the license management page:

```
sudo mv ./hot_fix/vendor/laravel/framework/src/Illuminate/View/Compilers/BladeCompiler.php
./vendor/laravel/framework/src/Illuminate/View/Compilers/BladeCompiler.php
```

Lets now build Ticaga:

```
sudo npm run build
```

Clear Ticaga's caches:

```
php artisan optimize:clear
```

Reboot the server to kick Ioncube into action:

```
sudo reboot
```

Log back into the server and go to Ticaga:

```
cd /var/www/ticaga
```

Log into the username:

```
su demo
```

Load up Ticaga:

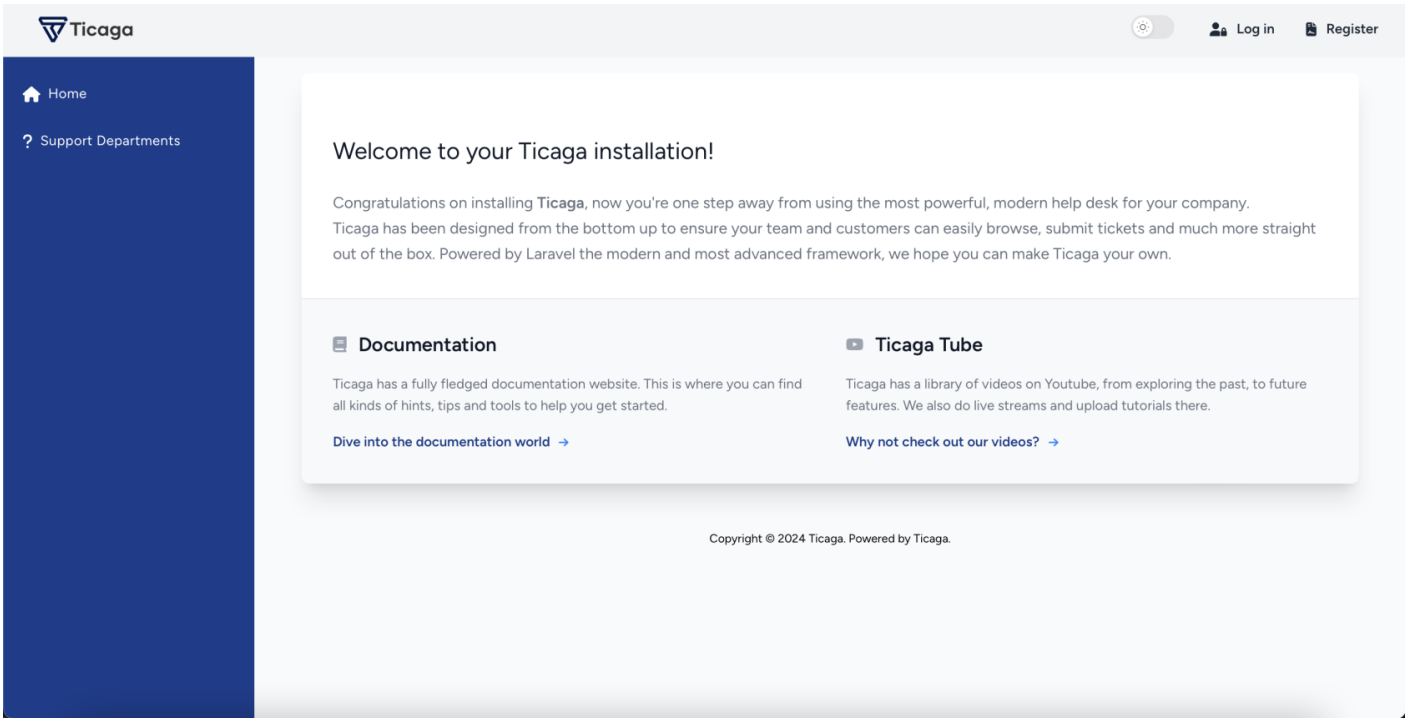
```
php artisan up
```

That's it now you have a fully working Ticaga.

Create an Agent

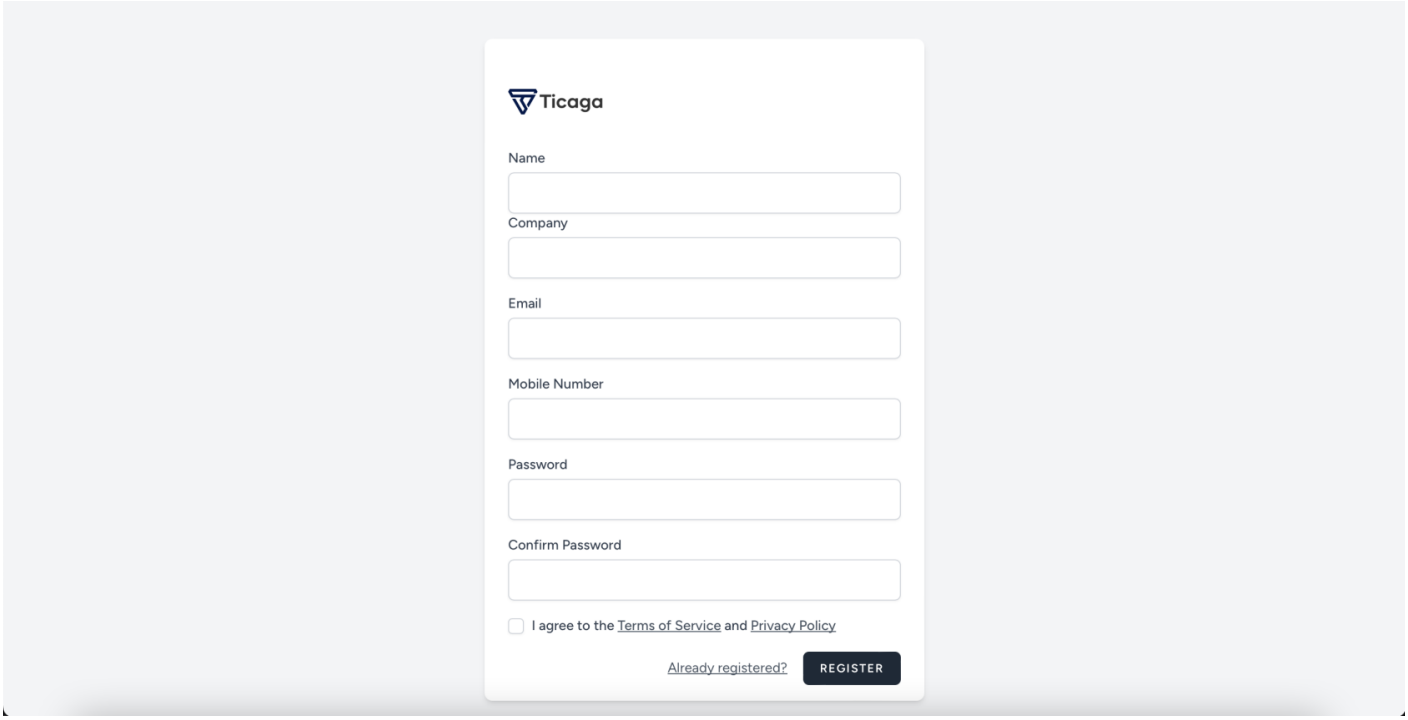
Now you've got your Ticaga up and running go to your installation:

eg: **https://demo.ticaga.com**



Now click on **Register**

Fill in the form, click **Register**.



Voila! You're the super admin of your Ticaga installation.