ANGKORCMS

DOCUMENTATION

LUCAS BURDEYRON

Contents

Introduction

About AngkorCMS

A content management system (CMS is a computer application that allows publishing, editing and modifying content, organizing, deleting as well as maintenance from a central interface. Such systems of content management provide procedures to manage workflow in a collaborative environment. These procedures can be manual steps or an automated cascade. CMSs have been available since the late 1990s.

CMSs are often used to run websites containing blogs, news, and shopping. Many corporate and marketing websites use CMSs. CMSs typically aim to avoid the need for hand coding, but may support it for specific elements or entire pages.

(http://en.wikipedia.org/wiki/Content_management_system)

AngkorCMS is a CMS build with the framework Laravel 5.

Laravel is a web application framework with expressive, elegant syntax. We believe development must be an enjoyable, creative experience to be truly fulfilling. Laravel attempts to take the pain out of development by easing common tasks used in the majority of web projects, such as authentication, routing, sessions, queueing, and caching.

Laravel is accessible, yet powerful, providing powerful tools needed for large, robust applications. A superb inversion of control container, expressive migration system, and tightly integrated unit testing support give you the tools you need to build any application with which you are tasked.

http://www.laravel.com

What does AngkorCMS do?

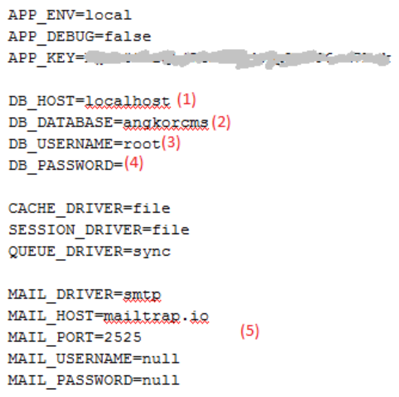
AngkorCMS can manage:

- Multi languages
- Templates and themes
- Users management
- Medias
- Plugins (Interactive Maps, Slideshow, Text Content…) (Open to development)

Installation

The .env file:

All the information needed to configure AngkorCMS.



```
APP_ENV=local
APP_DEBUG=false
APP_KEY=[redacted]

DB_HOST=localhost (1)
DB_DATABASE=angkorcms (2)
DB_USERNAME=root(3)
DB_PASSWORD=(4)

CACHE_DRIVER=file
SESSION_DRIVER=file
QUEUE_DRIVER=sync

MAIL_DRIVER=smtp
MAIL_HOST=mailtrap.io
MAIL_PORT=2525            (5)
MAIL_USERNAME=null
MAIL_PASSWORD=null
```

1. The link to the database
2. Name of the database
3. Username to access the database
4. Password to access the database
5. Info to access mail functionality

Then when the connection to database is operational you can install the tables using this command:

```
php artisan angkorcms:install
```

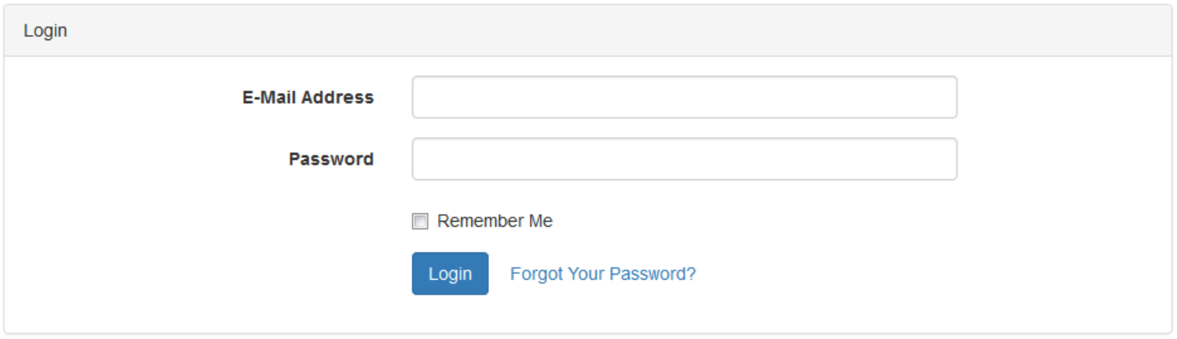
Usage

Back Office

The back-office is reachable to the URL '/admin'.

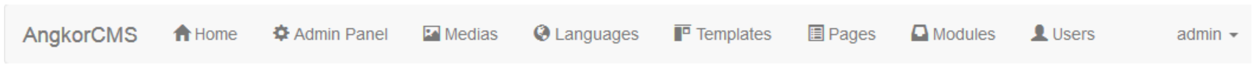
By default the login is "admin@angkorcms.com" and the password is "password".

If you're not logged in, it will display the login screen.



Login screen

When you'll be logged in, it will display the admin panel where you can manage the website using the navbar.

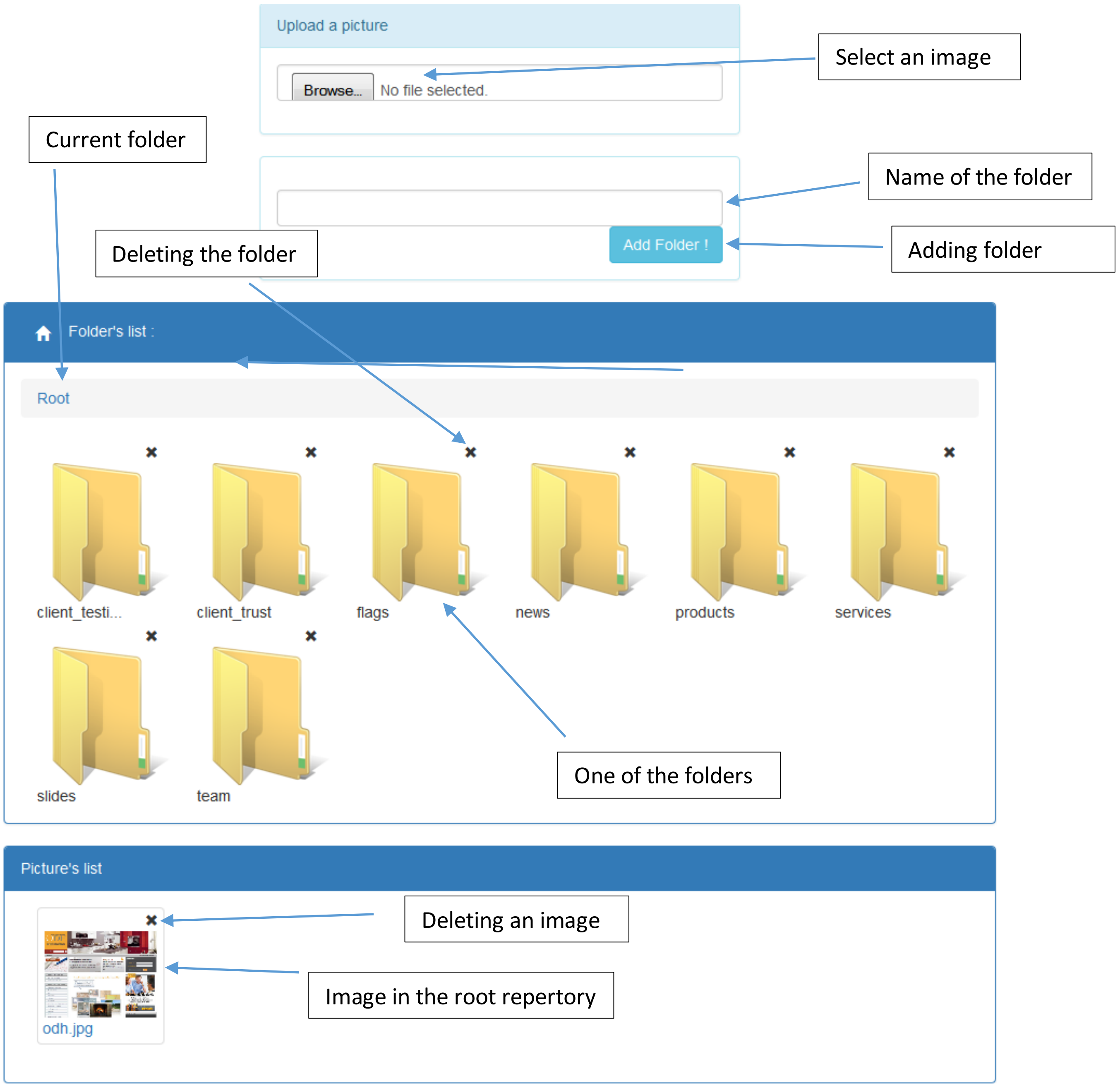


Navigation bar

Medias

The images are used by some other objects like languages or slideshows.

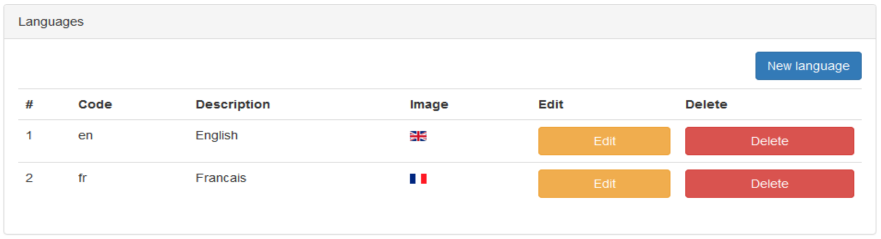
You can manage the media in media's menu. You can upload a picture in the current folder by selecting it. You can browse folders by clicking on them. And you can add a folder by writing the name of the folder and click on the button add folder.



Languages

The languages are used to translate pages or modules.

You can add, edit or delete a language in the language's menu.



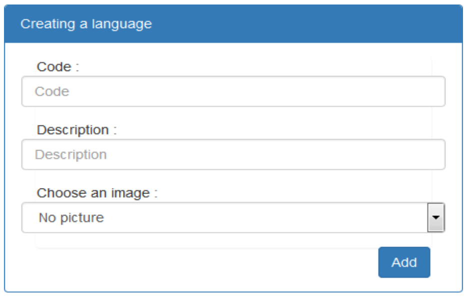
Languages

New language

#	Code	Description	Image	Edit	Delete
1	en	English	🇬🇧	Edit	Delete
2	fr	Francais	🇫🇷	Edit	Delete

Languages index

You can add a language by clicking the button "New language". You have to set a code corresponding to the standard ISO 639-1.



Creating a language

Code : Code

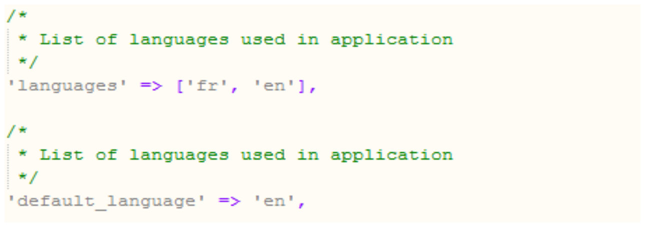
Description : Description

Choose an image : No picture

Add

Creating a language

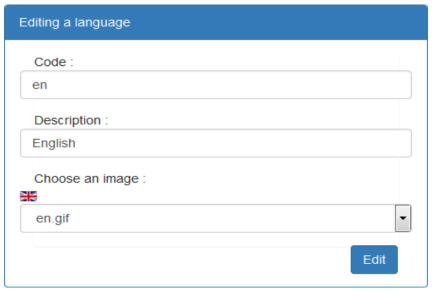
When you add a languages think to add it in the config file ('config/angkorcmsmultilanguages.php')



```
/*
 * List of languages used in application
 */
'languages' => ['fr', 'en'],

/*
 * List of languages used in application
 */
'default_language' => 'en',
```

Adding the language in the config file

You can edit the language by clicking an "Edit" button.



Editing a language

Code : en

Description : English

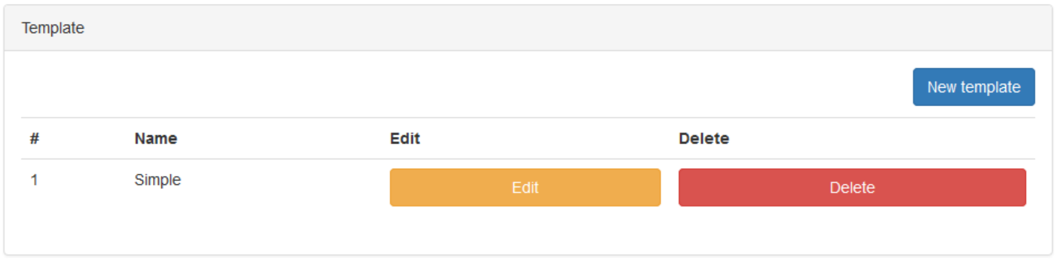
Choose an image : 🇬🇧 en.gif

Edit

Editing a language

Templates/Themes

Each templates contains a list of "blocks" and each themes available in the template can use those blocks in their view.

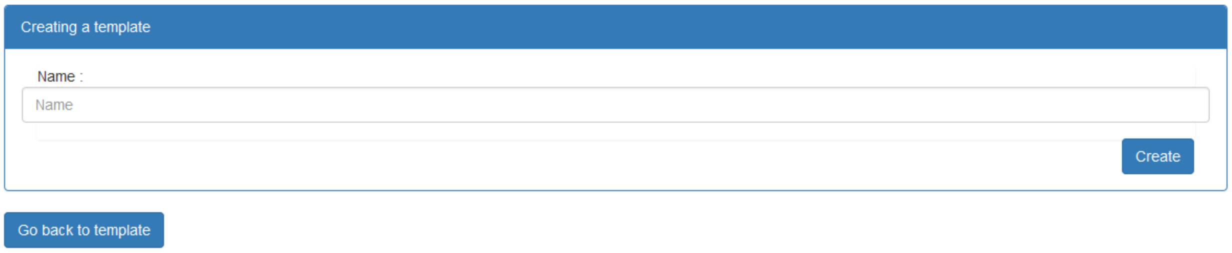


Template

New template

#	Name	Edit	Delete
1	Simple	Edit	Delete

Template index



Creating a template

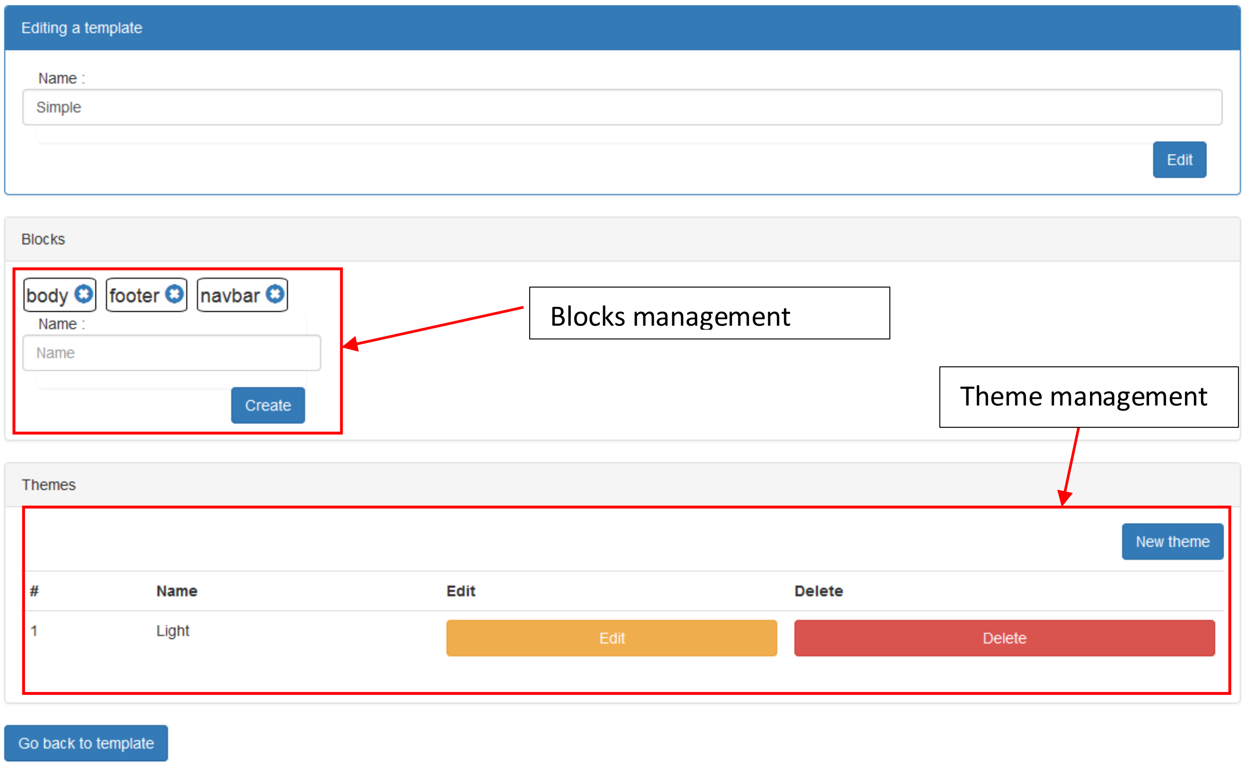
Name :

Name

Create

Go back to template

Create Template



Editing a template

Name :

Simple

Edit

Blocks

body footer navbar

Name :

Name

Create

Blocks management

Theme management

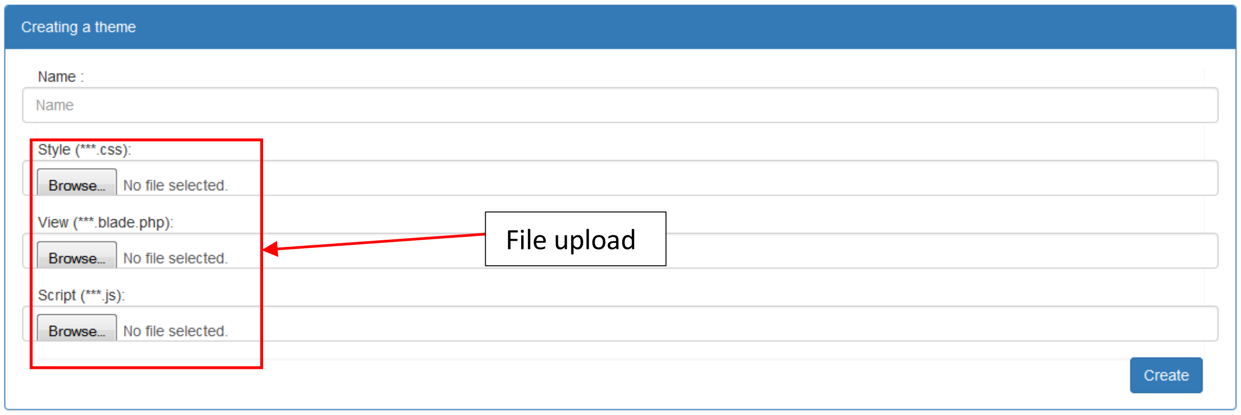
Themes

New theme

#	Name	Edit	Delete
1	Light	Edit	Delete

Go back to template

Edit template


Creating a theme

Name :

Name

Style (***.css):

Browse... No file selected.

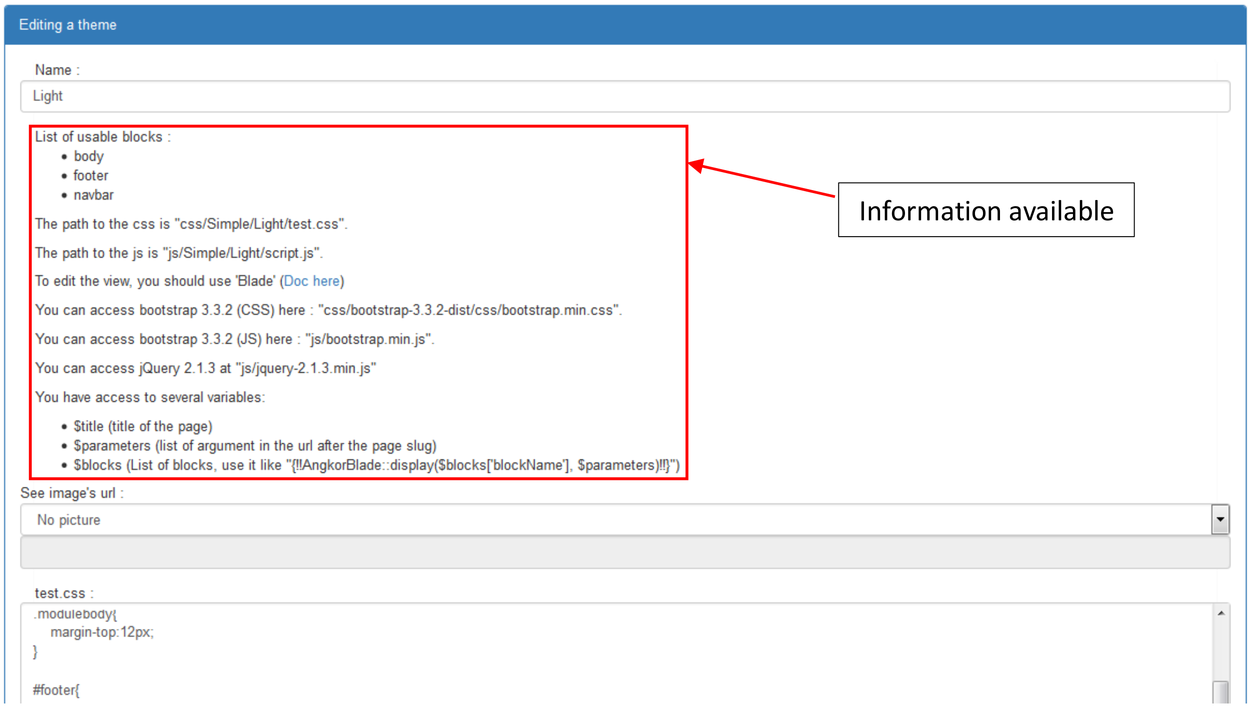
View (***.blade.php):

Browse... No file selected.

Script (***.js):

Browse... No file selected.

Theme create


Editing a theme

Name :

Light

List of usable blocks :

- body
- footer
- navbar

The path to the css is "css/Simple/Light/test.css".

The path to the js is "js/Simple/Light/script.js".

To edit the view, you should use 'Blade' (Doc here)

You can access bootstrap 3.3.2 (CSS) here : "css/bootstrap-3.3.2-dist/css/bootstrap.min.css".

You can access bootstrap 3.3.2 (JS) here : "js/bootstrap.min.js".

You can access jQuery 2.1.3 at "js/jquery-2.1.3.min.js"

You have access to several variables:

- $title (title of the page)
- $parameters (list of argument in the url after the page slug)
- $blocks (List of blocks, use it like "{!!AngkorBlade::display($blocks['blockName'], $parameters)!!}")

See image's url :

No picture

test.css :

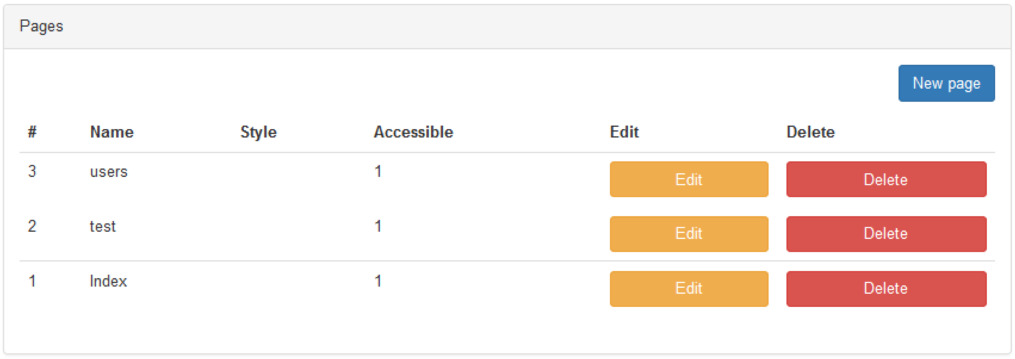
```
.modulebody{
    margin-top:12px;
}

#footer{
```

Theme edit

Pages

Each pages contains a list of translations according to the languages defined in the multi-languages plugin.

Each translation have a slug to access it. As '/{slug}'

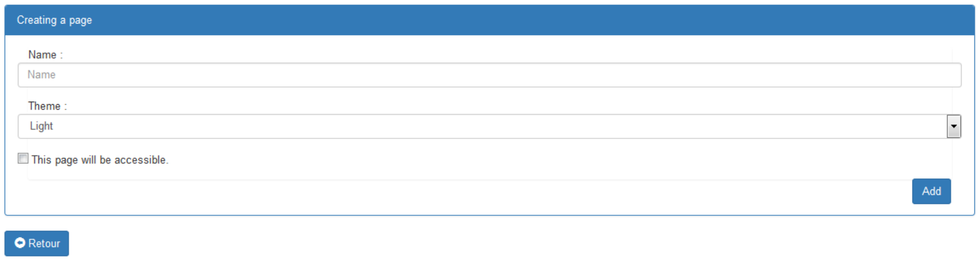


Pages

New page

#	Name	Style	Accessible	Edit	Delete
3	users		1	Edit	Delete
2	test		1	Edit	Delete
1	Index		1	Edit	Delete

Pages index



Creating a page

Name : Name

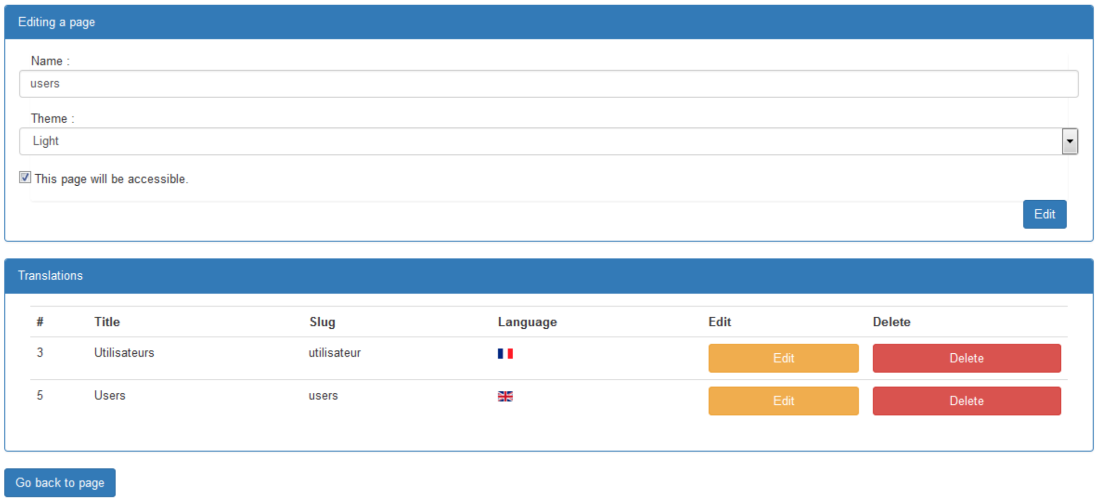
Theme : Light

☐ This page will be accessible.

Add

Retour

Create page



Editing a page

Name : users

Theme : Light

☑ This page will be accessible.

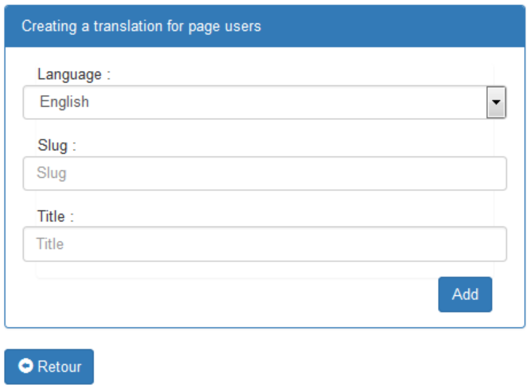
Edit

Translations

#	Title	Slug	Language	Edit	Delete
3	Utilisateurs	utilisateur	🇫🇷	Edit	Delete
5	Users	users	🇬🇧	Edit	Delete

Go back to page

Edit Page



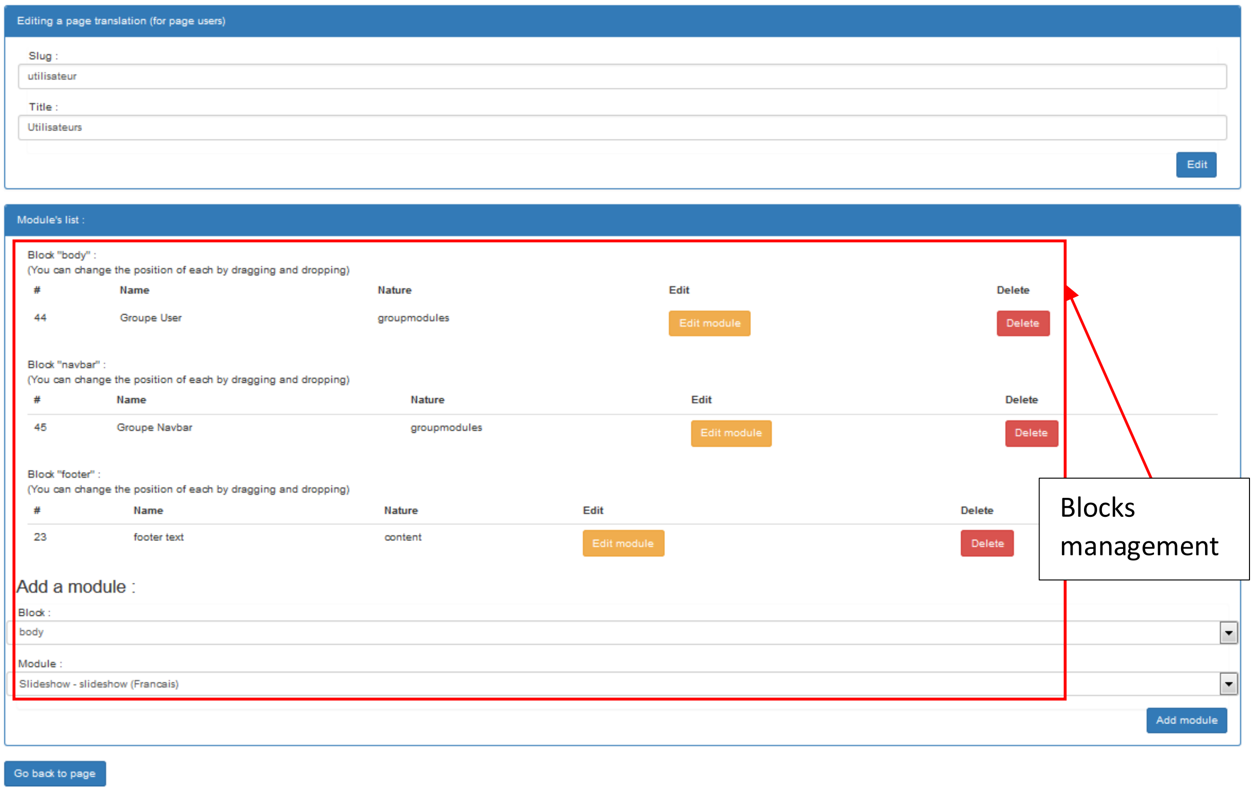
Creating a translation for page users

Language : English

Slug : Slug

Title : Title

Create Page Translation



Editing a page translation (for page users)

Slug : utilisateur

Title : Utilisateurs

Module's list :

Block "body" :
(You can change the position of each by dragging and dropping)

#	Name	Nature	Edit	Delete
44	Groupe User	groupmodules	Edit module	Delete

Block "navbar" :
(You can change the position of each by dragging and dropping)

#	Name	Nature	Edit	Delete
45	Groupe Navbar	groupmodules	Edit module	Delete

Block "footer" :
(You can change the position of each by dragging and dropping)

#	Name	Nature	Edit	Delete
23	footer text	content	Edit module	Delete

Add a module :

Block : body

Module : Slideshow - slideshow (Francais)

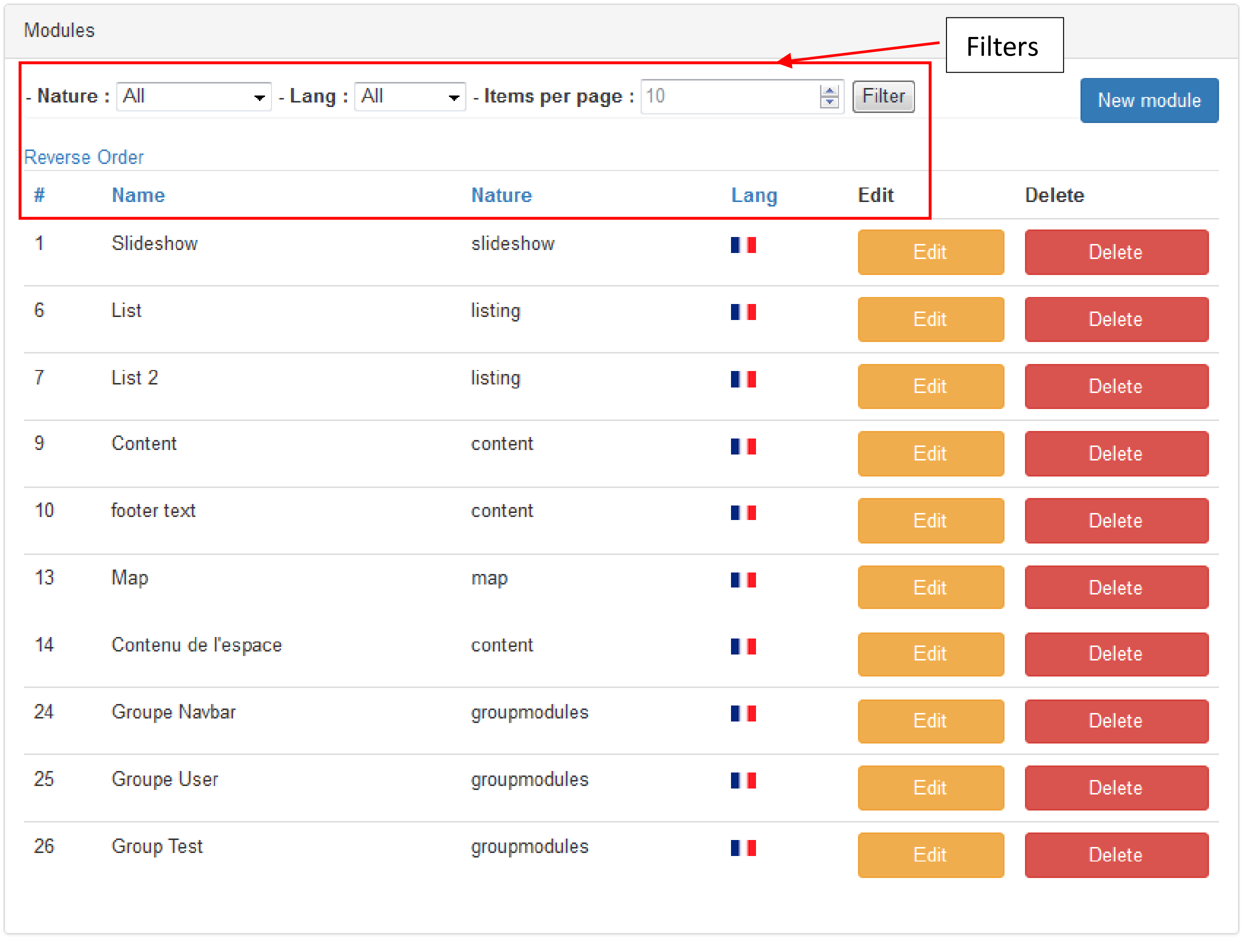
Edit page translation

You can add the number of module you want for each blocks. You also can order them by dragging and dropping them.

Modules

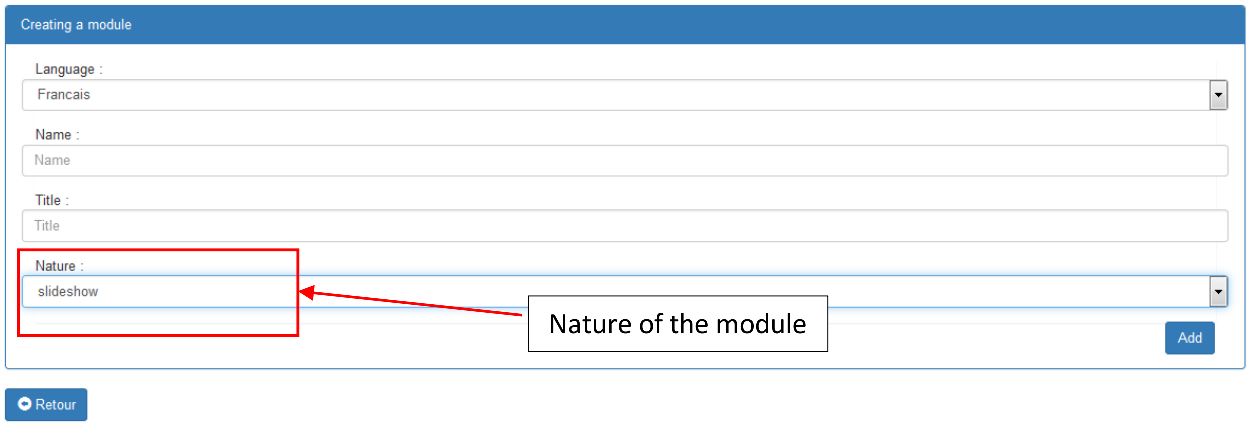
Each modules have a nature defining their usage (by default there is Slideshow, Content or group of modules…). Some module are unique (login, profile, changelang).

The edit page, will be different for each nature of module.



#	Name	Nature	Lang	Edit	Delete
1	Slideshow	slideshow	🇫🇷	Edit	Delete
6	List	listing	🇫🇷	Edit	Delete
7	List 2	listing	🇫🇷	Edit	Delete
9	Content	content	🇫🇷	Edit	Delete
10	footer text	content	🇫🇷	Edit	Delete
13	Map	map	🇫🇷	Edit	Delete
14	Contenu de l'espace	content	🇫🇷	Edit	Delete
24	Groupe Navbar	groupmodules	🇫🇷	Edit	Delete
25	Groupe User	groupmodules	🇫🇷	Edit	Delete
26	Group Test	groupmodules	🇫🇷	Edit	Delete

Module index



Creating a module

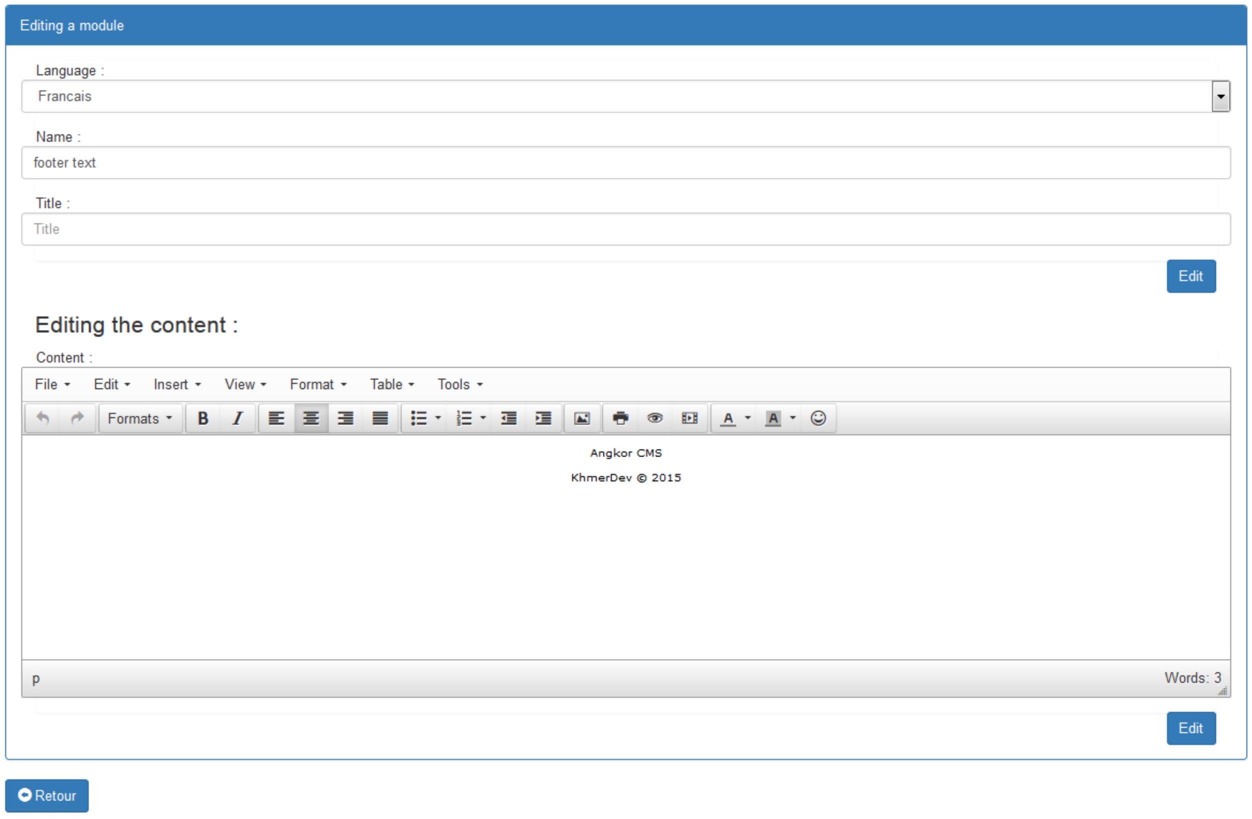
Language : Francais

Name : Name

Title : Title

Nature : slideshow

Create module



Editing a module

Language : Francais

Name : footer text

Title : Title

Editing the content :

Content :

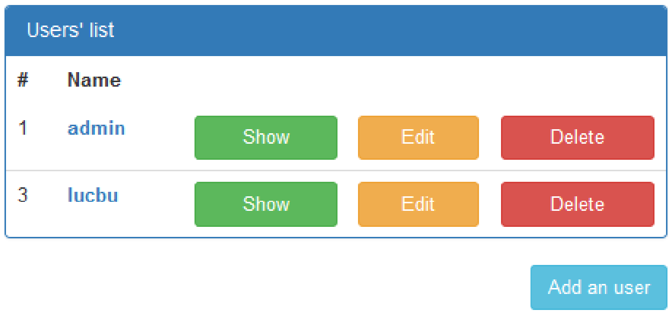
Angkor CMS

KhmerDev © 2015

Words: 3

Edit module (content)

Users

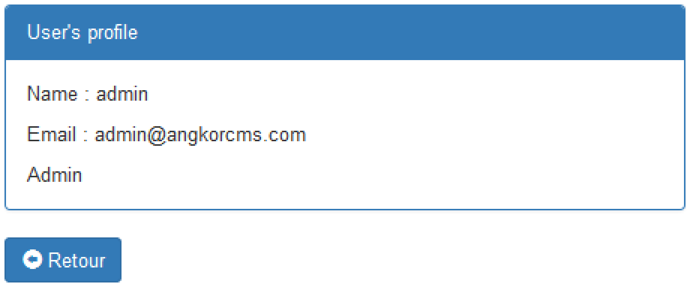


Users' list

#	Name			
1	admin	Show	Edit	Delete
3	lucbu	Show	Edit	Delete

Add an user

User index



User's profile

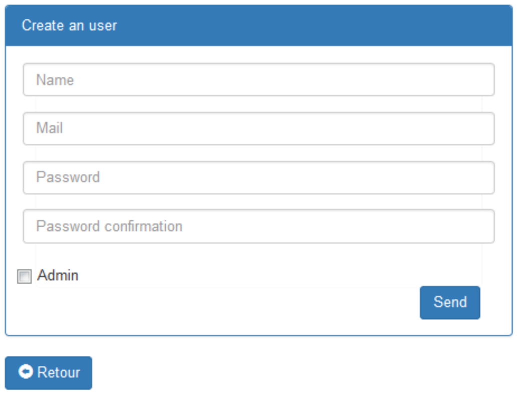
Name : admin

Email : admin@angkorcms.com

Admin

Retour

Show User



Create an user

Name

Mail

Password

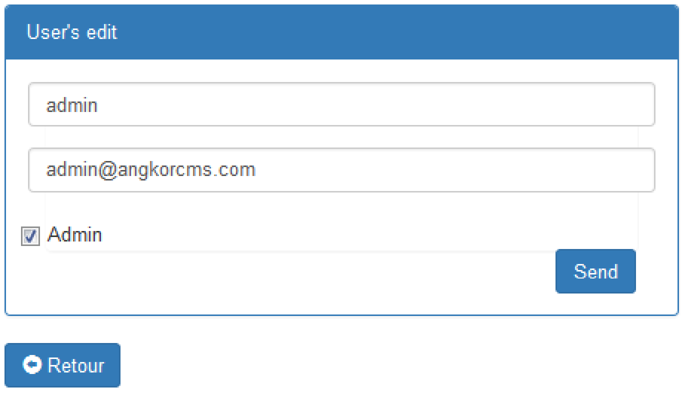
Password confirmation

☐ Admin

Send

Retour

Create user



User's edit

admin

admin@angkorcms.com

☑ Admin

Send

Retour

Edit User

Development

Create a new nature of module

For each module, you'll have to add a line in the file 'config\angkorcmsmodules' in the array 'nature' you'll have to add an item with as the key, the name you want to give to the module, and as value, the name of the config files associated to the modules.

(For example the module map is included like " **'map'=>'angkorcmsmap',** ")

Create the folder

To create each plugin I'm using the package called franzliedke/studio (GitHub).

I create plugins with the command:

```
vendor\bin\studio create foo/bar
```

Two kinds of modules

Unique:

A unique module is a module that can't be adapted to a new item, like the login plugin or the changelang plugin.

There is a need of a ViewComposer to render the view which has to be shown.

The installation of the plugin will be a seed of his information in the table angkorcms_modules and the publishing of his Blade/JS/CSS file.

Customizable:

A customizable module, is a module that depend on an item like a slideshow which will contains slides, or a map which will contains location or others information.

There is a need of a ViewComposer to render the view (named maker) which allow the user to edit the module.

The view used to show the module, will be loaded with the principal object defined in the config file with the name that you give to the module in 'angkorcmsmodules.php'. For example for the module map, the object AngkorCMSMap associated to each module of nature map, will be accessible in the show view under the variable $map. But you still can set a view composer to add data.

Adapt the Default-Plugin model

There is a folder called Default Plugin which contain all files needed to get include in a plugin. All files should be modified like below:

Customizable:

• -Object-	: Name of the main object
• -Plugin-	: Name of the plugin (ex. AngkorCMSSlideshow)
• -pluginmin-	: Name of the plugin with little letters (ex. angkorcmsslideshows)
• -namespace-	: Namespace (ex. AngkorCMS\Slideshow)
• -folderName-	: Name of the plugin's folder (ex. slideshow)
• -vendorName-	: Name of the vendor (ex. angkorcms)
• -showView-	: Name of the view to be shown

Unique:

• -Plugin-	: Name of the plugin (ex. AngkorCMSSlideshow)
• -pluginmin-	: Name of the plugin with little letters (ex. angkorcmsslideshows)
• -namespace-	: Namespace (ex. AngkorCMS\Slideshow)
• -folderName-	: Name of the plugin's folder (ex. slideshow)
• -vendorName-	: Name of the vendor (ex. angkorcms)
• -showView-	: Name of the view to be shown

You'll also have to customize the repository which will have to implement the interface in the contract folder.

You also can add translations to modules the way, you'll do it in any Laravel application.

For coding the view, you'll access to some data:

- $unique_id
- $module
- $'name_of_the_plugin' (Only if customizable plugin)
- $attr (attributes gave by the theme)
- $parameters (Parameters appearing after the slug in the URL)
- $attributes (attributes of the div around each module of a block)
- $div (if the module should have a div around itself)

But you also can add data by creating a ViewComposer getting launch when your view is called.

ViewComposer

A ViewComposer is a class that will fill a view with data whenever the view is called.

You declare the view composer in the service provider: (Example with the changelang)

```
View::composer('path/to/view/changelang', 'Namespace\To\Composer\ShowComposer');
```

And the view composer:

```
use AngkorCMS\MultiLanguages\Repositories\Eloquent\AngkorCMSLangRepository;
use Illuminate\Contracts\View\View;

class ShowComposer {

	protected $lang_repository;

	public function __construct(AngkorCMSLangRepository $lang_repository) {
		$this->lang_repository = $lang_repository;
	}

	public function compose(View $view) {
		//Get the data from the view
		$viewParameters = $view->getData();

		//Set languages to the $data
		$parameters = $viewParameters['parameters'];
		$data = array('langs' => $this->lang_repository->all());

		// Agregate and Send the data to the view
		$data = array_merge($data, $viewParameters);
		$view->with($data);
	}
}
```

Template

Create a template

When you'll create a theme, you'll have to upload a css, a js and a blade file.

For example the theme Light of the template Simple has his files like:

- Blade : AngkorCMS\resources\views\templates\Simple\Light
- JavaScript : AngkorCMS\public\js\Simple\Light
- CSS : AngkorCMS\public\css\Simple\Light

Like it's explained in the edition of a theme:

List of usable blocks:

- body
- footer
- navbar

The path to the css is "css/Simple/Light/file.css".

The path to the js is "js/Simple/Light/file.js".

To edit the view, you should use 'Blade' (Doc here)

You can access bootstrap 3.3.2 (CSS) here: "css/bootstrap-3.3.2-dist/css/bootstrap.min.css".

You can access bootstrap 3.3.2 (JS) here: "js/bootstrap.min.js".

You can access jQuery 2.1.3 at "js/jquery-2.1.3.min.js"

You have access to several variables:

- $title (title of the page)
- $parameters (list of argument in the url after the page's slug)
- $blocks (use it like "{!!AngkorBlade::display($blocks['blockName'], $parameters)!!}")

Share the template

To share the template, you have to create a seeding class to seed data of your template in the database (template, blocks, and theme).

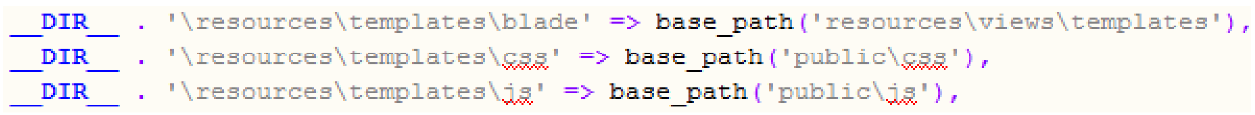
```
class ChangeLangModuleTableSeeder extends Seeder {
    public function run() {
        DB::table('angkorcms_modules')->insert(array(
            'name' => 'Change Lang',
            'title' => '',
            'unique' => true,
            'nature' => 'lang',
            'lang_id' => null,
        ));
    }
}
```

And the method run in the class of the command:

```
$this->call('db:seed', array('--class' => 'AngkorCMS\Users\Database\Seeds\DatabaseSeeder'));
```

Plus, you'll have to publish all the files needed in their appropriate folder.

The ServiceProvider will contains something like:



```
__DIR__ . '\resources\templates\blade' => base_path('resources\views\templates'),
__DIR__ . '\resources\templates\css' => base_path('public\css'),
__DIR__ . '\resources\templates\js' => base_path('public\js'),
```