

Partie IV

Création d'une page d'administration avec Prestashop

-AdminController-

Un **AdminController** dans PrestaShop est une classe utilisée pour gérer les opérations et l'affichage dans le **Back Office**. Il agit comme un contrôleur pour les interfaces administratives, permettant d'interagir avec des données, d'exécuter des actions (ajouter, modifier, supprimer, exporter, etc.), et de créer des interfaces administratives pour les modules.

Nous allons voir les différentes étapes pour réaliser facilement un contrôleur admin basique lié à un module Prestashop. Puis nous verrons ensuite les caractéristiques de base d'un contrôleur admin

Ajout de liens de modules dans le menu latéral du back-office

Sur le back office PrestaShop, les liens du menu latéral sont liés aux classes AdminControllers et ModuleAdminController. Les premiers proviennent du PrestaShop Core, mais les seconds sont définis par les modules.

Si vous souhaitez ajouter un lien vers votre ModuleAdminController dans la barre latérale du back-office :

- Créer dans le dossier monmodule un sous dossier « controllers/admin »
- Créer le fichier php « AdminTest.php » contenant la classe « AdminTestController » suivante :

```
class AdminTestController extends ModuleAdminController {}
```

- Maintenant, dans la classe MonModule du fichier monmodule.php créer deux fonctions installTab() et uninstallTab(). Ces deux méthodes permettent d'installer et de désinstaller des contrôleurs dans le backoffice.

```

private function installTab()
{
    $tab = new Tab();
    $tab->class_name = 'AdminTest';
    $tab->module = $this->name;
    $tab->id_parent = (int)Tab::getIdFromClassName('DEFAULT');
    $tab->icon='settings_applications';

    foreach (Language::getLanguages() as $lang) {
        $tab->name[$lang['id_lang']] = 'Test Admin controller';
    }
    return $tab->add();
}

private function uninstallTab()
{
    $id_tab = (int)Tab::getIdFromClassName('AdminTest');
    if ($id_tab) {
        $tab = new Tab($id_tab);
        return $tab->delete();
    }

    return false;
}

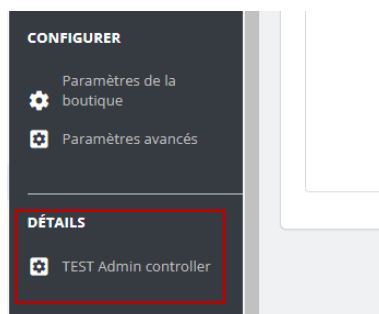
```

Classe appelée lorsque l'utilisateur cliquera sur votre lien. C'est le nom de la classe sans la *Controller* partie.

Spécifier le menu parent

Libellé affiché dans le menu

- Invoquer la méthode `installTab()` dans la fonction `install()` de la classe `MonModule`, ainsi que la méthode `uninstallTab()` dans la fonction `uninstall()`.
- Réinitialiser maintenant votre module installé



Création du modèle

- Tout d'abord, créer une table dans la base de données nommée « ps_testcomment » :

☐	1	id	int		Non	Aucun(e)	AUTO_INCREMENT	Modifier	Supprimer
☐	2	username	varchar(30)	utf8mb4_0900_ai_ci	Non	Aucun(e)		Modifier	Supprimer
☐	3	comment	varchar(100)	utf8mb4_0900_ai_ci	Non	Aucun(e)		Modifier	Supprimer

- Maintenant, dans le dossier monmodule, créer un sous dossier nommé « classes »
- Créer dans ce dossier un fichier nommé « comment.class.php » contenant le code suivant :

```
<?php
class CommentTest extends ObjectModel
{
    public $id;
    public $username;
    public $comment;
    public static $definition = [
        'table' => 'testcomment', // Nom de la table dans la base de données sans _ps
        'primary' => 'id', // Clé primaire de la table
        'fields' => [
            'username' => [
                'type' => self::TYPE_STRING, // Type de données
                'validate' => 'isString', // Méthode de validation
                'required' => true, // Champ obligatoire
                'size' => 255, // Longueur maximale
            ],
            'comment' => [
                'type' => self::TYPE_STRING,
                'validate' => 'isCleanHtml',
                'required' => true,
                'size' => 255,
            ],
        ],
    ];
}
```

Affichage des objets 'TestComment'

La fonction par défaut du contrôleur est d'afficher une liste des objets. Pour cela il faut renseigner toutes les informations nécessaires dans la fonction `__construct()` de la classe AdminTestController

```

<?php
//inclure le fichier php contenant la classe de l'objet que nous allons gérer
require_once(_PS_MODULE_DIR_.'mymodule/classes/comment.class.php');
class AdminTestController extends ModuleAdminController
{
    public function __construct() {
        // Configuration du contrôleur
        $this->table = 'testcomment'; // Nom de la table à gérer
        $this->className = 'CommentTest'; // Classe associée
        $this->bootstrap = true; // Utilisation de Bootstrap pour l'interface
        $this->identifiant = CommentTest::$definition['primary']; // Clé primaire de la table
        // Définir les champs de la liste
        $this->fields_list = [
            'id' => [
                'title' => 'ID',
                'align' => 'center',
            ],
            'username' => [
                'title' => 'USERNAME',
                'align' => 'center',
            ],
            'comment' => [
                'title' => 'COMMENT',
                'align' => 'center',
            ]
        ];
        //ajout d'actions sur chaque ligne
        $this->addRowAction('edit');
        $this->addRowAction('delete');
        $this->addRowAction('view');
        // Appeler le constructeur parent
        parent::__construct();
    }
}

```

Avec ce code nous aurons l’affichage suivant :

Détails / Test Admin controller

Test Admin controller

Aide

Test Admin controller 0

ID	USERNAME	COMMENT
 Aucun enregistrement trouvé		

Ajout / Édition d'un objet TestComment

Pour l'ajout ou l'édition d'un objet, le contrôleur doit implémenter la fonction *renderForm()* avec le contenu suivant :

```
// Affichage du formulaire d'ajout / création de l'objet
public function renderForm()
| //définition du formulaire d'edition
| { $this->fields_form=[
|     'legend'=>[
|         'title'=>'New Comment',
|         'icon'=>'icon-cog'],
|     //champs
|     'input'=>[
|         [
|             'type'=>'text',
|             'required'=>true,
|             'label'=>'The user',
|             'class'=>'input fixed-with-sm',
|             'name'=>'username',
|             'empty_message'=>'Please fil the input'],
|         [
|             'type'=>'text',
|             'required'=>true,
|             'empty_message'=>'Please fil the comment',//message d'erreur si vide
|             'label'=>'The Comment',
|             'class'=>'input fixed-with-sm',
|             'name'=>'comment']
|     ],
|     //Boutton de soumission
|     'submit'=>[
|         'title'=>'Submit the comment',
|         'class'=>'btn btn-primary pull-right']
| ];
| return parent::renderForm(); }
```

Le rendu sera ensuite le suivant :

Détails / TEST Admin controller

Ajouter ? Aide

NEW COMMENT

* The user

* The Comment

 Annuler

 SUBMIT THE COMMENT

Consulter les détails d'un objet avec DBQuery

Pour consulter les détails d'un objet, le contrôleur doit implémenter la fonction *renderView()* avec le contenu suivant :

```
public function renderView()
{
    $tplFile=dirname(__FILE__)."/../views/templates/admin/view.tpl";
    $tpl= $this->context->smarty->createTemplate($tplFile);
    $sql=new DbQuery();
    $sql->select('*')
        ->from($this->table)
        ->where('id='.Tools::getValue('id'));

    $data= DB::getInstance()->executeS($sql);

    $tpl->assign([
        'data' => $data[0]
    ]);
    return $tpl->fetch();
}
```

- Créer un fichier view.tpl dans « views/templates/admin » contenant le code suivant :

```
<h2>
{$data.user_name}
</h2>
```

- Lorsque vous cliquez sur une ligne du tableau le nom de cet objet sera affiché :

Détails / TEST Admin controller

Afficher

meriem