



Rapport technique

Refonte de l'espace client de Franche Comté net



Stage de fin d'étude DUT Informatique
Du 28 Mars au 04 Juin 2016



Table des matières

Table des matières	3
1 Smart Admin.....	5
2 Espace client.....	6
2.1 Généralités	6
2.1.1 Structure de l'espace client.....	6
2.1.2 Bases de données.....	8
2.1.3 Design.....	9
2.2 Classes système.....	10
2.2.1 DBControl.....	10
2.2.2 Model	10
2.2.3 Smart Admin	10
2.2.4 Session Control	11
2.2.5 Controller	11
2.3 Fonctionnalités.....	12
2.3.1 Système de connexion	12
2.3.2 Oubli de mot de passe	12
2.3.3 Demande d'accès	13
2.3.4 Gestion des DNS.....	13
2.3.5 Gestion des boites mails	14
2.3.6 Boites mails Exchange.....	15
2.3.7 Boites mails Icewarp	19
3 Back Office.....	22
3.1Généralités	22
3.1.1 Structure du back office	22
3.1.2 Base de données	23
3.2 Fonctionnalités.....	24
3.2.1 Système de connexion	24
3.2.2 Gestion du carrousel	25
3.2.3 Gestion des références web	28
3.2.4 Gestion des actualités	29
3.2.5 Gestion de l'espace client	29
4 Table des illustrations.....	33

1 Smart Admin

Smart admin est une template de panneau d'administration proposant un design complet ainsi qu'une structure sous plusieurs formes. La structure adoptée pour la réalisation du nouvel espace client et celle de son back-office est basée sur du PHP / HTML. Etant une template complète et facilement modulable, elle fut choisie par l'entreprise comme étant la base du nouvel espace client.

Elle possède divers fichiers de configuration qui permettent son initialisation, allant de la liste des liens de la navbar à l'affichage des éléments de la page.

```
$page_nav = array(  
    "dashboard" => array(  
        "title" => "Dashboard",  
        "icon" => "fa-home",  
        "sub" => array(  
            "analytics" => array(  
                "title" => "Analytics Dashboard",  
                "url" => APP_URL  
            ),  
            "social" => array(  
                "title" => "Social Wall",  
                "url" => APP_URL."/dashboard-social.php"  
            )  
        )  
    ),  
);
```

Exemple de configuration de navbar 1

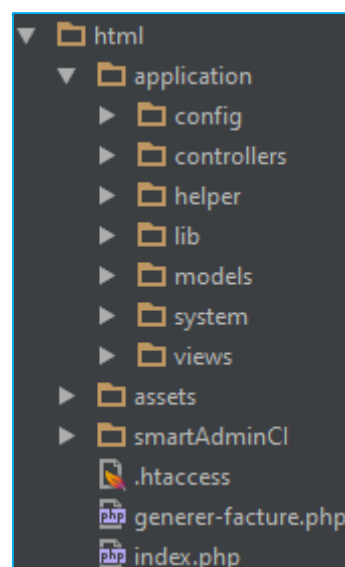
2 Espace client

2.1 Généralités

2.1.1 Structure de l'espace client

La structure de l'espace client est basée sur un MVC inspiré de la template CodeIgniter. En effet, il fallait proposer un système permettant l'intégration de la template SmartAdmin comprenant aussi tous ses fichiers de configuration.

- **Config** : contient les fichiers de configuration de l'application, de SmartAdmin ainsi que le kernel
- **Controllers** : contient les différents contrôleurs
- **Helper** : contient les fichiers parfois nécessaires pour certaines pages
- **Lib** : contient les librairies et les class de SmartAdmin
- **Models** : contient les models
- **System** : contient les classes systèmes (Controller, Base de données, models, Session et SmartAdmin) dont la plupart des fichiers héritent
- **Views** : contient les différentes pages de vue
- **Model** : contient les différents models



La navigation se fait grâce à un fichier nommé kernel.php qui permet la gestion des controllers, des models ainsi que des vues. La structure inclut aussi différents éléments tels que les fichiers de configuration, notamment ceux nécessaire au bon Fonctionnement de SmartAdmin.

Le kernel sépare, à l'aide du fichier HTaccess, les paramètres entrés dans l'url. Cela est possible grâce aux paramètres « RewriteEngine » et « RewriteRule ».

```
RewriteEngine on

RewriteRule ^assets - [L,NC]
RewriteRule ^generer-facture.php - [L,NC]

RewriteRule ^.+ $ index.php [L]
```

Fichier htaccess 1

```
<?php
/*
 * Router de l'application
 * Ce fichier sépare les Controllers, les méthodes et les arguments et les appelle si possible
 * Si cela n'est pas possible, il renvoie l'erreur 404
 */
define("SITE_PATH",realpath(dirname( __FILE__ ).DIRECTORY_SEPARATOR);
$tab_url1=explode('/', $_SERVER['REQUEST_URI']);
$tab_url2=explode('/', $_SERVER['SCRIPT_NAME']);
$tab3=array_diff($tab_url1, $tab_url2);
if($c=array_shift($tab3)) $controllerName=$c; else $controllerName=default_controller;
if($m=array_shift($tab3)) $methodName=$m; else $methodName="index";
if(isset($tab3[0])) $argsArray=$tab3; else $argsArray=array();
if($controllerName == '/') $controllerName = default_controller;
$controllerFile=CONTROLLERS_PATH.strtolower($controllerName)."_c.php";

//404 not found si la valeur de nf est égale à "true"
$nf = false;
```

Kernel de l'application 1

Ce code ci-dessus représente la partie du kernel qui permet la séparation des différents arguments passés dans l'url.

```
define("APP_PATH","application/"); //Dossier contenant l'application
define("CONFIG_PATH", APP_PATH."config/"); //Dossier contenant les configs de l'application

/*
 * Configuration de base
 * Définit les constantes et inclus les fichiers nécessaire
 */
require_once(CONFIG_PATH.'config.php');

/*
 * Inclusion du kernel
 * Kernel : fichier qui vérifie l'url et appelle les controllers / méthodes, en passant les arguments
 */
require_once(CONFIG_PATH."kernel.php");

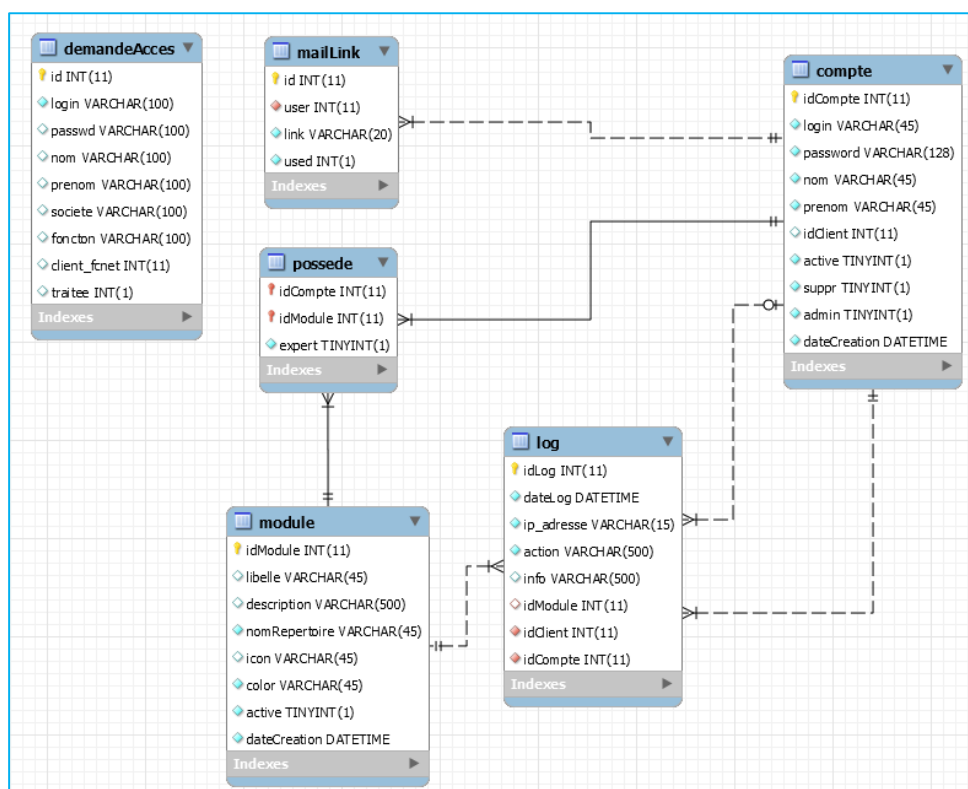
?>
```

Fichier index.php 1

Le fichier index.php, base de l'application définit les constantes **APP_PATH** et **CONFIG_PATH**, qui définissent respectivement la localisation des fichiers de l'application et la localisation des fichiers de configuration. Il inclut aussi les différents fichiers nécessaires au fonctionnement de l'application.

2.1.2 Bases de données

La base de données de l'espace client est une base de données MySQL. Celle-ci comporte différentes tables nécessaires au bon fonctionnement du site.



MCD Espace client 1

- La table « compte » contient les différents comptes utilisateurs. Le login correspond à l'adresse mail du compte. L'idClient correspond au numéro de compte FCnet du client. En effet, chaque client de FCnet possède un identifiant unique et cette colonne permet de faire le lien entre le compte FCnet et le compte de l'espace client.

« Active » permet de savoir si l'utilisateur est en ligne ou non. Lorsque « Suppr » est à la valeur « 1 », cela empêche l'utilisateur de se connecter. La variable « Admin » définit si le client est administrateur ou non.

- La table « module » contient les différents modules de l'espace client ainsi que ses diverses informations.

- La table « possède » contient les modules auquel ont accès chaque client.

- La table « log » contient les actions effectuées sur l'espace client ainsi que diverses informations

- La table « mailLink » contient les liens de changement de mot de passe

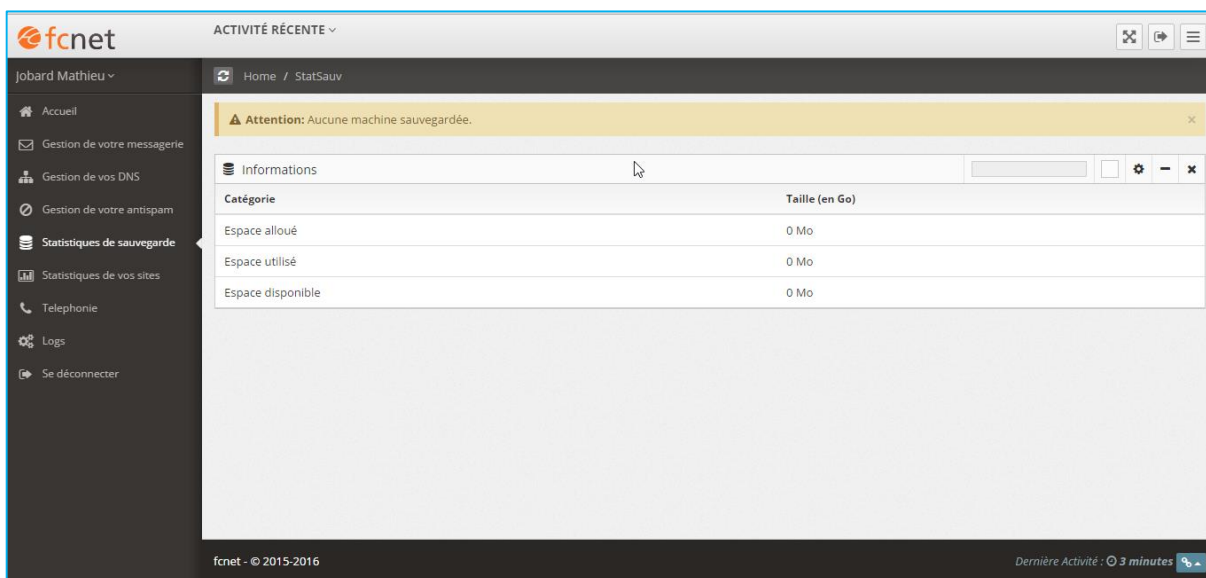
- La table « demandeAcces » contient les demandes de création de compte.

2.1.3 Design

L'espace client est, comme dit précédemment, basé sur celui d'un panneau d'administration. La template utilisée est celle de Smart Admin. Celle-ci permet de rajouter automatiquement des feuilles de styles selon le besoin. Les feuilles de style se trouvent dans le dossier « assets/css/ ». L'ensemble du code permettant la personnalisation de la template est présent dans le fichier style.css et contient notamment la modification des couleurs afin que celle-ci soient en accord avec celle de la société.

```
.btn-primary.disabled,
.btn-primary[disabled],
fieldset[disabled] .btn-primary,
.btn-primary.disabled:hover,
.btn-primary[disabled]:hover,
fieldset[disabled] .btn-primary:hover,
.btn-primary.disabled:focus,
.btn-primary[disabled]:focus,
fieldset[disabled] .btn-primary:focus,
.btn-primary.disabled.focus,
.btn-primary[disabled].focus,
fieldset[disabled] .btn-primary.focus,
.btn-primary.disabled:active,
.btn-primary[disabled]:active,
fieldset[disabled] .btn-primary:active,
.btn-primary.disabled.active,
.btn-primary[disabled].active,
fieldset[disabled] .btn-primary.active {
background-color: #e8490f;
border-color: #f76833;
}
.btn-primary .badge {
color: #e8490f;
background-color: #ffffff;
}
```

Exemple de css 1



Design de l'espace client 1

2.2 Classes système

2.2.1 DBControl

La class DBControl contient les différentes méthodes permettant l'interaction avec les bases de données. Elle récupère les accès aux bases de données depuis un fichier de configuration nommé dbConfig.php.

2.2.2 Model

La class Model est héritée par tous les models et permet l'interaction avec la class DBControl.

```
/*
 * Class Model
 * elle permet aux model d'accéder aux bases de données
 */
class Model
{
    /*
     * db => Instance de la base de donnée
     */
    public $db;

    public function __construct()
    {
        require_once("DBControl.php");
        $this->db = new DBControl();
    }
}
```

Class model 1

2.2.3 Smart Admin

La class Smart Admin contient toutes les méthodes permettant l'affichage de l'interface de la template smart admin. Il était plus simple de créer une class pour gérer toutes les parties de la template plutôt que d'inclure chaque fichier un par un. Cette classe a donc pour but de faciliter la lecture et la manipulation du code.

```
/*
 * On inclut les fichiers de config dont a besoin smartAdmin
 */
public function __construct()
{
    $this->inc = VIEWS_PATH."inc/";
    require_once(LIB_PATH."smartadmin/config.php");
    require_once(LIB_PATH."smartadmin/smartui/class.smartui.php");
    $this->db = new DBControl();
}
```

Constructeur de SmartAdmin 1

La class Smart Admin est composé de la méthode « init » et initialise la page grâce à un tableau passé en paramètre. Ce tableau comporte des éléments tels que le nom de la page ou encore les feuilles de styles supplémentaires à inclure. La méthode affiche ensuite le header et la navbar. Cette méthode doit donc être appelé à chaque instantiation d'une page.

La class est aussi composé de la méthode « end » qui elle, inclut le footer ainsi que les scripts de la page.

```
public function index(){
    $this->smartAdmin()->init(array("page" => "home"));
    $this->loadView("pages/dashboard");
    $this->smartAdmin()->end();
}
```

Exemple d'utilisation de smartadmin 1

2.2.4 Session Control

La class Session Control permet la gestion des variables de session. Elle instancie une session et comporte diverses méthodes qui simplifie la récupération de valeurs contenue dans les variables de session.

2.2.5 Controller

La class Controller est la classe héritée par chaque controller. Elle possède différentes méthodes qui permettent de charger des fichiers tels que les vues, les models, les librairies ainsi que les helpers. Le chargement des différents fichiers se fait par l'intermédiaire de la fonction « load ». Elle fait aussi la liaison entre les différents fichiers système.

```
/*
 * Fonction privé qui charge les éléments.
 * Elle inclut le fichier contenu dans $path
 * Elle charge les données de $data avec extract :
 * $data est un tableau associatif contenant le nom des variables et leur contenu
 * Il est initialisé à null car on est pas toujours obligé de passer des données en chargeant un élément
 * Un modèle n'a pas besoin de données pour être appelé par exemple
 *
 * Sur la plupart (non toute mais presque), la classe UI était instanciée
 * et il y avait une variable js vide.
 * Les deux premières lignes de code servent à les ajouter en appelant la page.
 * Si on en a pas besoin, cela ne change rien, or, si on en a besoin,
 * ces deux variables seront déjà instanciées / initialisé
 */
private function load($path, $data = null){
    if(!isset($data['js'])) $data['js'] = "";
    if(!isset($data['ui'])) && class_exists("SmartUi") $data['ui'] = new SmartUi();

    if($data != null)
        extract($data);
    require_once($path."_php");
}

/*
 * Permet de charger une vue avec ses éventuelles données
 */
public function loadView($path, $data = null){
    $this->load(VIEWS_PATH.$path, $data);
}
```

Méthode load de Controller 1

```
class Controller
{
    /*
     * Class controller qui gère les controllers
     */
    public $SmartAdmin; //La classe SmartAdmin
    public $model; //La classe Model (actuellement vide)
    public $session; //La classe SessionControl
    public $db; //La classe DBControl

    public function __construct() {
        $this->SmartAdmin = new SmartAdmin();
        $this->session = new SessionControl();
        $this->db = new DBControl();
    }
}
```

Liaison entre les différentes class 2

2.3 Fonctionnalités

2.3.1 Système de connexion

La page permettant le login d'un utilisateur est gérée par le controller « Login_c » et le model « Login_m ». Si l'utilisateur est déjà en ligne, il est redirigé vers la page d'accueil de l'espace client.

Les mots de passes utilisateur sont cryptés par un algorithme de type SHA256

```
private function _hashPassword($clear, $salt = null)
{
    if(!$salt) $salt = bin2hex(openssl_random_pseudo_bytes(8));
    return '$SHA$' . $salt . '$' . hash('sha256', hash('sha256', $clear) . $salt);
}

private function _checkPassword($clear, $hash)
{
    $parts = explode('$', $hash);
    return $hash === $this->_hashPassword($clear, $parts[2]);
}
```

Cryptage sha256 1

avec *salt* et sont stockés ainsi dans la base de données.

2.2.2 Oubli de mot de passe

L'oubli de mot de passe géré par le même controller et le même model que le système de connexion. Le code est géré depuis la méthode « forgotPassword » présente dans le model Login_m.

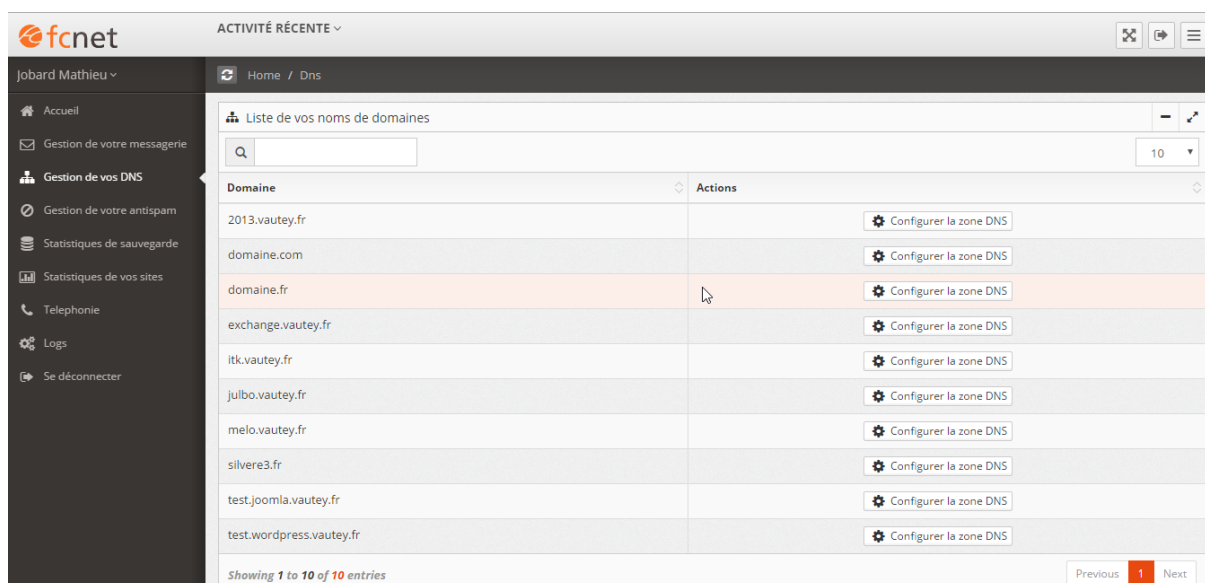
Lorsqu'une demande de récupération de mot de passe est faite, l'utilisateur doit saisir un captcha ainsi que son adresse e-mail. Le serveur génère ensuite un lien grâce à une chaîne de caractères aléatoire qu'il envoie ensuite par e-mail à l'adresse en question et qu'il stocke dans la table « mailLink ». Depuis ce lien, l'utilisateur peut changer son mot de passe. Une fois le changement opéré, le lien devient invalide.

2.2.3 Demande d'accès

La demande d'accès consiste à demander un accès à l'espace client dans le cas où l'utilisateur possède un compte client FCnet. Chaque demande d'accès est stockée dans la table « demandeAcces » et se suit par l'envoi d'un mail au personnel de l'entreprise qui comprend les informations entrées par l'utilisateur. Les comptes sont ensuite créés manuellement depuis le back-office. Les demandes d'accès sont gérées depuis la méthode « new_account » présente dans le model Login_m.

2.2.4 Gestion des DNS

Le module de gestion des DNS se divise en 2 parties : L'affichage et le paramétrage. Le serveur récupère tout d'abord les noms de domaines correspondant au client connecté depuis la base de données et les affiche ensuite dans un tableau.



ACTIVITÉ RÉCENTE

Jobard Mathieu

Home / Dns

Liste de vos noms de domaines

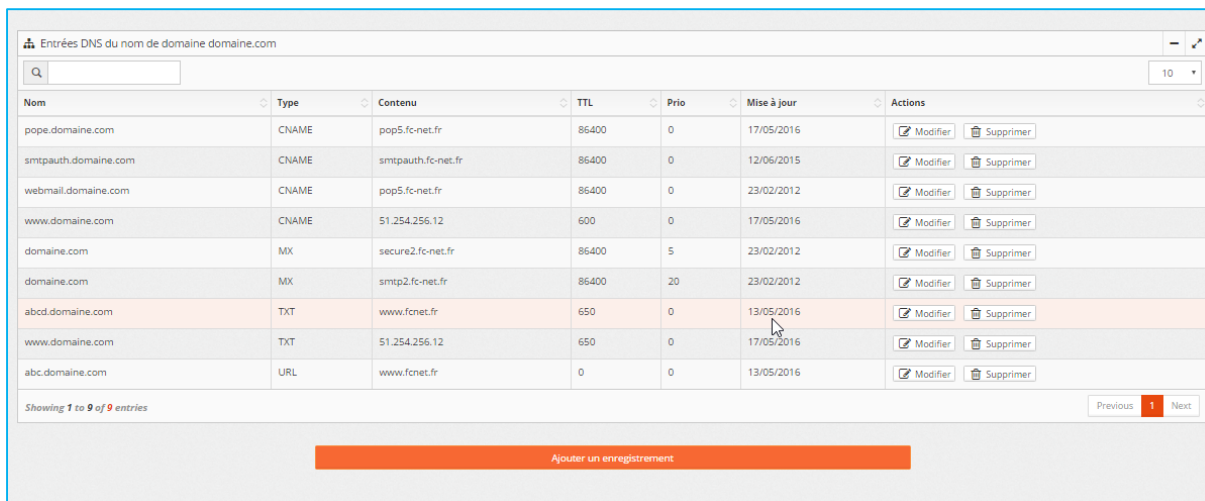
Domaine	Actions
2013.vautey.fr	Configurer la zone DNS
domaine.com	Configurer la zone DNS
domaine.fr	Configurer la zone DNS
exchange.vautey.fr	Configurer la zone DNS
itk.vautey.fr	Configurer la zone DNS
julbo.vautey.fr	Configurer la zone DNS
melo.vautey.fr	Configurer la zone DNS
silvere3.fr	Configurer la zone DNS
test.joomla.vautey.fr	Configurer la zone DNS
test.wordpress.vautey.fr	Configurer la zone DNS

Showing 1 to 10 of 10 entries

Previous 1 Next

Tableau de gestion des DNS 1

L'interface de paramétrage des entrées DNS a été simplifiée au maximum et permet l'ajout, la suppression ou la modification d'une entrée DNS.



Nom	Type	Contenu	TTL	Prio	Mise à jour	Actions
pope.domaine.com	CNAME	pop5.fc-net.fr	86400	0	17/05/2016	Modifier Supprimer
smtpauth.domaine.com	CNAME	smtpauth.fc-net.fr	86400	0	12/06/2015	Modifier Supprimer
webmail.domaine.com	CNAME	pop5.fc-net.fr	86400	0	23/02/2012	Modifier Supprimer
www.domaine.com	CNAME	51.254.256.12	600	0	17/05/2016	Modifier Supprimer
domaine.com	MX	secure2.fc-net.fr	86400	5	23/02/2012	Modifier Supprimer
domaine.com	MX	smtp2.fc-net.fr	86400	20	23/02/2012	Modifier Supprimer
abcd.domaine.com	TXT	www.fcnet.fr	650	0	13/05/2016	Modifier Supprimer
www.domaine.com	TXT	51.254.256.12	650	0	17/05/2016	Modifier Supprimer
abc.domaine.com	URL	www.fcnet.fr	0	0	13/05/2016	Modifier Supprimer

Showing 1 to 9 of 9 entries

[Ajouter un enregistrement](#)

Tableau de paramétrage des DNS 1

Il existe de nombreux types d'entrées DNS. Celles gérées par l'espace client sont les entrées DNS de type A, CNAME, TXT, URL, MX et SRV.

- Un enregistrement de type A doit correspondre à une adresse IP.
- Un enregistrement de type CNAME doit être un nom
- Un enregistrement de type URL doit être une adresse WEB.
- Un enregistrement de type MX (ou Mail eXchanger) associe un nom de domaine à un serveur de messagerie électronique associé à son numéro de préférence. Ils ne doivent pouvoir être ajoutés, modifiés ou supprimés que par les administrateurs
- Un enregistrement de type SRV (ou enregistrement de service) est une catégorie de données du DNS d'Internet qui vise à indiquer quels sont les services disponibles. Ce sont des entrées très complexes gérables uniquement, comme dans le cas des types MX, par les administrateurs.

La gestion des entrées DNS se fait par la modification de la base de données ou elles sont stockées. Ceci est géré par le model « dns_m ». Les différentes actions ont été regroupées dans des méthodes. Les méthodes d'ajout, de modification et de suppression permettent respectivement l'ajout, la modification ou la suppression de n'importe quel type d'entrée DNS. Chaque action est suivie d'une vérification d'entrée des données très stricte afin que celle-ci correspondent aux critères de l'entrée spécifiée.

2.2.5 Gestion des boîtes mails

Les services mails proposés par FCnet sont des boîtes mails Icewarp et des boîtes Mails exchange. L'interface de gestion des boîtes mails est la même pour les

deux types, mais il a pour cela recours à de nombreuses fonctions. Le tout est géré par le controller « Mail_c » et par le model « Mail_m ».

Le serveur récupère tout d'abord les noms de domaines possédés par l'utilisateur. Il n'affiche ensuite que les noms de domaines possédant une boîte mail. Pour cela, il vérifie s'il existe des boîtes mails Icewarp présente sur le domaine. Si oui, le serveur en conclut que le domaine en question possède une boîte mail Icewarp, si non, il vérifie le nombre de boîte mail Exchange. S'il n'existe pas de boîtes mail Exchange, le serveur en conclut que le domaine n'a aucun service mail associé.

2.2.6 Boîtes mails Exchange

Microsoft Exchange est un serveur mail fonctionnant par organisation. Chacune d'entre elle peut contenir plusieurs domaines mais ne possède qu'un seul administrateur, celui-ci ayant accès à la gestion de l'organisation en question. Les services Exchange proposés par FCnet sont la gestion des boîtes mails, des contacts externes et la gestion des groupes de distribution.

- Un contact externe est un contact extérieur à l'organisation
- Un groupe de distribution contient plusieurs boîtes mails et redirige ses messages sur l'ensemble de ses boîtes.

Le serveur Exchange de FCnet est hébergé sur un Windows-server et est administrable manuellement depuis le langage PowerShell. Afin de permettre l'interaction entre l'espace client et le serveur, un web service a été mis en place. Celui-ci est développé en C# et il permet l'exécution de commandes Power Shell.

Un web service (ou service web) est un programme informatique de la famille des technologies web permettant la communication et l'échange de données entre applications et systèmes hétérogènes dans des environnements distribués.

L'espace client communique depuis ce web service grâce un client SOAP. Un client SOAP autorise un objet à invoquer des méthodes physiquement situés sur un

autre serveur. Il est mis en place tel que ceci :

```
// Récupère les boîtes mails d'une organisation Exchange
function listingBoite_Exchange($domain)
{
    $wsdl = "http://ws.exch.fc-net.fr/webservice1.asmx?wsdl";
    try
    {
        $soap_options = array(
            'exceptions' => true,
            'trace' => true,
            'cache_wsdl' => WSDL_CACHE_NONE,
            'features' => SOAP_SINGLE_ELEMENT_ARRAYS,
        );

        $admin = $this->get_admin_account($domain);
        if($admin == '') return null;
        $client = new SoapClient($wsdl, $soap_options);
        // $params->param = 'Hello World';
        $result = $client->GetMailBox(array("admin" => $admin));
        return $result;
    }
    catch (SoapFault $fault)
    {
        echo 'ERROR: '.$fault->faultcode." - ".$fault->faultstring."<br/>";
        return;
    }
}
```

Récupération de boîtes depuis un client SOAP 1

La seule boîte mail pouvant agir sur l'organisation Exchange étant la boîte mail administrateur, il faut donc passer la passer en paramètre lors de chaque appel de méthode. Pour cela, il a été mis en place une table qui associe à chaque domaine Exchange sa boîte mail Administrateur. La méthode « get_admin_account » permet de récupérer l'administrateur en question.

```
function get_admin_account($domain) {
    if($this->selectTypeBoiteMail($domain) == 1)
        return $this->get_domains_info($domain)["administrateur"];
    else
        return $this->getAdminIfExistsIcewarp($domain);
}

function get_domains_info($domain) {
    $this->db->Connect_MSSQL();
    $query = mssql_query("SELECT * FROM domaines_mail WHERE domaine='".$domain."'");
    if(!mssql_num_rows($query))
        return null;
    $res = mssql_fetch_array($query);
    return $res;
}
```

Récupération administrateur exchange 1

Au niveau du web service, afin de pouvoir agir sur l'organisation, celui-ci récupère les droits administrateurs depuis la méthode « connectExchange » qui prend en paramètre la boîte mail administrateur. Le mot de passe étant toujours le même pour chaque domaine, celui-ci est stocké en dur dans le code.

```
// Connexion au serveur exchange pour permettre les requetes PowerShell
public WSMANConnectionInfo connectExchange(String admin)
{
    string userName = admin;
    string password = "*****";
    System.Security.SecureString securePassword = new System.Security.SecureString();
    foreach (char c in password)
        securePassword.AppendChar(c);

    PScredential credential = new PScredential(userName, securePassword);
    WSMANConnectionInfo connectionInfo = new WSMANConnectionInfo(new Uri("https://exchange.fcnet.fr/powershell"),
        "http://schemas.microsoft.com/powershell/Microsoft.Exchange", credential);
    connectionInfo.AuthenticationMechanism = AuthenticationMechanism.Basic;
    connectionInfo.MaximumConnectionRedirectionCount = 2;

    return connectionInfo;
}
```

Connexion Exchange depuis le Web Service 1

PowerShell dispose de nombreuses commandes permettant de gérer un serveur exchange. Il retourne ses résultats sous forme de tableau.

```
PS C:\Users\admiis\Desktop> Get-Mailbox

Name                Alias                ServerName            ProhibitSendQuota
----                -
Administrator       Administrator        hexe                  1.98 GB (2,126,009,344 bytes)
contact              contact              hexe                  1.98 GB (2,126,009,344 bytes)
DiscoverySearchMailbox... DiscoverySearchMa... hexe                  50 GB (53,687,091,200 bytes)
```

Exemple d'utilisation de PowerShell 1

L'exécution de Power Shell depuis une méthode C# se fait par le biais de l'objet « Command ». On peut ainsi y passer des paramètres et recevoir le résultat de la commande.

```
/// <summary>
/// Ajoute une boîte mails (commande New-Mailbox) avec les paramètres nécessaire
/// </summary>
[WebMethod]
public Retour addMailBox(String alias, String name, String firstname, String lastname, String userprincipalname, String passw
{
    WSMANConnectionInfo connectionInfo = connectExchange(admin);
    String conf = "";
    String err = "";
    bool created = false;

    System.Security.SecureString securePass = new System.Security.SecureString();
    password.ToCharArray().ToList().ForEach(p => securePass.AppendChar(p));
    using (Runspace runspace = RunspaceFactory.CreateRunspace(connectionInfo))
    {
        Pipeline pl = runspace.CreatePipeline();
        try
        {
            runspace.Open();
            Command newMailbox = new Command("New-Mailbox");
            newMailbox.Parameters.Add("Alias", alias);
            newMailbox.Parameters.Add("Name", name);
            newMailbox.Parameters.Add("FirstName", firstname);
            newMailbox.Parameters.Add("LastName", lastname);
            newMailbox.Parameters.Add("UserPrincipalName", userprincipalname);
            newMailbox.Parameters.Add("Password", securePass);
            newMailbox.Parameters.Add("Organization", organization);

            pl.Commands.Add(newMailbox);
            var results = pl.Invoke();

            if (pl.Error != null && pl.Error.Count > 0)
```

Ajout d'une boîte mail web service 1

Le web service propose au total 15 méthodes, chacune exécutant des actions différentes et représentant l'ensemble des fonctionnalités disponibles pour les boîtes mails exchanges :

- **public** WSMANConnectionInfo connectExchange(String admin) : Permet de récupérer une session administrateur
- **public** MailBox[] GetMailBox(String admin) : Récupère les boîtes mails d'une organisation.
- **public** Retour addMailBox(...) : Créé une boîte mail.
- **public** Retour deleteMailBox(String name, String admin) : Supprime une boîte mail.
- **public** String[] getDistributionGroup(String admin) : Récupère les groupes de distribution d'une organisation.
- **public** Member[] getMember(String admin, String distribGroup) : Récupère les membres d'un groupe de distribution.
- **public** Retour createDistributionGroup(String admin, String name) : Créé un groupe de distribution.
- **public** Retour deleteDistributionGroup(String admin, String name) : Supprime un groupe de distribution.
- **public** Retour deleteDistributionGroupMember(String admin, String member, String group) : Supprime un membre du groupe de distribution passé en paramètre.
- **public** Retour addDistributionGroupMember(String admin, String member, String group) : Ajoute un membre à un groupe de distribution.
- **public** Retour addContact(...) : Ajoute un contact externe.
- **public** Member[] getContact(String admin) : Récupère les contacts externes d'une organisation.
- **public** Retour deleteContact(String admin, String contact) : Supprime un contact externe d'une organisation.
- **public** Retour changePassword(String admin, String name, String password) : Change le mot de passe d'une boîte mail.
- **public** Details getMailBoxDetails(String admin, String name) : Récupère des détails concernant une boîte mail.

2.2.7 Boites mails Icewarp

Icewarp, anciennement Merak Mail Server, est un serveur de communication unifié. Il comporte les composants pour toute forme de communication, que ce soit des services mails, chat, voix par adresse IP, SMS, échanges de fichiers ou encore des réunions en ligne avec partage d'écran.

La gestion des boites mails Icewarp diffère totalement de celle des boites mails Exchange. Les méthodes permettant d'interagir avec le serveur Icewarp se font elles aussi par l'intermédiaire d'un web service développé en WinDev, un langage de programmation développé par une société française privée. La différence réside dans la façon de communiquer avec ces méthodes, SOAP n'étant pas compatible avec celle-ci.

```
monTag=merak_cree_APIObject("Account")

SI monTag <> "" AND monTag<>False ALORS
    //creation du nouveau compte
    cmd est une chaîne="<methodCall><methodName>"+monTag+"-
    &gt;New</methodName><params><param><value>"+mesInfos:monAlias+"@"+mesInfos:monDomaine+"</value></param></params></methodCall>"

    merak_rpc_query(cmd)

    merak_set_property(monTag,"U_Type","4")

    merak_set_property(monTag,"R_Alias",mesInfos:monAlias)

    merak_set_property(monTag,"R_Name",mesInfos:monAlias)

    merak_set_property(monTag,"R_ActivityValue",mesInfos:mesForwards)

    merak_save(monTag)
FIN
```

Création alias windev 1

Chaque objet Icewarp est considéré comme une boite mail. Afin de les différencier, ces boites mails possèdent un type. Dans l'exemple de création d'alias plus haut, le type de la boite est « 4 ». Elle ne comporte ni mot de passe, ni tous les autres paramètres d'une boite mail classique, hormis le nom et l'alias. Une boite mail classique sera de type « 1 ».

Afin de communiquer avec les fonctions WinDev, le serveur utilise un client XML-RPC. Celui-ci permet d'appeler une fonction sur un serveur distant à partir de n'importe quel système et avec n'importe quel langage de programmation. Le serveur est lui-même sur n'importe quel système et est programmé dans n'importe quel langage. La fonction « createClientRPC » permet d'instancier la connexion avec le web service d'Icewarp.

```
// creation du client pour communication webservice Icewarp via XML-RPC
function createClientRPC()
{
    $server_path      = "/rpc/";
    $server_hostname  = "mailssl.fc-net.fr";
    $server_port      = 80;
    // Création du client
    $client = new xmlrpc_client($server_path, $server_hostname, $server_port);
    // $client->setDebug(1); // activation mode debug
    $client->setCredentials("fcnetclient@melo.fc-net.fr", "*****");
    return $client;
}
```

Client XML RPC 1

Chaque méthode s'exécute de la manière suivante :

1. Instanciation du client XML-RPC

```
$client = $this->createClientRPC();
```

2. Renseignement du type d'objet et de la méthode

```
$methodName = "Delete";
$msg = new xmlrpcmsg($methodName, array(new xmlrpcval("IceWarpServer.AccountObject")));
```

3. Appel de la méthode

```
$resp = $client->send($msg);
```

4. Instanciation d'un paramètre

```
$msgType = new xmlrpcmsg($methodName, array(new xmlrpcval("monAlias"), new xmlrpcval($nom_Boite)));
```

5. Envoi du paramètre

```
$respType = $client->send($msgType);
```

6. Sauvegarde de la méthode

```
$methodName = $tag . "->Save";
$msgSave = new xmlrpcmsg($methodName, array());
```

L'instanciation ainsi que l'envoi de paramètre doit être pour chaque paramètre envoyé à la fonction.

Il existe deux types d'objets :

- Les Account Objects
- Les Domains Objects

Les accounts objects permettent d'agir sur une boîte mail précisée lors de l'appel de la méthode. Les domains objects eux, permettent d'agir directement sur le domaine en question et non sur une boîte. Ils sont utilisés lors de la création d'une boîte mail, par exemple.

3 Back Office

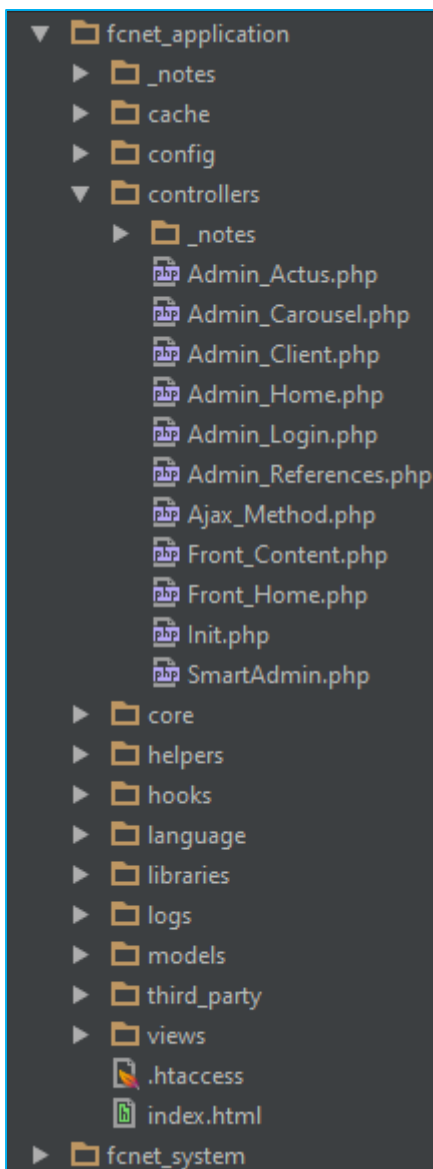
3.1 Généralités

3.1.1 Structure du back office

Le back-office est développé sous le framework CodeIgniter. Il possède donc une architecture MVC complète. Il intègre aussi la template SmartAdmin, sous laquelle il est développé.

Le back-office ne comprend aussi la maquette du site d'accueil de FCnet. Les deux sont donc développés en parallèle sous la même structure. Chaque classe correspondant à l'administrateur est composé de « Admin_ » suivi du nom de la class. Cela permet donc de différencier les class Admin des class Fronts.

Chaque controller hérite de la classe SmartAdmin qui permet la liaison entre le layout et le controller et qui fonctionne du même principe que celle présente dans l'espace client.



Architecture back-office 1

```
function index()
{
    $layout = new stdClass();
    $data = new stdClass();

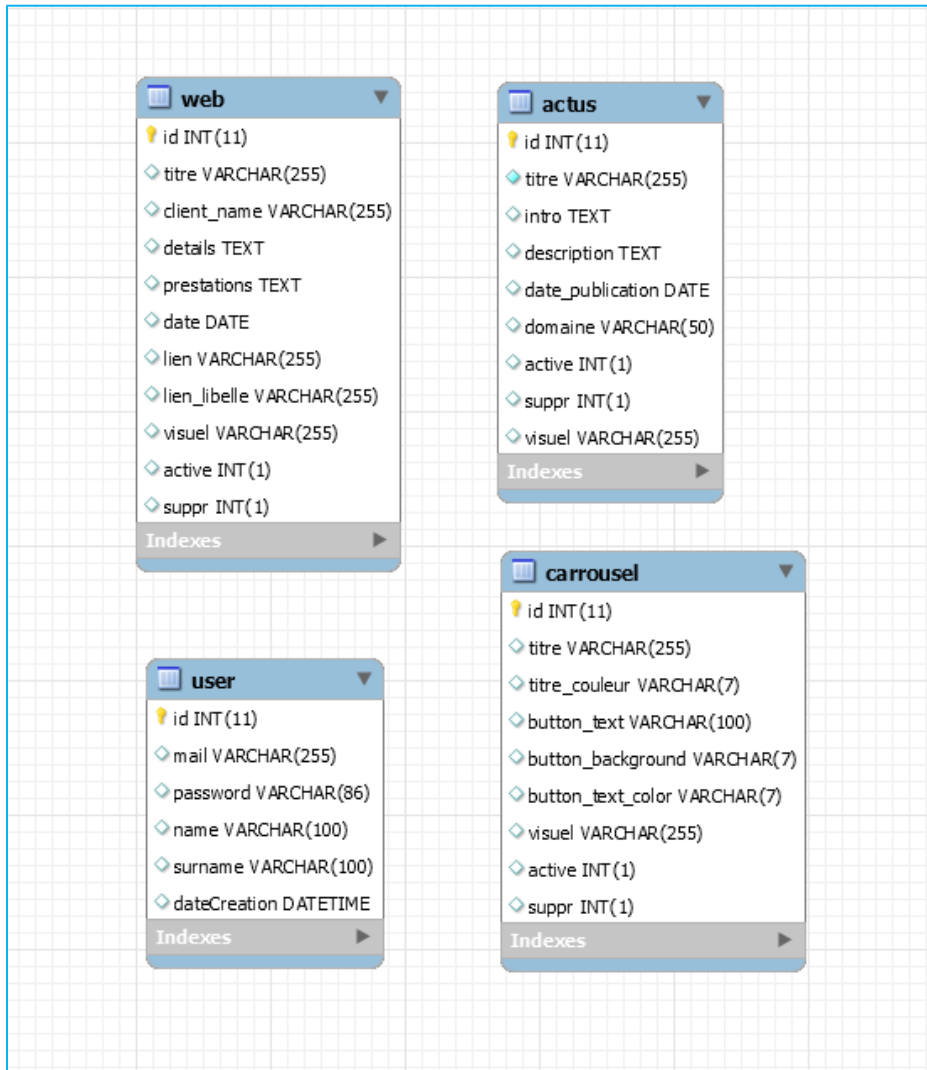
    // Layout
    $layout->page_title = "Homepage";
    $layout->page_short = "Page d'accueil";
    $layout->page_nav = $this->config('PageNav');
    $layout->breadcrumbs = $this->config('Breadcrumbs');

    $layout->main_content = $this->load->view('admin/pages/home/home', $data, TRUE);
    $this->render('admin/layout', $layout);
}
```

Exemple d'utilisation de SmartAdmin 1

3.1.2 Base de données

La base de données du back-office comporte de quoi stocker les utilisateurs y ayant accès ainsi que les diverses informations que doivent contenir les modules. Elle est développée en MySQL.



MCD Back office 1

3.2 Fonctionnalités

3.2.1 Système de connexion

Le système de connexion est sécurisé grâce au cryptage des mots de passe en SHA256 avec salt. Il fonctionne de la même manière que celui de l'espace client.

Il comporte aussi une gestion des cookies qui permet à l'utilisateur de ne pas avoir à rentrer ses identifiants à chaque connexion. Ces paramètres sont vérifiés dans le constructeur du model « User » qui est chargé par défaut. Cela permet donc de rediriger l'utilisateur en cas de « non-connexion ».

```
public function __construct()
{
    parent::__construct();
    if ( !isset( $_SESSION ) )
        session_start();

    public function connect(){
        if ( isset($_POST) && !empty( $_POST ) )
        {
            if ( isset($_POST['mail']) && isset($_POST['password']) )
            {
                // Si un email et un password sont spécifiés
                // On vérifie si l'utilisateur peut se connecter
                $_response = $this->User->_connect( $_POST['mail'], $_POST['password'] );
                if ( $_response )
                {
                    // Si oui, on vérifie l'insertion de cookie et on connecte l'utilisateur
                    if ( isset($_POST['remember']) && $_POST['remember'] == 'on' )
                    {
                        $this->User->createCookie();
                    }
                }
                else
                {
                    $_SESSION['info'] = 'Adresse mail ou mot de passe incorrect !';
                }
            }
            else
            {
                $_SESSION['info'] = 'Les champs sont vides !';
            }
            redirect($this->urlAdmin);
        }
    }
}
```

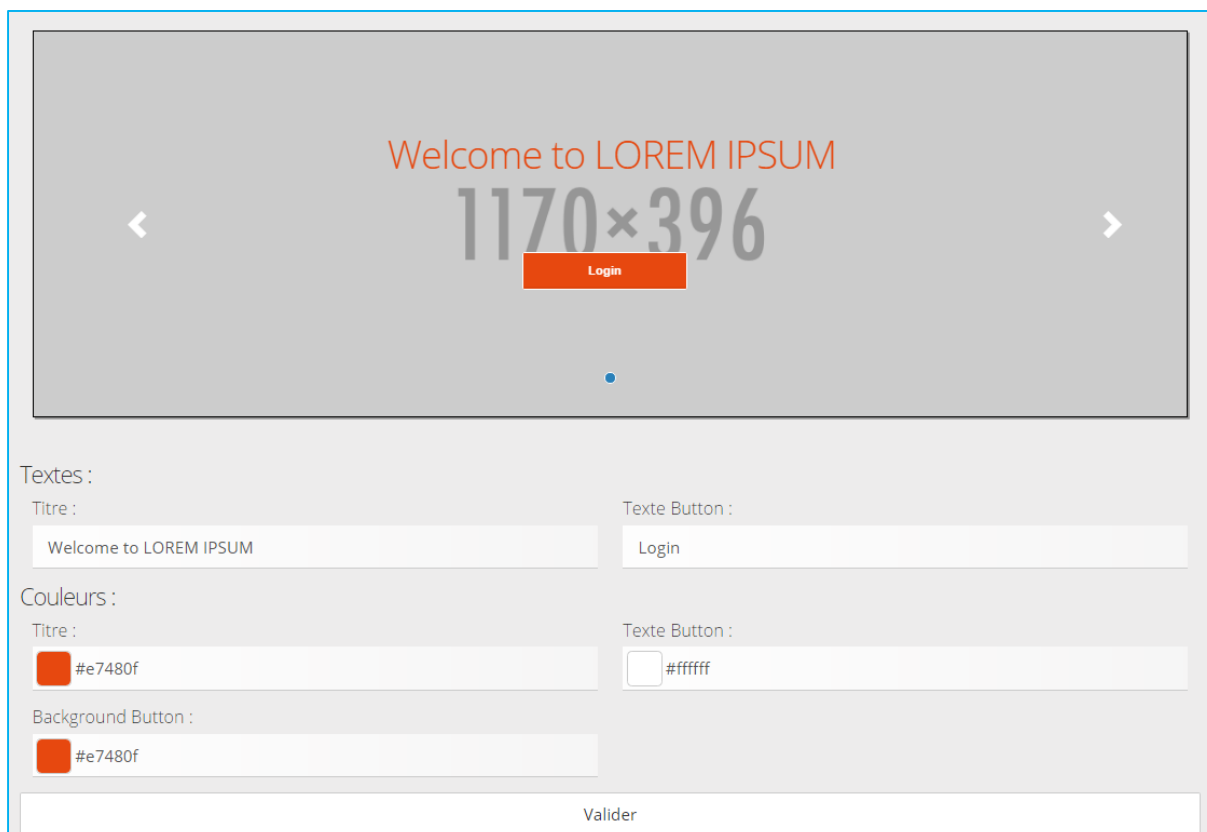
Fonction connect du back office 1

Constructeur de User 1

La gestion de la connexion se fait par l'intermédiaire du controller « Admin_login » et du model User. Elle possède une méthode connect() qui sert à connecter l'utilisateur en cas d'envoi de données et qui gère aussi l'utilisation ou non de cookies.

3.2.2 Gestion du carrousel

Le back-office possède une gestion des slides présents dans le carrousel du site d'accueil de FCnet. Celui-ci est géré par le controller Admin_carousel et par le modèle « A_Carousel ».



Textes :

Titre :

Texte Button :

Couleurs :

Titre :

Texte Button :

Background Button :

Création de slide d'un carrousel 1

Cette fonctionnalité permet une gestion dynamique des couleurs grâce à un « color-picker » issu des plug-ins « bootstrap colorpicker » et « mini color ».

Différents problèmes ont été rencontrés lors de l'ajout d'images. En effet, du à un problème de sécurité, le javascript empêche la récupération du chemin relatif d'un fichier lors de son upload. Or, la page étant entièrement dynamique, il fallait une méthode afin de pouvoir afficher l'image directement lors de son upload.

Pour se faire, j'ai mis en place une fonction javascript permettant de récupérer l'image en base64.

```
function readURL(input) {
    if (input.files && input.files[0]) {
        var reader = new FileReader();

        reader.onload = function (e) {
            //Lorsque l'image charge
            $imgUrl = e.target.result;
            //On récupère le résultat en base64 et on l'affiche dans le carousel
            $(".carousel-formImage").fadeOut(400, function() {
                $('.carousel-formImage').attr('src', $imgUrl);
                $("input[name='visuel']").attr("value", $imgUrl);
            }).fadeIn(400);
        };
        reader.readAsDataURL(input.files[0]);
    }
}
```

Fonction readUrl 1

Afin de pouvoir sauvegarder l'image dans un format adapté, j'ai aussi développé une fonction php, appellable en Ajax, permettant de convertir une chaîne de caractère base64 en image de format jpg.

```
function saveImage($path, $name, $dimension, $original)
{
    header("Content-Type: image/jpeg");

    $path = $path."/". $name. ".jpg";

    /*
     * Image 1 : accueil
     */
    // Get new dimensions
    list($width, $height) = getimagesize($path."original.jpg");
    $new_width = 400;
    $new_height = 290;

    // Resample
    $image_p = imagecreatetruecolor($new_width, $new_height);
    $image = imagecreatefromjpeg($path.'original.jpg');
    imagecopyresampled($image_p, $image, 0, 0, 0, 0, $new_width, $new_height, $width, $height);

    // Output
    imagejpeg($image_p, $path.'accueil.jpg');
}
```

fonction saveImage 1

La page de configuration des slides se fait par l'intermédiaire du plugin « data-Table » qui est complètement intégré par Smart Admin. Cette page permet l'activation la suppression ainsi que la visualisation d'un slide et de ses caractéristiques.

ADMIN > MODIFIER LE CAROUSEL

Listing des slides

10 ▾

N°	Image	Titre	Button	Titre C*	Button C*	Button F*	Actif	Supprimé	#
13		Welcome to LOREM IPSUM	Login				<input checked="" type="checkbox"/>	<input type="checkbox"/>	✎ Editier
14		dzzadzad	Test				<input checked="" type="checkbox"/>	<input type="checkbox"/>	✎ Editier
15		Titre du slide !	Chouette bouton plusieurs lignes				<input checked="" type="checkbox"/>	<input type="checkbox"/>	✎ Editier
16		Welcome to LOREM IPSUM	Login				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	✎ Editier

Showing 1 to 4 of 4 entries

[Previous](#)
[1](#)
[Next](#)

Légende :

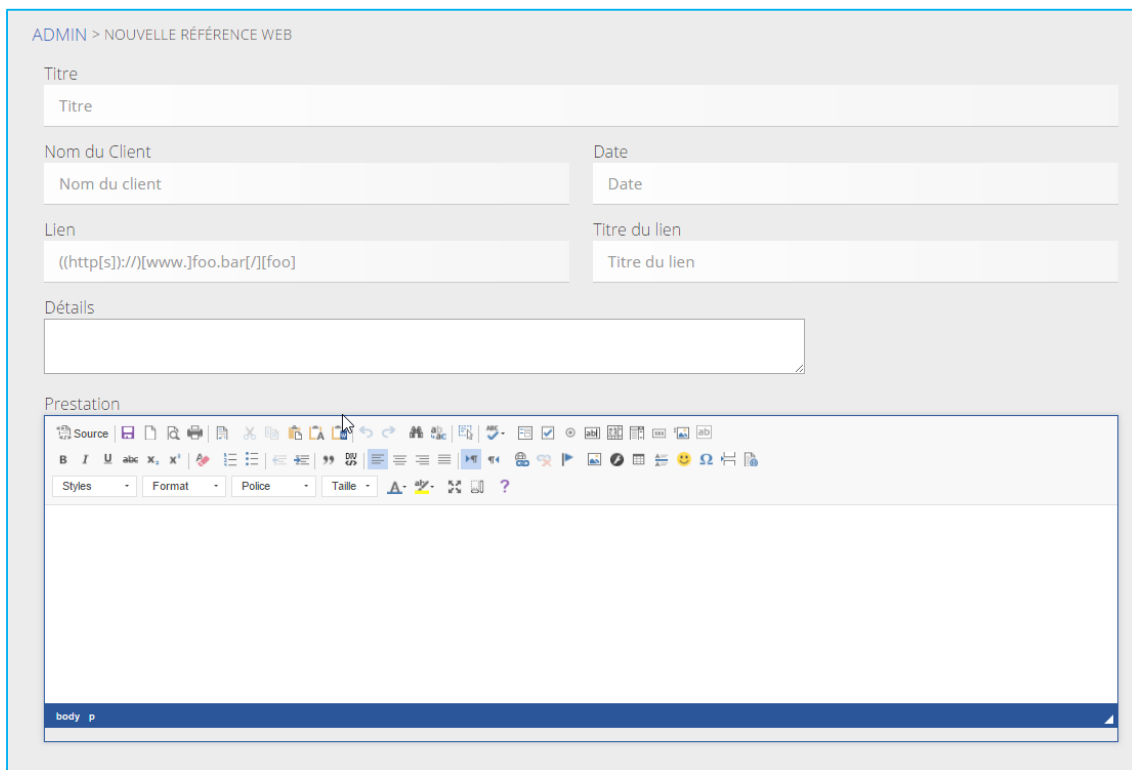
C* : Couleur du texte
 F* : Couleur du fond

Configuration des slides 1

Les images des slides sauvegardées sont récupérées grâce au titre du slide parsé suivi d'un identifiant unique. Seul l'identifiant est sauvegardé en base de données car celui-ci permet amplement de récupérer l'image.

3.2.3 Gestion des références web

Une référence web correspond à une prestation réalisée par FCnet. Elle se compose principalement d'un détail, d'une description, d'un titre, de plusieurs images.



Ajouter une référence web 1

La gestion des images se fait de la même façon que pour la carrousel. La seule différence réside dans la gestion de plusieurs images, dont une principale (L'image de garde). La méthode Ajax permettant la sauvegarde d'image est donc appelée plusieurs fois afin de pouvoir toutes les sauvegarder. Une méthode spécifique est appelée pour sauvegarder l'image principale.

L'écriture de la prestation est améliorée par le plugin CKeditor avec un thème semblable à celui de Microsoft Office. Son instance se fait, en javascript, comme ci-dessous :

```
CKEDITOR.replace('prestations', {allowedContent: true});
```

Instance de CKeditor 1

CKeditor possède un fichier de configuration dans lequel on peut définir différents paramètres

```
CKEDITOR.editorConfig = function( config ) {  
    config.language = 'fr';  
    config.skin = 'office2013';  
};
```













Configuration de ckeditor 1

La gestion des actualités se fait elle aussi par l'intermédiaire du plugin data-Table.

ADMIN > LISTING DES RÉFÉRENCES

Liste des articles

Q 10 ▾

N°	Images	Titre	Client	Date	Lien	Actif	Supprimé	#
19		Référence web dwfsfsf	Nom du client	2015-01-01	Fcnet	<input checked="" type="checkbox"/>	<input type="checkbox"/>	 
20		ref	Nom du client	2015-01-01	fcnet	<input checked="" type="checkbox"/>	<input type="checkbox"/>	 
22		zadzaddza	Nom du client	2015-01-01	fcnet	<input checked="" type="checkbox"/>	<input type="checkbox"/>	 
23		Test depuis un autre poste	test	2016-05-03	fcnet	<input checked="" type="checkbox"/>	<input type="checkbox"/>	 

Showing 1 to 4 of 4 entries

Previous **1** Next

Configuration des références web 1

3.2.4 Gestion des actualités

Une actualité est composée d'un titre, d'une image, d'un domaine, d'une date, d'une introduction et d'un corps de texte. La créations de celles-ci fonctionne de la même manière que le reste des modules. Le confort est assuré grâce au plugin CKeditor et le plugin datePicker pour les dates.

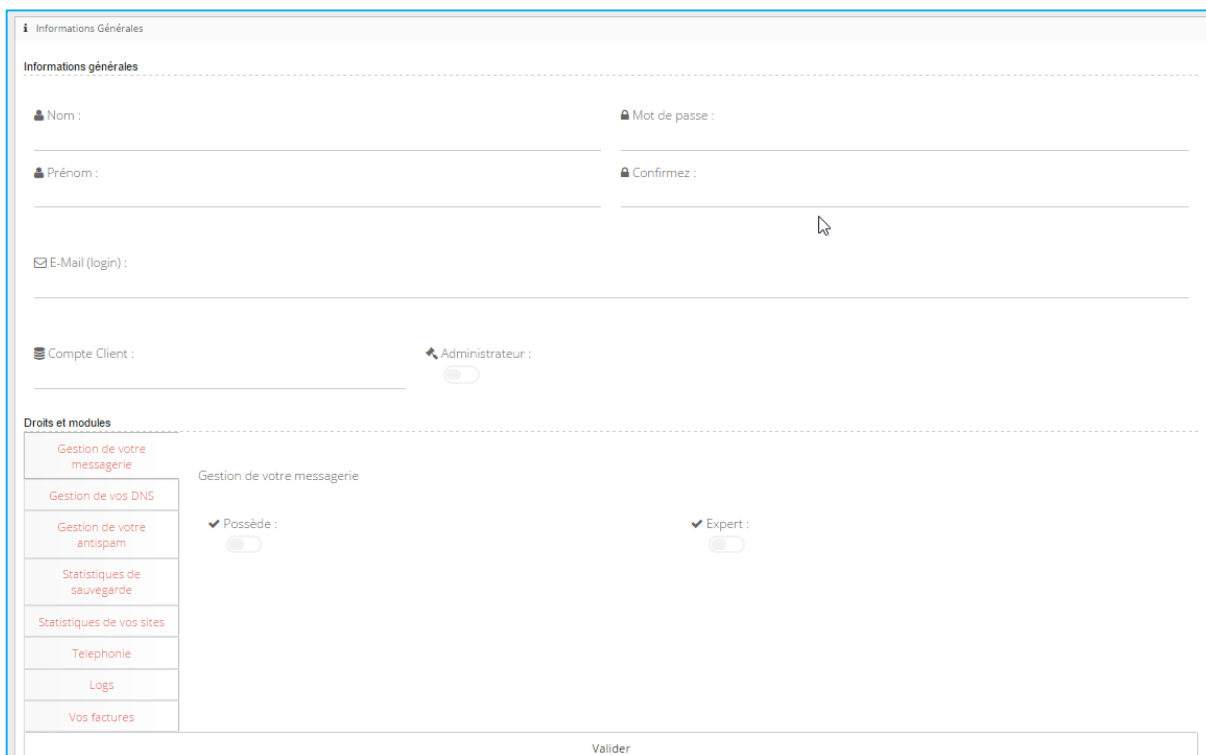
L'ajout d'image se fait lui aussi par l'intermédiaire de la fonction save_image.

3.2.5 Gestion de l'espace client

La gestion de l'espace client comporte différents modules. Nous ne verrons dans ce rapport que les détails techniques concernant la création d'un client ainsi la gestion des droits.

Les autres modules importants présents sont , premièrement, la gestion des clients, qui fonctionne de la même manière que la gestion des références webs, des slides et des actualités et secondement, l'affichage des logs, qui fonctionne exactement de la même manière que le module de l'espace client.

La création d'un client peut se faire par le biais d'une demande d'accès ou par le biais d'une création manuelle. Nous verrons tout d'abord la création manuelle.



Page de création d'un client 1

La page ci-dessus permet la création manuelle d'un client. Elle est composée, dans la partie supérieure, d'un form basique, et dans la partie inférieure, de la partie permettant l'attribution de droits. La difficulté résidant dans l'attribution de droits. Elle est composée de checkbox. A l'ajout d'un utilisateur, il faut donc vérifier chaque checkbox et attribuer le droit en fonction de sa valeur.

La création automatique de client est possible par le biais d'une demande d'accès.

Liste des demandes non traitées

Q 10 ▼

Demande N°	Client N°	Mail	Mot de passe	Nom	Prenom	Société	fonction	Actions
21	123	Teeest@test.test	abcd	abcd	abcd	abcd	abcd	+ Créer ✎ Editer
22	1234	test_demande@fcnet.fr	tcacdb	toto	tutu	fcnet	titi	+ Créer ✎ Editer
23	1234	test_demande@fcnet.fr	tcacdb	toto	titi	fcnet	tutu	+ Créer ✎ Editer
24	123	contact@gaelhuot.fr	abc	HUOT	Gael	abc	abc	+ Créer ✎ Editer
25	441	contact@gaelhuot.fr	abc	HUOT	Gael	abc	abc	+ Créer ✎ Editer
26	123	ghuot90@gmail.com	tcacdb	HUOT	Gael	fcnet	toto	+ Créer ✎ Editer

Showing 1 to 6 of 6 entries Previous 1 Next

Demande accès 1

L'action « créer » va créer le client automatiquement avec les données stockées. L'action « éditer » va, quant à elle, permettre l'édition des données rentrées depuis la page de création manuelle d'un client. Les données sont envoyés depuis un formulaire invisible créé au préalable en javascript et sont traitées depuis la page de création.

La gestion des droits des clients se faire depuis la page « gestion des droits ». Elle comporte un tableau de liens. Ces liens ont été affichés sous forme de checkbox afin de permettre un meilleur confort.

Liste des droits (En bleu : **Possède** / En vert : **Expert**)

Q 10 ▼

Client	Gestion de votre messagerie	Gestion de vos DNS	Gestion de votre antispam	Statistiques de sauvegarde	Statistiques de vos sites	Telephonie	Logs	Vos factures
tsm@fcnet.fr	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> <input type="checkbox"/>	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>
test@maty.fr	<input type="checkbox"/> <input type="checkbox"/>	<input checked="" type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>
jobard@fcnet.fr	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> <input type="checkbox"/>	<input checked="" type="checkbox"/> <input type="checkbox"/>	<input checked="" type="checkbox"/> <input type="checkbox"/>	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> <input type="checkbox"/>	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> <input type="checkbox"/>
testlog@maty.fr	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> <input type="checkbox"/>	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>

Configuration des droits. 1

Les liens permettent d'ajouter ou de supprimer un droit selon la valeur par défaut. Cela se fait grâce à la méthode « setDroit ». Elle prend en paramètre l'id du client, l'id du module ainsi que le type (« expert » ou « classique ») et va switcher la valeur de la possession dans la base de données. Elle redirige ensuite directement sur la page précédente.

```
function setDroit($idClient, $idModule, $type){  
    switch ($type)  
    {  
        case ("add") :  
            $this->Client->addDroit($idClient, $idModule);  
            break;  
        case ("remove") :  
            $this->Client->removeDroit($idClient, $idModule);  
            break;  
        case ("expert") :  
            $this->Client->setExpert($idClient, $idModule);  
            break;  
        case ("removeExp") :  
            $this->Client->removeExpert($idClient, $idModule);  
            break;  
        default :  
            break;  
    }  
    redirect(site_url("Admin/Client/droits"));  
}
```

fonction set Droit 1

4 Table des illustrations

Exemple de configuration de navbar 1	4
Fichier htaccess 1	5
Kernel de l'application 1	6
Fichier index.php 1	6
MCD Espace client 1	7
Design de l'espace client 1	8
Exemple de css 1	8
Class model 1	9
Constructeur de SmartAdmin 1	9
Exemple d'utilisation de smartadmin 1	10
Méthode load de Controller 1	10
Liaison entre les différentes class 2	11
Cryptage sha256 1	11
Tableau de gestion des DNS 1	12
Tableau de paramétrage des DNS 1	13
Récupération de boîtes depuis un client SOAP 1	15
Récupération administrateur exchange 1	15
Connexion Exchange depuis le Web Service 1	16
Exemple d'utilisation de PowerShell 1	16
Ajout d'une boîte mail web service 1	16
Récupération des boîtes mails web service 1	16
Création alias windev 1	18
Client XML RPC 1	19
Architecture back-office 1	21
Exemple d'utilisation de SmartAdmin 1	21
MCD Back office 1	22
Constructeur de User 1	23
Fonction connect du back office 1	23
Création de slide d'un carrousel 1	24
Fonction readUrl 1	25
fonction saveImage 1	25
Configuration des slides 1	26
Ajouter une référence web 1	27
Instance de CKeditor 1	27
Configuration de ckeditor 1	27
Configuration des références web 1	28
Page de création d'un client 1	29
Demande accès 1	30
Configuration des droits. 1	30
fonction set Droit 1	31