



PROJET mini LW1 : CMS

**TIDAF JUBA
MEDDAH AMINE**

2016/2017

Table des matières

I.	Introduction	1
II.	Analyse.....	4
1.	Identification des acteurs et leurs rôles	4
2.	Diagramme des séquences :.....	6
2.1.	Cas d'utilisation consulter les articles	6
2.2.	Cas d'utilisation consulter les articles.....	7
2.3.	Cas d'utilisation ajouter un rédacteur	8
2.4.	Cas d'utilisation ajouter un article	9
2.5.	Cas d'utilisation supprimer un rédacteur	10
III.	Conception.....	11
1.	Langages et choix des outils utilisés	11
▪	HTML/CSS	11
▪	PHP/MySQL	12
▪	JavaScript	12
▪	WampServer	13
2.	Base de données	13
3.	Les différentes étapes du développement	14
A.	LA PROGRAMMATION	14
B.	Le Design	22
4.	Résultats obtenus.....	23
5.	Problèmes Rencontré.....	24
6.	Résultats obtenus.....	24
-	LA PROGRAMMATION	24
-	Le Design	24
	Conclusion	24

Table des Figures

Figure 1 : Tableau spécification des tâches.....	4
Figure 2 : diagramme de cas d'utilisation générale	5
Figure 3 : Diagramme de séquence Consulter les articles	7
Figure 4 : Diagramme de séquence s'authentifier.	8
Figure 5 : Diagramme de séquence ajouter un rédacteur	9
Figure 6 : Diagramme de séquence ajouter un article	10
Figure 7 : Diagramme de séquence supprimer un rédacteur	11
Figure 8 : image de l'éditeur de texte.	14
Figure 9 : l'instruction de composite.	15
Figure 10 : une section.	16
Figure 11 : une sous-section.	16
Figure 12 : le prompt qui oblige la saisie d'un titre.	17
Figure 13 : la génération du sommaire.	17
Figure 14 : lancement de la page « max.html ».	19
Figure 15 : le résultat de test sur « max.html »	20
Figure 16 : la structure MVC de site.....	21

I. Introduction

Dans le cadre de ce mini-projet « réalisation d'un min CMS », nous avons entamé sa réalisation en suivant un plan que on a tracer pour bien se synchroniser entre binôme.

Ce rapport comporte les éléments suivants :

- la partie Analyse : dans cette partie on a étudié les cas d'utilisations et les besoin de site, et les probables scénario dans le système complet.
- La conception et la réalisation du projet : nous évoquerons dans cette partie le choix des outils et des langages utilisés pour la réalisation du site, la base de données créée pour stocker les différentes informations, du développement proprement dit le codage, le design du site.
- Enfin nous parlerons aussi des résultats obtenus, des problèmes rencontrés lors de la conception.

Dans les lignes qui suivent nous allons détailler le développement de chacune de ces parties en étant le plus précis possible pour que le lecteur de ce rapport soit éclairé sur les différentes étapes de ce travail qui demande beaucoup de temps et qui tout de même est vraiment passionnant.

II. ANALYSE

Cette activité commence par l'étude des cas d'utilisations et de leurs scénarios ainsi que les besoins fonctionnels du système (ce que le système doit faire en réponse à une requête d'un utilisateur).

1. Identification des acteurs et leurs rôles :

Acteur	Tâches
Visiteur	T01 - se connecter au site. T02 - consulter les articles.
Rédacteur	T03 - s'authentifier. T04 - idem que les tâches du visiteur. T05 - insérer un article. T06 - modifier ou supprimer un article.
Administrateur (du site)	T03 - T07 - idem que les tâches du visiteur. T08 - gérer les rédacteurs (insérer ,supprimer, modifier). T09 - gestion des catégories.

Figure 1 : Tableau spécification des tâches.

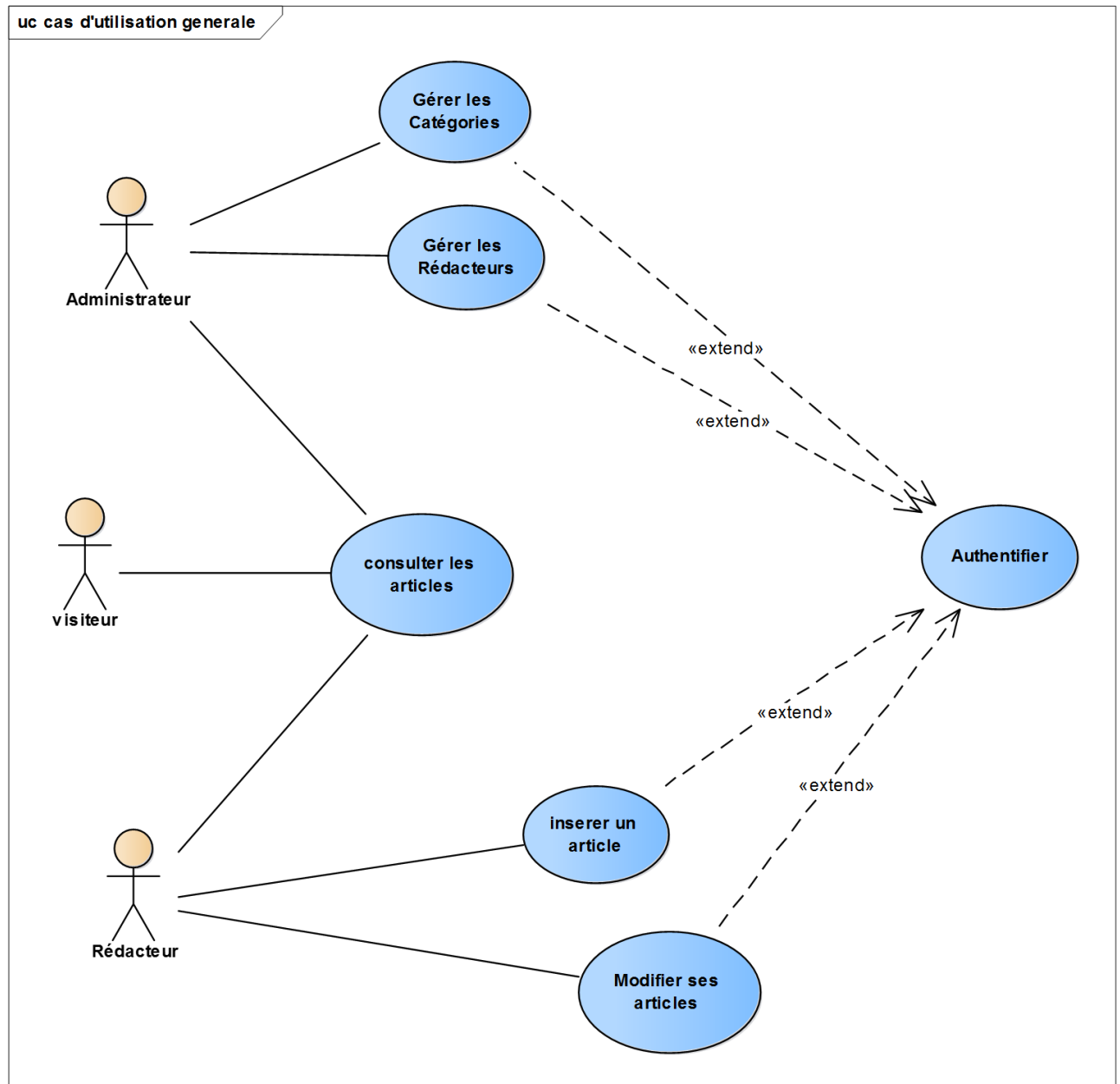


Figure 2 : diagramme de cas d'utilisation générale.

Acteur	Tâches	Scénarios
Visiteur	T01 - se connecter au site. T02 - consulter les articles.	S0 - Taper l'URL S01 - Sélectionné le lien des articles qu'il voudra chercher par catégorie. S02 - Avoir une liste d'article qu'il voudra.
Rédacteur	T03 - s'authentifier. T04 - idem que les tâches du visiteur.	S0-S02 S03 - Saisir le login et le mot. S04 - S'authentifier.

	T05- insérer un article. T06- modifier ou supprimer un article.	S05- Sélectionner l'onglet Ajout un article. S06- une page d'un article à remplir sera afficher. S07- consulter ses articles. S08- modifier ou supprimer un article. S09- Valider le changement.
Administrateur (du site)	T03- T07- idem que les tâches du visiteur. T08- gérer les rédacteurs (insérer, supprimer, modifier). T09- gestion des catégories.	S0-S02- S10- Ajouter un rédacteur. S11- remplir le formulaire. S12- ajouter une catégorie. S13- remplir le formulaire. S14- consulter les rédacteurs. S15- une liste de rédacteurs s'affiche, il peut supprimer ou modifier un rédacteur puis il valide le changement. S16- consulter la liste des catégories. S17- une liste de catégories s'affiche, il peut supprimer ou modifier une catégorie puis il valide le changement.

2. Diagramme des séquences :

2.1. Cas d'utilisation consulter les articles :

Nom :	Ajouter dans la mosaïque	
Acteurs concernés	Administrateur, Rédacteur, Visiteur	
Description	Consulter les articles existant	
Préconditions		
Evénements déclenchants		
Conditions d'arrêt	Afficher l'article voulu	
Description du flot d'événements principal :		
Acteur(s)		Système

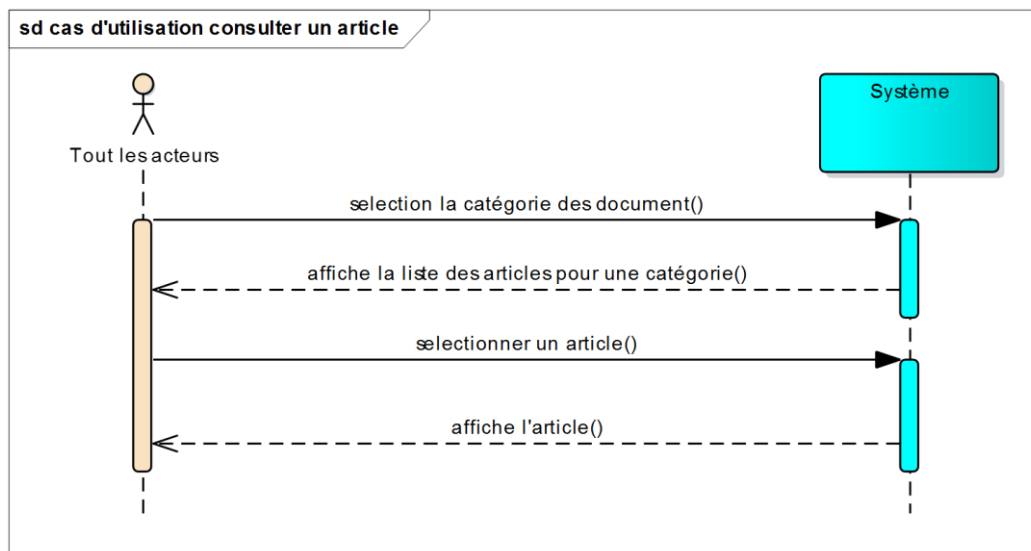


FIGURE 3 : DIAGRAMME DE SEQUENCE CONSULTER LES ARTICLES

l'utilisateur se connecte sur le site, la consultation des articles existant se fait par catégorie

2.2. Cas d'utilisation authentifié :

Nom :	<i>Ajouter dans la mosaïque</i>
Acteurs concernés	Administrateur, Rédacteur
Description	Consulter les articles existant
Préconditions	Avoir un compte
Événements déclenchants	
Conditions d'arrêt	Afficher l'article voulu
<i>Description du flot d'événements principal :</i>	
Acteur(s)	Système

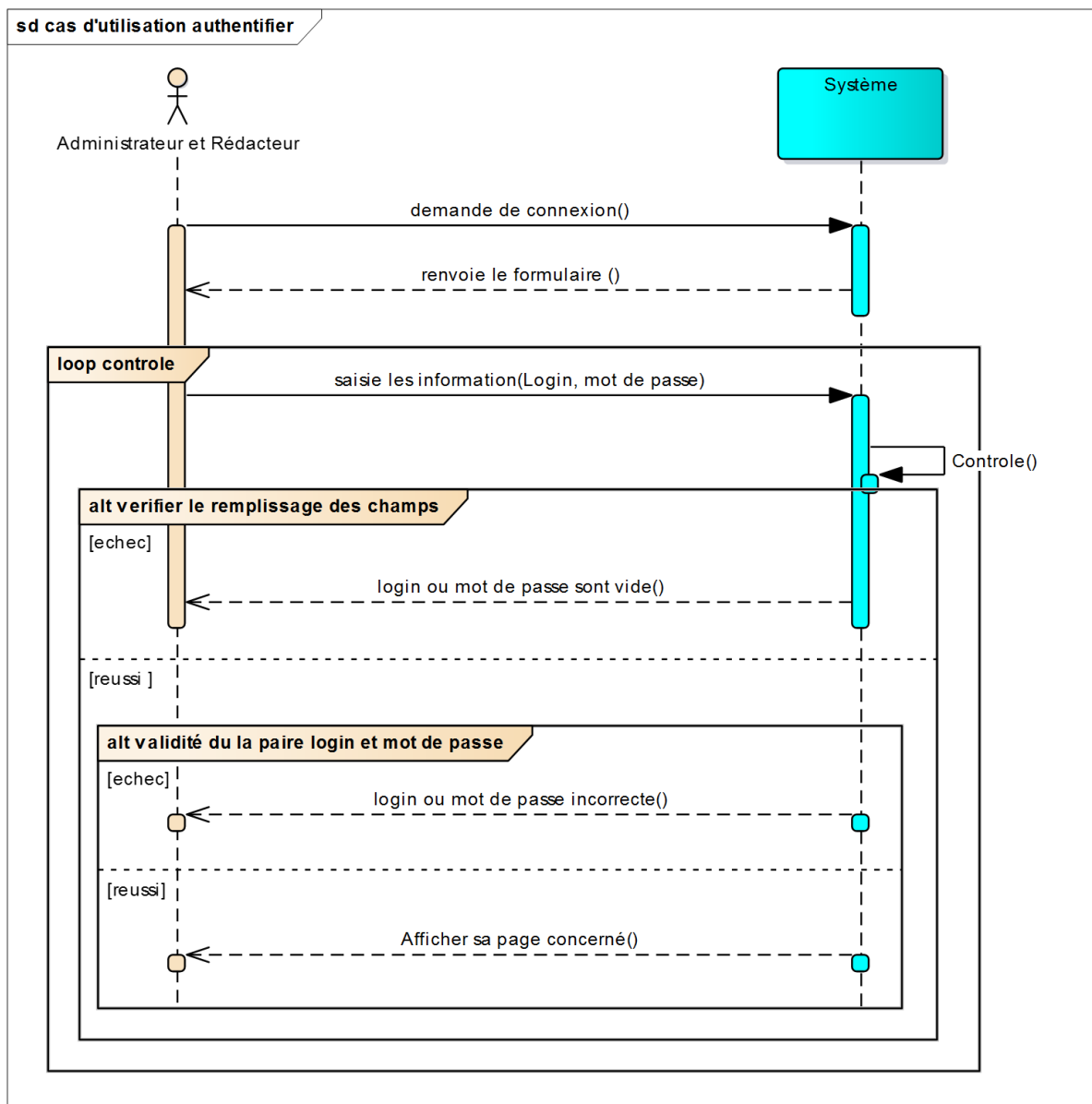


FIGURE 4 : DIAGRAMME DE SEQUENCE S'AUTHTIFIER.

*L'administrateur et le rédacteur se connecte à leurs espace via leurs login et mot de passe, vérification des champs renseignés, validité des identifiants (login, mot de passe).
L'administrateur et le rédacteur accède a leurs espace selon leurs profil.*

2.3. Cas d'utilisation ajouter un rédacteur

Nom :	Ajouter dans la mosaïque
Acteurs concernés	Administrateur

Description	Ajouter un nouveau rédacteur
Préconditions	S'authentifier
Evénements déclenchants	
Conditions d'arrêt	Ajout du rédacteur
Description du flot d'événements principal :	
Acteur(s)	Système

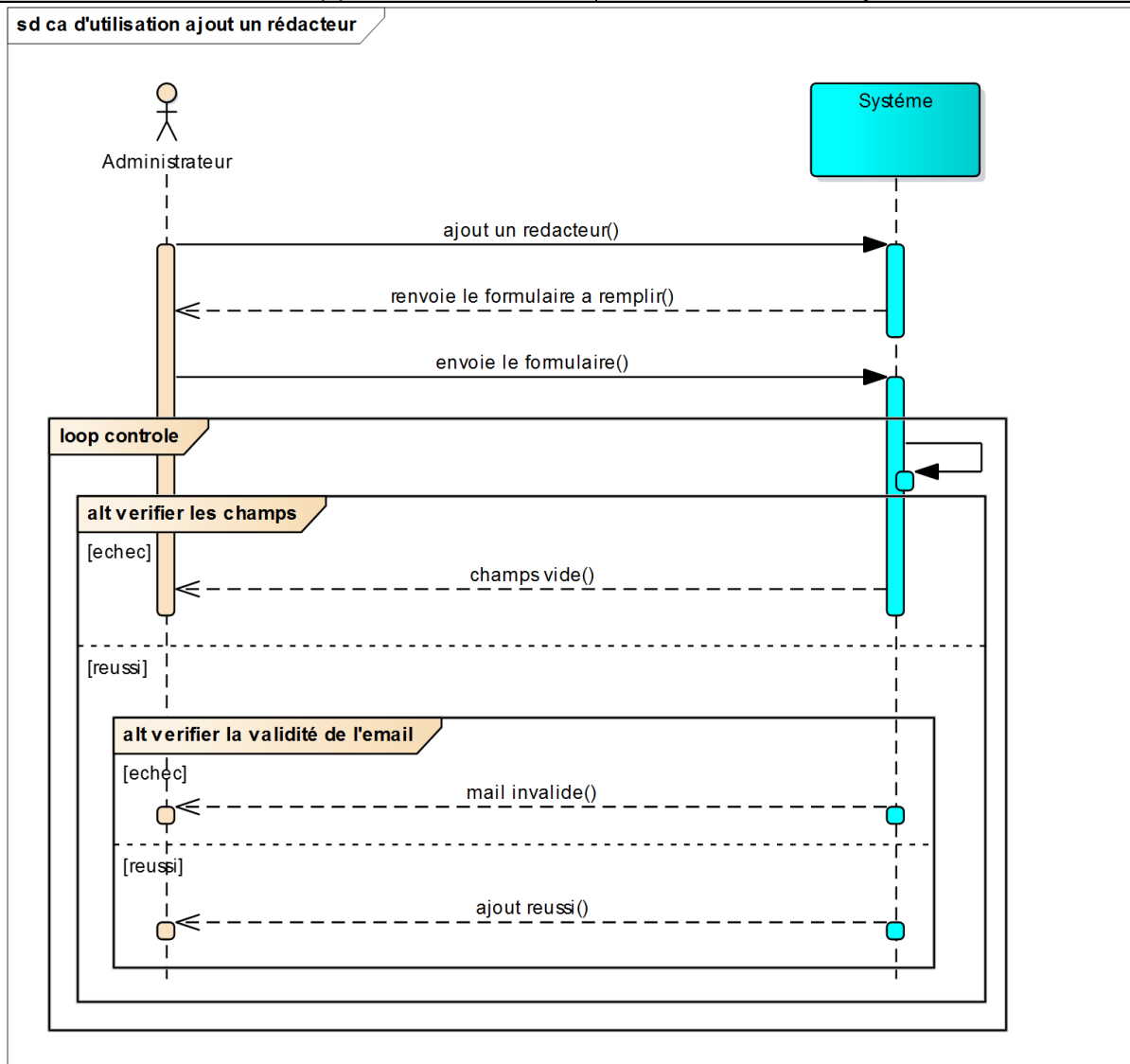


FIGURE 5 : DIAGRAMME DE SEQUENCE AJOUTER UN REDACTEUR.

L'administrateur se connecte sur son compte, il demande au système d'ajouter un nouveau rédacteur. Le système lui fournit un formulaire à remplir, le système ajoute le nouveau rédacteur si tous les champs sont bien remplis, il vérifie la validité de l'email.

2.4. Cas d'utilisation ajouter un article

Nom :	<i>Ajouter dans la mosaïque</i>
Acteurs concernés	Rédacteur
Description	Ajouter un nouveau article
Préconditions	S'authentifier
Evénements déclenchants	
Conditions d'arrêt	Ajouter un article

Description du flot d'événements principal :

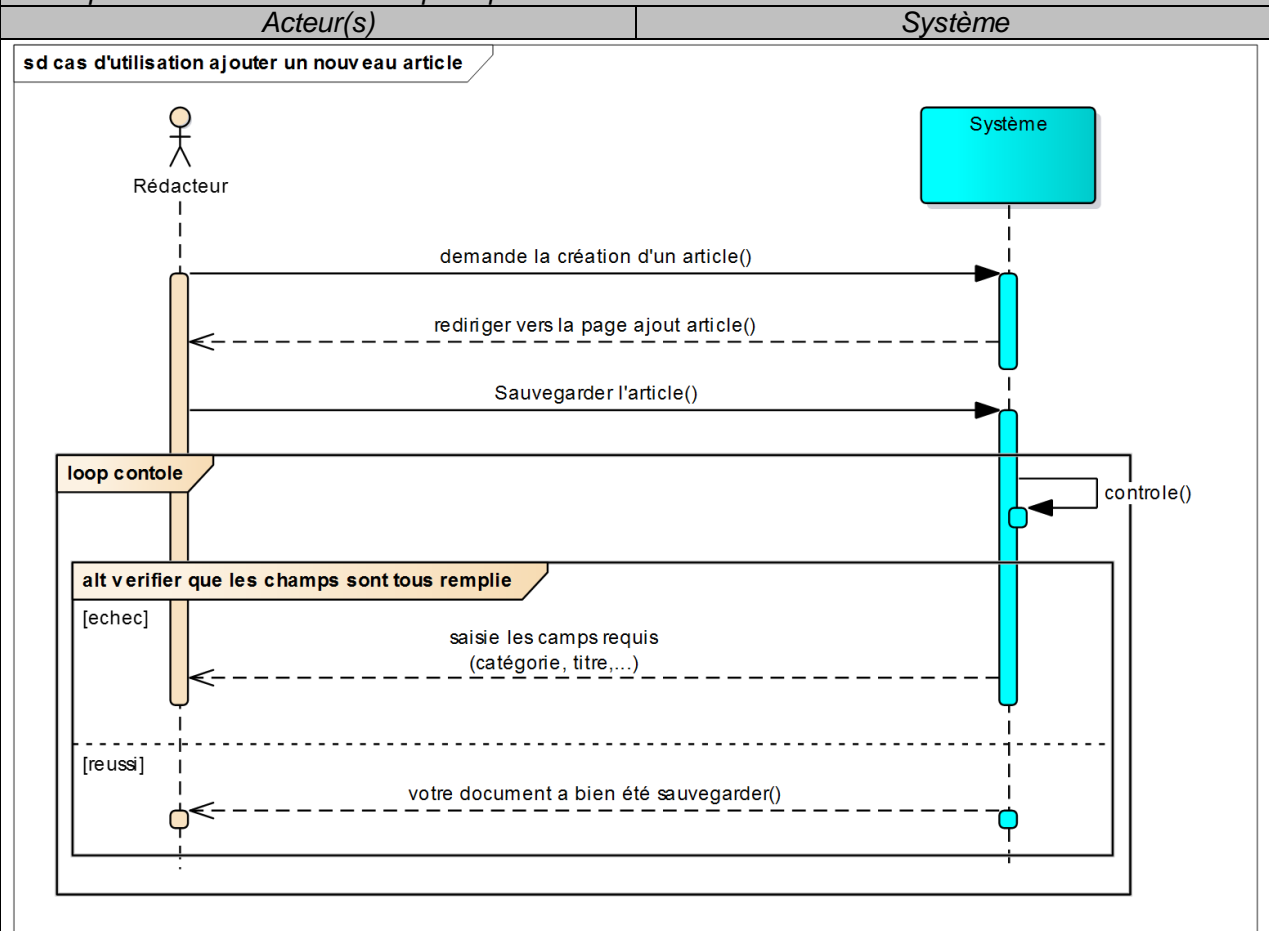
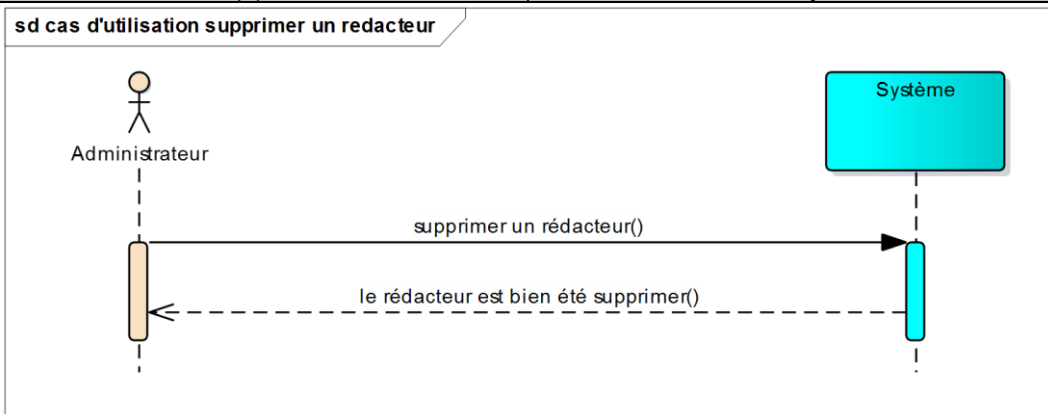


FIGURE 6 : DIAGRAMME DE SEQUENCE AJOUTER UN ARTICLE.

Le rédacteur se connecte sur son compte il demande au système de créer un article, le système lui fournit la page, quand il sauvegarde l'article le système vérifie si tous les champs (catégorie, titre document, titre section...) sont bien remplie.

2.5. Cas d'utilisation supprimer un rédacteur

Nom :	<i>Ajouter dans la mosaïque</i>
-------	---------------------------------

Acteurs concernés	Administrateur
Description	Supprimer un Rédacteur
Préconditions	S'authentifier
Evénements déclenchants	
Conditions d'arrêt	Rédacteur Supprimer
Description du flot d'événements principal :	
Acteur(s)	Système
<p>sd cas d'utilisation supprimer un redacteur</p>  <pre> sequenceDiagram actor Admin as Administrateur participant System as Système Admin->>System: supprimer un rédacteur() activate System System-->>Admin: le rédacteur est bien été supprimer() deactivate System </pre>	
<p><i>L'administrateur sélectionne le rédacteur a supprimer, le système exécute la requête en supprimant le compte rédacteur avec ses articles.</i></p>	

III. CONCEPTION

Dans cette partie, nous allons détailler les étapes du développement du site, les langages et le choix des outils dans un premier temps, puis un descriptif de la base de données, les étapes de la programmation avec son design, un aperçu du résultat obtenu et enfin les principales difficultés rencontrées.

1. Langages et choix des outils utilisés

- **HTML/CSS**

Le HTML (« HyperText Mark-Up Language ») est un langage dit de « balisage » ou de « structuration » permettant la conception de pages web avec des balises de formatage. Les balises permettent d'indiquer la façon dont doivent être présentés le document et les liens qu'il établit avec d'autres documents. Le CSS (« Cascading Style

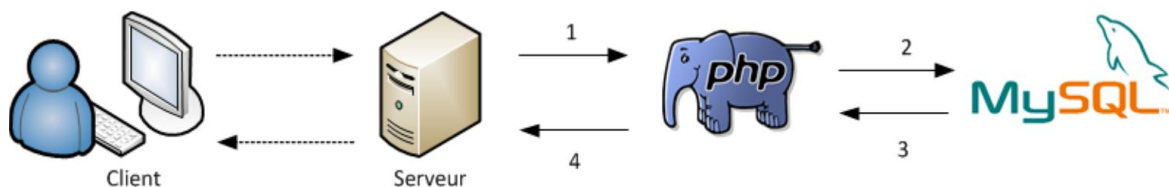
Sheets » : feuilles de style en cascade) est un langage informatique complétant le HTML. Alors que le HTML structure la page Web, le CSS va la mettre en forme en y apportant du style.

▪ PHP/MySQL

Le PHP est un langage de script exécuté du côté serveur (comme les scripts CGI, ASP, ...) et non du côté client (un script écrit en JavaScript ou une applet Java s'exécute sur son ordinateur...). La syntaxe du langage provient de celles du langage C, du Perl et de Java. Ses principaux atouts en font un des langages web le plus utilisé :

- Une grande communauté de développeurs partageant des centaines de milliers d'exemples de script PHP ;
- La gratuité et la disponibilité du code source ;
- La simplicité d'écriture de scripts ;
- La possibilité d'inclure le script PHP au sein d'une page HTML
- L'intégration au sein de nombreux serveurs web (Apache, Microsoft IIS, etc.).

C'est également un langage simple à utiliser avec des bases de données (de nombreux SGBD sont supportés, mais le plus utilisé avec ce langage est MySQL, un SGBD gratuit disponible sur de nombreuses plateformes : Unix, Linux, Windows, MacOS X, ...) ;



▪ JavaScript

Le JavaScript est un langage de script incorporé dans un document HTML. Historiquement il s'agit même du premier langage de script pour le Web. Ce langage est un langage de programmation qui permet d'apporter des améliorations au langage HTML en permettant d'exécuter des commandes du côté client, c'est-à-dire au niveau du navigateur et non du serveur web.

▪ WampServer

« WampServer est une plate-forme de développement Web sous Windows pour des applications Web dynamiques à l'aide du serveur Apache2, du langage de scripts PHP et d'une base de données MySQL. Il possède également PHPMyAdmin pour gérer plus facilement vos bases de données. »

Nous avons décidé d'utiliser WampServer car c'est un logiciel que nous avons déjà utilisé et c'est celui que nous connaissons le mieux. De plus il est régulièrement mis à jour et propose le français parmi ses langues d'utilisation.

2. Base de données

Comme pour tout site web, la réalisation de celui-ci a requis la création d'une base de données.

Dans notre cas, il nous a tout d'abord fallu créer des identifiants pour la gestion administrateur du site. L'inscription des visiteurs au site n'étant pas requis, nous avons tout simplement créé une table « Admin » avec un champ pour le pseudonyme et un autre pour le mot de passe.

A screenshot of a MySQL database interface showing the structure of the 'projetweb admin' table. The table has five columns: 'id' (int(11)), 'nom' (varchar(250)), 'prenom' (varchar(250)), 'username' (varchar(250)), and 'mdp' (varchar(20)).

Field	Type
id	int(11)
nom	varchar(250)
prenom	varchar(250)
username	varchar(250)
mdp	varchar(20)

Puis on a créé la table rédacteur gérée par l'administrateur, qui contient tous les rédacteurs insérés par ce dernier.

A screenshot of a MySQL database interface showing the structure of the 'projetweb users' table. The table has five columns: 'id' (int(11)), 'name' (varchar(20)), 'username' (varchar(20)), 'password' (varchar(50)), and 'email' (varchar(150)).

Field	Type
id	int(11)
name	varchar(20)
username	varchar(20)
password	varchar(50)
email	varchar(150)

On la table Catégorie gérée par l'administrateur

A screenshot of a MySQL database interface showing the structure of the 'projetweb cat' table. The table has two columns: 'id' (int(11)) and 'name' (varchar(250)).

Field	Type
id	int(11)
name	varchar(250)

Puis on créer les deux tables Document(**d**) et Section(**s**), la première stock les titres des documents et leurs auteur et la deuxième stock les sections des documents.

projetweb s	projetweb d
id : int(11)	id : int(11)
@titre : varchar(250)	@titre : varchar(250)
@contenu : text	@contenu : varchar(250)
#iddoc : int(11)	#iduser : int(11)
@idsec : varchar(250)	#idcat : int(11)

3. Les différentes étapes du développement

A. LA PROGRAMMATION

Dans cette partie on a commencé par la programmation JavaScript, on a commencé par la création d'un éditeur de texte, le fichier (editor.js) dans le zip de notre projet contient toute les fonctionnalités de l'éditeur.

-**L'éditeur de texte** contient les fonctionnalités essentielles pour éditer un texte, comme par exemple : la taille, la couleur, souligné, gras, italique, alignement...etc. aussi il contient un boutons « CreateSection » qui permet de générer une section dans le document de manière successive, on a aussi le boutons « Sauvegarder » qui permet de sauvegarder le document actuel.



Figure 8 : image de l'éditeur de texte.

Cet éditeur est fixé dans le header de la page.

-Puis on a entamé la partie de gestion de notre composite qui est la gestion des sections et des sous-sections avec les paragraphes et les images dans un document. Dans notre cas un document est structuré d'un titre et une catégorie à qui il appartient, un abstract, et des sections qui peuvent avoir des sous-sections et des images.

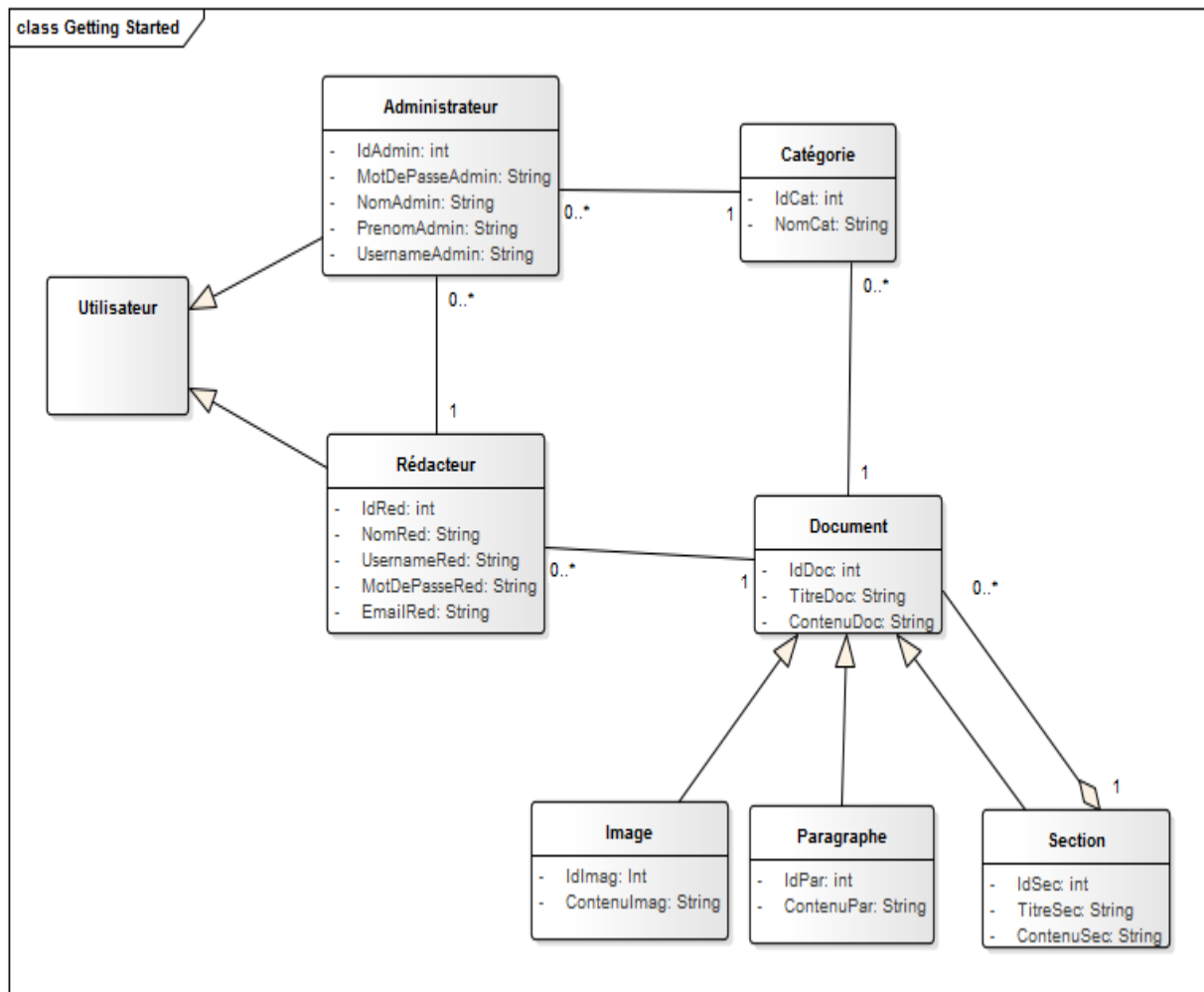


Figure 9 : l'instruction de composite.

Dans le fichier « **main.php** » on a implémenté les fonctions nécessaire pour la gestion d'un document.

Dans un document la sélection de la catégorie et la saisie d'un titre est obligatoire.

La création des sections et des sous sections se fait à la volé, c.-à-d. :

Si un rédacteur veut créer une section il clique sur le buttons « CreateSection » qui se trouve en haut dans l'éditeur de texte, l'image suivante nous montre la section créer après le clique :

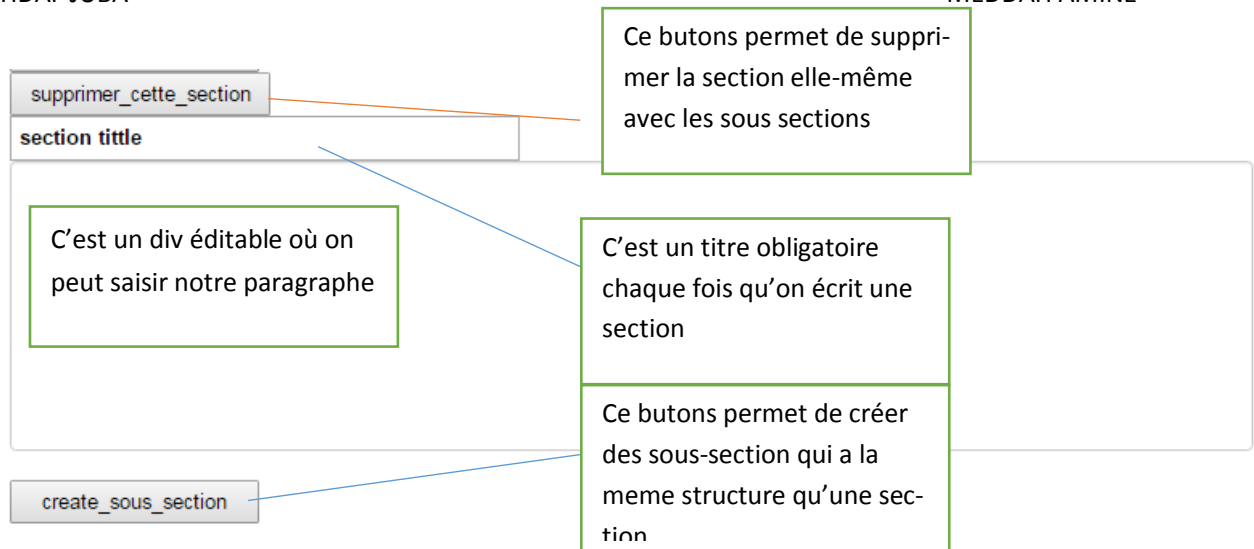


Figure 10 : une section.

Toujours dans la partie gestion des sections. Dans l'image précédente y a un buttons « create_sous_section » qui permet de créer une sous-section juste après la création de la section :

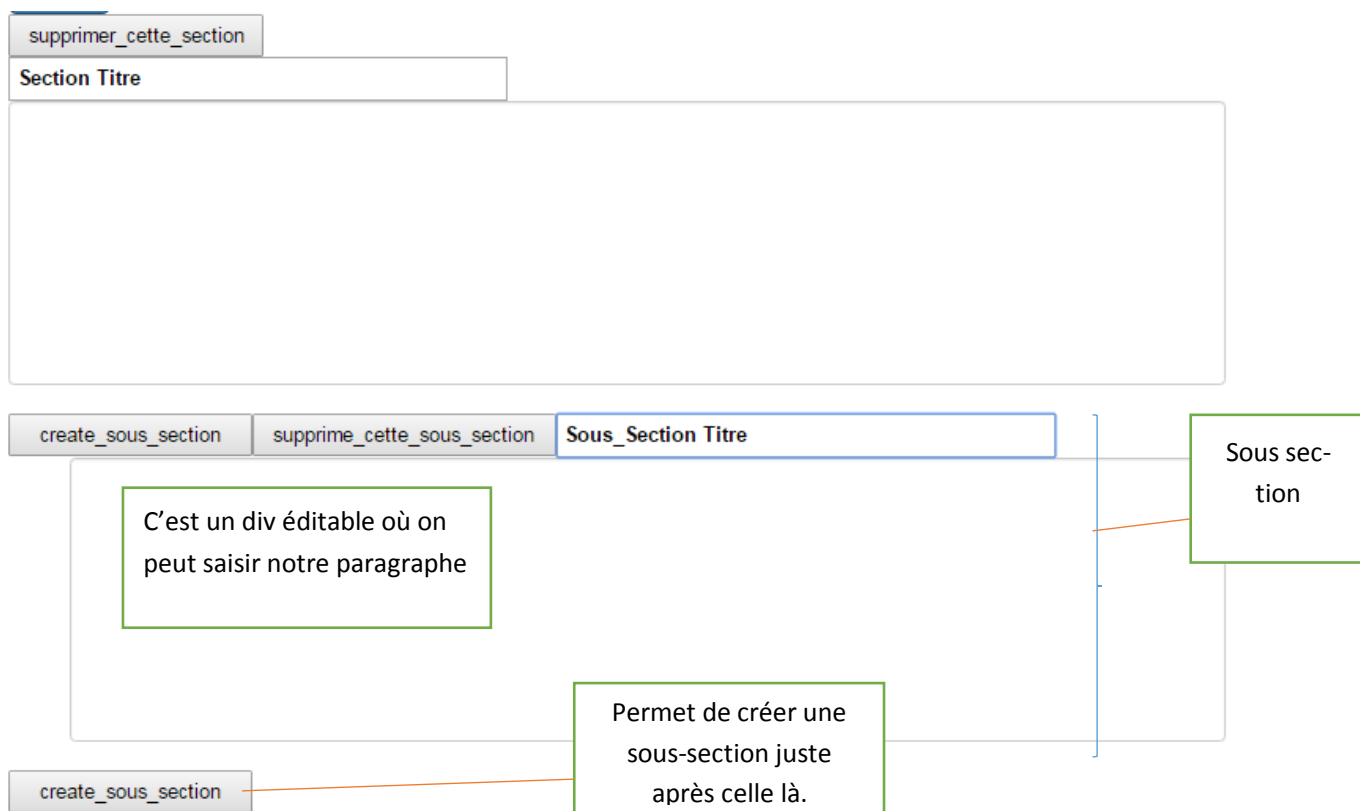


Figure 11 : une sous-section.

On voit très bien que la sous-section créer à la même structure qu'une sous section : un titre obligatoire, un buttons supprimer la sous-section et un buttons « create_sous_section ».

La gestion du sommaire se fait automatiquement chaque fois qu'en crée une section ou une sous-section, lorsque on clique sur l'un des buttons « CreateSection » ou « create_sous_section » un prompt de JavaScript nous oblige à saisir le titre.

Figure 12 : le prompt qui oblige la saisie d'un titre.

Si on n'accepte pas y'aura pas de création voulu, lorsque on appuyer sur « ok » le titre sera automatiquement inséré dans le sommaire qui se trouve à gauche de la page principale.



Figure 13 : la génération du sommaire.

Jusqu'à maintenant on a expliqué comment en créer les sections et les sous-sections pour avoir un document bien structuré, maintenant on va voir comment elle est la structure de notre logiciel soit sur la partie base de données ou sur la partie création document, pour cela on appliquer la structure MVC pour la gestion du site.

- la partie code « c » que on a intégrer pour la partie LATEX :

D'après nous recherche on a trouvé un compilateur open-source qui s'appelle « emscripten » permettant de compiler du bitcode LLVM en Javascript pour pouvoir l'exécuter dans n'importe quel navigateur web, il permet de prendre un code en C et le convertir en un code « .JS » ou « .html », alors notre but d'utilisation de compilateur et de créer un fichier « .c » et le convertir en « .js » et en « .html ».

Premièrement on a installé le compilateur, on l'a configuré sur une machine linux Ubuntu 14.04 LTS, puis on a créé un fichier « max.c » pour le test, on a pris l'exemple qui calcule le max entre deux nombre.

Puis à l'aide de certaine commande on a générer deux fichier « max.js » et « max.html ».

L'installation de compilateur

```
#Update the package lists
sudo apt-get update

# Install *gcc* (and related dependencies)
sudo apt-get install build-essential

# Install cmake
sudo apt-get install cmake

# Install Python
sudo apt-get install python2.7

# Install node.js
sudo apt-get install nodejs

# Install Java (optional, only needed for Closure Compiler minification)
sudo apt-get install default-jre

# Install git
sudo apt-get install git-core

# Fetch the latest registry of available tools.
./emsdk update

# Download and install the latest SDK tools.
./emsdk install latest

# Set up the compiler configuration to point to the "latest" SDK.
./emsdk activate latest

# Linux/Mac OS X only: Set the current Emscripten path
source ./emsdk_env.sh
```

Après l'installation on a créé le fichier max.c :

```
#include<stdio.h>

int max(int i,int j)
{
```

```

if (i > j) return(i); else return(j);
}

int main(int argc, char *argv[]) {
int i;
int j;
scanf("%d",&i);
//scanf("%d",&j);
int maxc=max(i,5);
printf("le max entre 5 et %d est : %d",i,maxc);
return 0;
}

```

Puis à l'aide des commandes suivante on a générer les fichier voulu :

```
$ emcc max.c // pour générer le « max.js »
```

```
$ emcc max.c -o max.html //pour générer le « max.html »
```

Le teste de lancement du fichier « max.html »

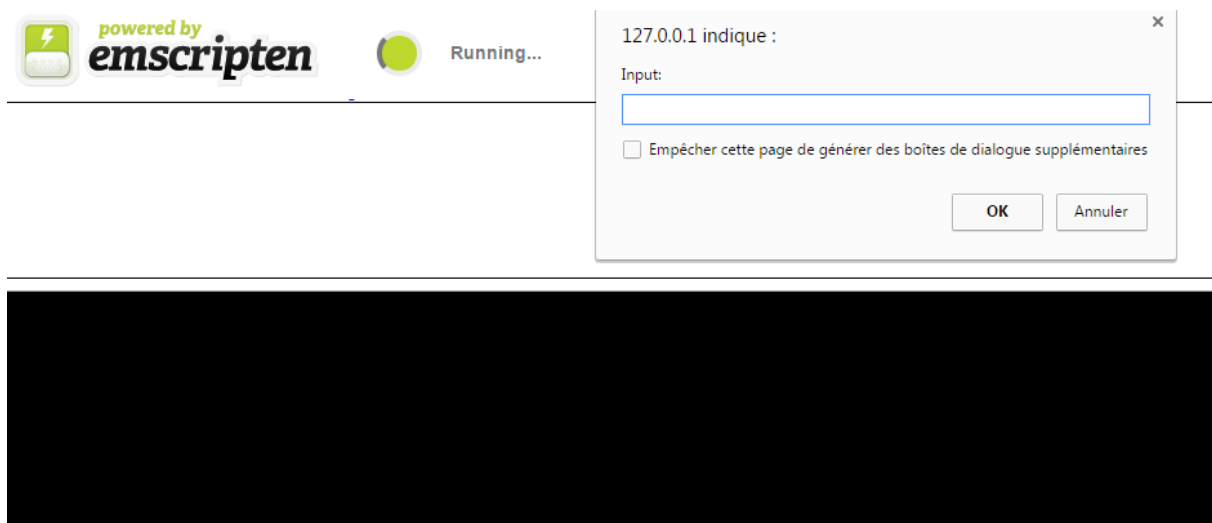


Figure 14 : lancement de la page « max.html ».

On voit très y a un prompt qui demande une entrée, dès que on rentre un entier on appuyer sur ok on va voir un message sur le textarea, « le max entre le 5 et ‘numéro entrée’ est : ..» (on a met 5 d’une manière standard pour faire une seul entrée), la figure suivante nous montre ça :



```
le max entre 5 et 8 est : 8
```

Figure 15 : le résultat de test sur « max.html ».

L'intégration dans notre système :

On a créé un lien vers cette page qui fait le calcul, et à partir de là on peut récupérer le résultat qui apparait sur le « **textarea** »

La structure MVC de notre site :

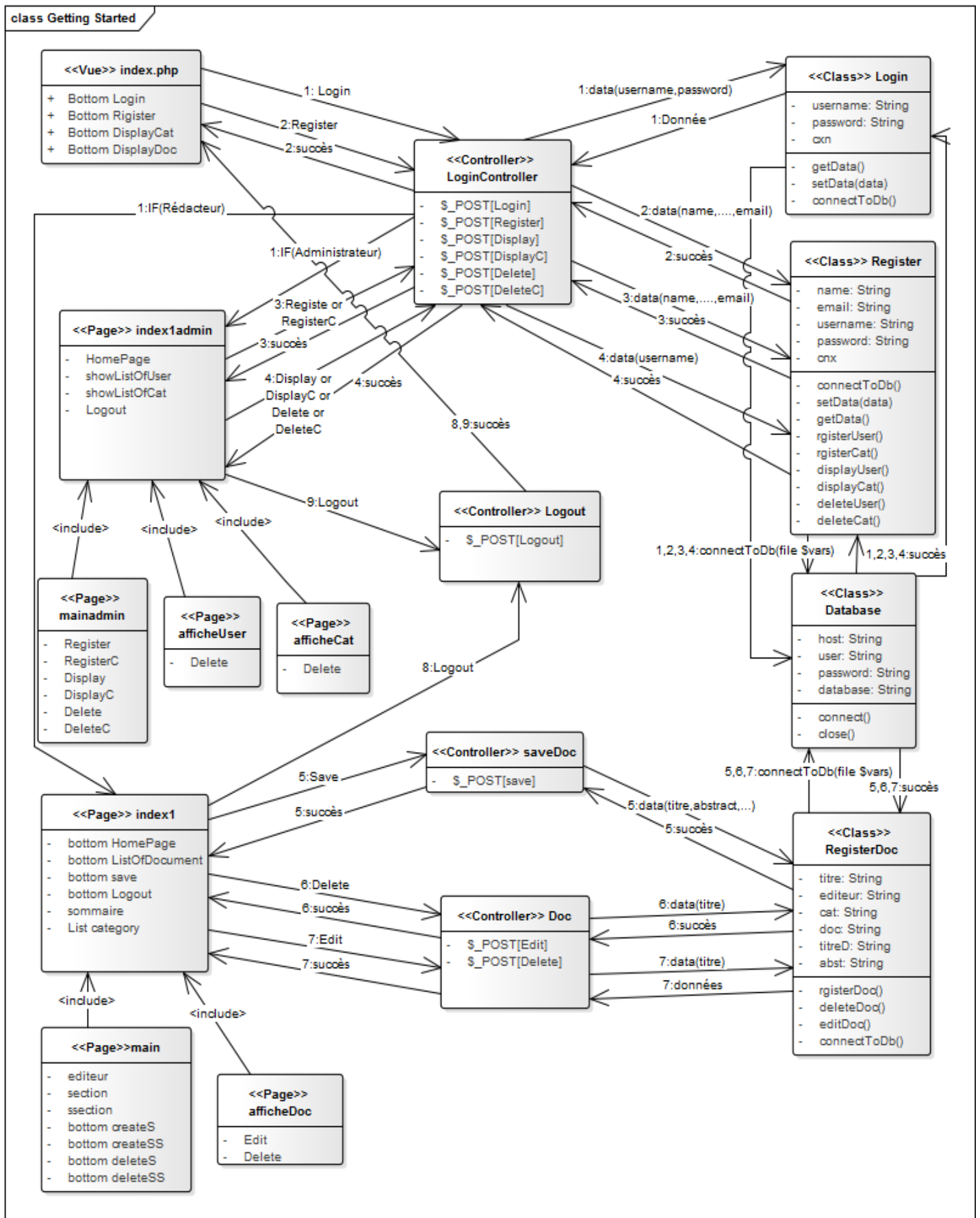


Figure 16 : la structure MVC de site.

B. Le Design

Au début nous avons fait une mise en forme basique avec des cadrages pour délimiter les différentes zones. Nous avons choisi de commencer avec un design basique, de développer les différentes fonctionnalités pour finir avec une mise en forme finale.

Sur ce premier design, nous distinguons trois parties séparées, l'en-tête (header), le corps composé de sections et d'asides) et le bas de page (footer). L'ensemble possédait un encadrement et était fixe, en partant des conteneurs jusqu'aux contenus. Nous avons pour cela paramétré les hauteurs et largeurs avec des valeurs en pixels.

Après on a entamé la partie de framework Bootstrap, il nous a fournie une bibliothèque riche du css et de javascript, alors pour la majorité de design du site on a utilisé les classe de style de Bootstrap. On a utilisé deux (3) fichiers de se framework :

-**bootstrap.css** comporte les classes de base de Bootstrap ;

-**bootstrap.min.css** comporte les mêmes classes de base que bootstrap.css mais est compressé.

-**bootstrap.js** contient le code JavaScript des composants de Bootstrap;

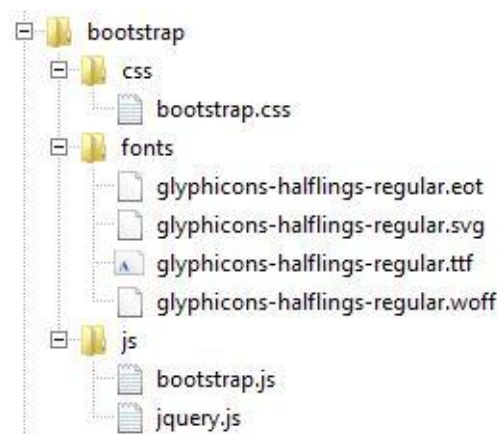


Figure 17 : l'arborescence de bootstrap.

La structure de design du site est comme suit :

Pour la page index.php, on s'est familiariser juste sur le conteneur du la page il est fixé au milieu de la page, il contient des buttons (afficher, login...) et des champs de saisie.

Puis pour les pages après authentification :

-le header contient un menu horizontale pour surfer dans le site, l'éditeur de texte qui est fixé en haut de la page et le nom de l'utilisateur a droite.

-la partie gauche de conteneur de la page contient un select pour sélectionner une catégorie et un sommaire.

-la partie conteneur et contenus c'est là on en insère les sections et les sous-sections d'une manière dynamique.

4. Résultats obtenus

Après beaucoup d'investissement, nous sommes satisfaits du résultat obtenu qui nous paraît être en adéquation avec le travail demandé. Ce qui manque c'est la partie intégration de la solution proposée à propos de « code c ».

Link - taille - - Couleur - - Align - G I

S	Ligne	UI
Ol	Lien	Image
MathML	Create section	Save

Welcome a

Home Page Show My documents

Document Category
Select Category ...

Sommaire
• section title

Your Document
Content of Document
WRITE A TITLE

Abstract

save
delete_this_section

Figure 18 : espace rédacteur.

5. Problèmes Rencontré

- Programmation

Les principaux problèmes rencontrés étaient principalement dus à des erreurs d'inattention.

- Design

Pour effectuer les cadrages basiques du 1^{er} design, il nous a fallu parcourir beaucoup de tutoriels (voir webographie) sur le CSS et se familiariser sur le Framework bootstrap, pour arriver aux objectifs attendus. Concernant le design, la principale difficulté était liée la gestion des sections et des sous-sections d'une manière à ce que les l'imbrication se fait d'une manière dynamique.

Conclusion

Le mini CMS que on a réalisé, dans la majorité pour rapport à ce qui a été demandé il répond au besoin.

Nous avons durant la réalisation de ce projet appliqué directement les connaissances acquises en cours. Pour réaliser un travail aussi important, nous avons fait beaucoup de recherches (bibliographie ou webographie) pour atteindre nos objectifs.

Nous avons réussi de manière efficace l'association de plusieurs langages. Pour arriver à ce résultat, nous avons fait preuve de beaucoup de patience pour arriver à coder correctement. La discussion entre collègues de promotion a été aussi bénéfique pour la réalisation de certaines parties du code. La répartition du travail a été efficace.

Webographie

- Langages utilisés : texte -> inspiré de Wikipédia
- Langages utilisés : image PHP -> source Site du zéro
- Choix des outils : description WampServer -> source WampServer.com
- Pour les tutoriels nous nous sommes fait aider des sites suivants :
 - Site du zéro
 - Développez.com
 - PHP débutant
 - Le manuel PHP en ligne
 - kripken.github.io
 - developer.mozilla.org/
 - gdcvault.com/
 - stackoverflow.com