LE QUELLEC Jean

Lycée Edmond MICHELET
BTS Systèmes Numériques Informatique et Réseau
2017/2018

RAPPORT DE STAGE

STAGE DEVELOPPEUR

Mairie de Clichy sous Bois
58 allée Auguste Geneviève
93390 Clichy sous Bois
du 23 octobre 2017 au 20 novembre 2017

TUTEUR :
DILAIN Alain
Attaché territorial

TABLE DES MATIÈRES

Table des matières

REMERCIEMENTS

Avant tout développement de ce rapport sur mon expérience au service informatique de la mairie de Clichy sous Bois, je tiens à remercier :

- Mon établissement, Edmond Michelet à Arpajon qui m'a permis de reprendre mes études et d'être sur la voie de réaliser mon projet professionnel, et à tous les professeurs pour leurs conseils et leur disponibilité.

- Le GRETA de l'Essonne pour la qualité des formations dispensées.

- Laurent DILAIN (Responsable du service informatique), pour m'avoir offert la possibilité d'effectuer mon stage au sein de la mairie de Clichy sous Bois.

- Tout le service, pour son aide, sa sympathie et pour avoir facilité mon intégration au sein de l'équipe.

- La mairie de Clichy sous Bois pour m'avoir permis d'effectuer ce stage.

CURRICULUM VITAE

EXPERIENCE PROFESSIONELLE

RATP (2005-2012)

Technicien de maintenance systèmes électroniques et informatique industrielle

SELECTRONIC (2014-2015)

Technicien vendeur

EARS SARL (2015-2016)**:**

Technicien réparateur

EQUINIX (2016-2017)**:**

Auditeur

COMPETENCES HARDWARE

Maintenance systèmes industriels:

- **Sécurisation biens et personnes** (GTB/GTC, Contrôle d'accès, Intrusion, Sécurité Incendie, Vidéoprotection)
- **Datacenter** (Audit baies et MMR, Brassage TELCO)
- **Evenementiel** (Sonorisation industrielle, Eclairage)

Conception circuits électroniques:

- **Analogique** (Audio, Mesure physique, Amplification)
- **Numerique** (IHM, Embarqué, Traitement du signal)
- **Conception et Test de Circuits Imprimés** (KiCAD, Proteus, Eagle)

COMPETENCES SOFTWARE

Technologies web:

- **HTML5**
- **CSS3** (SASS)
- **JAVASCRIPT** (Jquery)
- **PHP**
- **SQL**

Embarqué:

- **C/C++**
- **Bash, UBoot**
- **Assembleur AVR**

CAO/DAO:

- **Photoshop / Illustrator**
- **SolidWorks / Tinkercad**

FORMATION

- BAC STI Génie électronique (2004)

LANGUES

- **Anglais**, Autonome
- **Allemand**, Usuel

INTRODUCTION

J’ai choisi de m’inscrire au BTS SNIR proposé par le GRETA de l’Essonne afin de me réorienter dans le monde professionnel. J’ai comme objectif de devenir développeur ou administrateur système et idéalement pouvoir transmettre mes connaissances comme les professeurs l’ont fait pour moi.

Le BTS comporte un stage de 4 semaines en entreprise afin de mettre en pratique les connaissances acquises durant les cours.

C’est dans ce cadre que j’ai effectué un stage de développeur à la mairie de Clichy sous Bois, au service informatique.

Au cours de ce stage, j’ai réalisé une application WEB qui permet aux techniciens du service informatique un gain de productivité dans la maintenance des serveurs.

Mon stage a consisté, dans un premier temps, à analyser les besoins des techniciens et l’utilisation qu’ils faisaient des outils précédemment utilisés pour effectuer la maintenance des serveurs. J’ai ensuite dû faire les choix technologiques les plus pertinents afin de développer l’outil adéquat. Enfin j’ai réalisé l’application de façon itérative en demandant à, chaque étape, aux futurs utilisateurs si l’outil leur semblait pertinent et si son utilisation correspondait à leurs besoins.

L’élaboration de ce rapport a pour but de rendre compte, de manière fidèle et analytique, des 4 semaines passées au sein de la collectivité locale. Il apparaît donc logique de présenter, à titre préalable, la collectivité et le service où j’ai effectué le stage, d’un point de vue structurel et fonctionnel. Ensuite je détaillerai de manière générale les différentes tâches que j’ai réalisées durant ce stage ainsi que le savoir-faire acquis lors de cette expérience. Enfin je conclurai ce rapport par un bilan global, les apports humains et techniques de ce stage, afin de souligner les compétences que j’ai pu développer et de faire le lien entre cette expérience et mon projet professionnel.

PRÉSENTATION DE LA MAIRIE

Mairie de Clichy sous Bois

Clichy sous Bois est une commune de Seine Saint Denis, elle s'étend sur quatre kilomètres carrés et est peuplée par environ 30 000 habitants, près de 39% de la population a moins de 20 ans. Elle fait partie du Grand Paris ainsi que de Paris Métropole. Elle forme avec sa voisine Montfermeil une communauté d'agglomération, la communauté d'agglomération de Clichy-Montfermeil « *CaCM* ».

Cette commune proche de Paris a la particularité de n'être desservie par aucun axe routier ou autoroutier d'importance ni aucune voie ferrée, elle est par conséquent l'une des villes les plus enclavée de la petite couronne parisienne.

1.1 Budget

En 2014, le budget de fonctionnement de la commune de Clichy-sous-Bois se montait à 47 883 000€ et l'encours de la dette est de 30 364 000€. Il se décompose comme suit:

Recettes (47 883 000€):

- 10 933 000€ d'impôts locaux (taxe d'habitation + taxe foncière + CFE)
- 19 237 000€ de dotation globale de fonctionnement
- 7 801 000€ d'impôts et taxes diverses
- 9 912 000€ de revenus divers

Charges de fonctionnement (42 339 000€):

- 21 351 000€ de charges de personnel
- 12 922 000€ d'achats et charges externes
- 824 00€ de charges financières
- 2 190 000€ de contingents (Police, pompiers, aide sociale)
- 2 162 000€ de subventions versées (Associations, écoles, actions sociales)
- 2 890 000€ d'autres charges diverses

Résultat comptable: +5 543 000€

Le résultat comptable correspond au montant qui n' pas été dépensé, il peut servir, à rembourser la dette ou à investir.

1.2 Organisation

La commune emploie entre 600 et 700 agents dont à peu près 400 fonctionnaires de la fonction publique territoriale (titulaires et stagiaires). Les autres agents sont pour la plupart vacataires, les autres en CDD.

Le statut des fonctionnaires territoriaux est défini par le code général des collectivités territoriales.

Le statut des vacataires et agents en CDD est défini par le code du travail.

PRÉSENTATION DU SERVICE

Bureau du service informatique

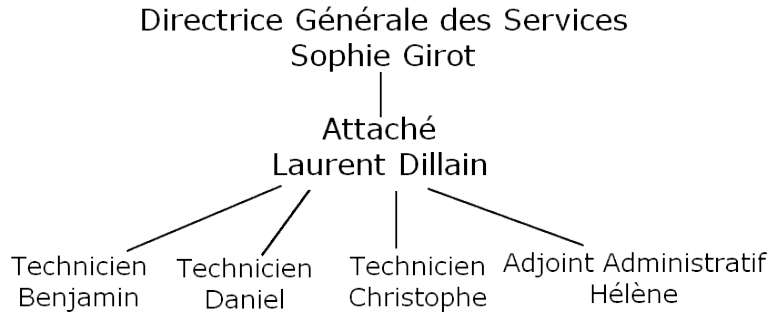
2.1 Historique

Du milieu des années 1970 à 2012, les services informatiques des communes de Clichy sous Bois et Montfermeil étaient mutualisés au sein de la communauté d'agglomération de Clichy-Montfermeil (CaCM). Les logiciels étaient fournis par le Syndicat Inter-Communal du Centre d'Informatique de Montreuil (SiCiM), les communes ayant adhéré au syndicat.

Les services rendus par le SiCiM étant de moins en moins pertinents pour un coût de plus en plus élevé, le CaCM a fait le choix d'internaliser les compétences. L'année suivante les communes ont choisi de ne plus partager leurs compétences et un service informatique a été créé dans chaque commune.

2.2 Organisation

Le service est composé de cinq personnes. Le responsable a le statut d'attaché territorial. Il y a trois techniciens et une adjointe administrative. Étant un service transverse il n'y a, hiérarchiquement, que la Directrice générale des services au dessus du responsable du service.



2.3 Services fournis

Le service informatique fournit plusieurs services :

- Les applications métiers
- Le support et la formation des utilisateurs
- Les postes de travail utilisateurs (environ 200) et leur maintenance.
- Les lignes ADSL (42)
- Les lignes de téléphonie fixes (465)
- Les lignes de téléphonie portable (212)
- Les points d'accès WiFi (12)
- La reprographie (18 photocopieurs, 2 presses, 80 imprimantes)

2.4 Infrastructure

Salle Serveurs

Le service informatique intervient dans 15 bâtiments municipaux et locaux d'associations ainsi que dans 23 écoles maternelles et élémentaires.

Afin de réaliser au mieux les missions dans un périmètre de cette importance, le service a fait le choix de déployer une infrastructure locale. Elle est composée de:

- 6 fibres noires pour relier les principaux bâtiments.

- 2 salles serveurs redondantes afin d'assurer la plus meilleure disponibilité possible.

- 47 switchs et autant de proxys.

Les différentes applications métiers étant directement hébergées par le service, cette infrastructure permet d'assurer le maintien en fonction des services même en cas de panne sur le réseau de l'opérateur. Il est aussi possible de relier un bâtiment, qui aurait perdu sa connexion ADSL à l'internet en se servant de l'infrastructure locale.

2.5 Budget du service

Le service, pour l'année 2015, a disposé d'un budget de 655 945€. Le service a dépensé 462 273€, soit un solde de 192 259€.

Un bilan comptable est disponible en annexe.

2.6 Exemple d'un marché

Le marché photocopieur

Le service informatique ne choisit pas directement ses prestataires. Les règles encadrant les marchés publics étant complexes, un service y est dédié.

Lors du renouvellement du marché photocopieur, un audit de la situation actuelle a été réalisé incluant les nouveaux besoins identifiés par les utilisateurs.

Un récapitulatif des demandes et une aide à l'arbitrage a été transmis au service « *marchés publics* ». (cf. Annexes)

Le service *marchés publics* lance ensuite un appel d'offre, une fois la date d'échéance atteinte, le service *marchés publics* effectue une analyse, suite à laquelle un des répondants est choisi. Dans le cas du marché photocopieur, la société RICOH a été retenue. L'analyse se trouve dans les annexes.

Enfin un acte d'engagement est signé entre la société retenue et la collectivité locale.

Une ligne budgétaire est affectée et permet d'honorer les factures émises par le prestataire. Une facture est disponible en annexe.

MON STAGE

Au cours de ce stage, j'ai eu l'opportunité de découvrir un métier sous toutes ses formes et de comprendre de manière concrète en quoi consiste le métier de développeur d'application WEB.

3.1 Présentation de la mission

Précédemment pour se connecter au serveurs, les techniciens utilisaient plusieurs fichiers Excel, y récupéraient les adresses IP correspondantes et les copiaient-collaient dans l'outil adéquat (KiTTY, Remote Desktop, Navigateur WEB).

Au cours du stage il m'a été demandé de réaliser une application permettant un gain de productivité pour les techniciens dans la maintenance des serveurs.

L'application doit permettre de:

- Trier les serveurs par catégories.
- Se connecter directement à un serveur en cliquant sur l'icône correspondant au type de connexion. Une information peut être fournie à l'utilisateur lorsque qu'il survole l'icône de connexion avec sa souris.
- Ajouter des serveurs, modifier un serveur existant, le supprimer et créer des catégories.
- Choisir un thème visuel.

3.2 Outils mis à disposition

Pour héberger l'application il m'a été fourni un serveur WEB Apache, incluant PHP 7 et une base de donnée SQL, MariaDB.

Pour mener à bien ma mission, il m'a été fourni un ordinateur portable et une connexion internet. J'ai dû choisir et y installer les logiciels utiles au développement. Notepad++, XAMPP, Chrome et Firefox.

3.3 Choix technologiques

Le choix d'utiliser une base de données de type SQL a été imposé par le tuteur du stage. L'autre impératif était de pouvoir se connecter depuis les navigateurs Chrome et Firefox.

Côté serveur, j'ai fait le choix d'utiliser PHP pour faire le lien entre la base de données et le navigateur grâce à une extension intégrée à PHP, PDO (PHP Data Objects).

Du côté utilisateur, j'ai utilisé les technologies HTML5, CSS3 et JAVASCRIPT. J'ai également utilisé la bibliothèque jQuery afin de simplifier la communication entre le code PHP et le JAVASCRIPT grâce aux appels AJAX.

J'ai également mis en place un dépôt GIT afin de faciliter la maintenance de l'application une fois celle ci livrée.

3.4 Méthodologie et réalisation

J'ai choisi de mettre l'utilisateur au cœur du développement en lui demandant à chaque ajout ou modification de fonctionnalité, si cette dernière lui semblait pertinente et correspondait à ses besoins. A l'issue du cycle de développement, j'ai rédigé une documentation, disponible en annexe, pour les utilisateurs, j'ai également commenté le code afin de faciliter sa maintenance.

3.4.1 Base de données

J'ai, en premier lieu réfléchi à l'architecture de la base de données correspondant le mieux aux besoins des utilisateurs. J'ai d'abord utilisé une seule table contenant toute les informations mais cette solution n'était pas assez souple pour les utilisateurs. Certains serveurs permettant de s'y connecter de plusieurs façons différentes.

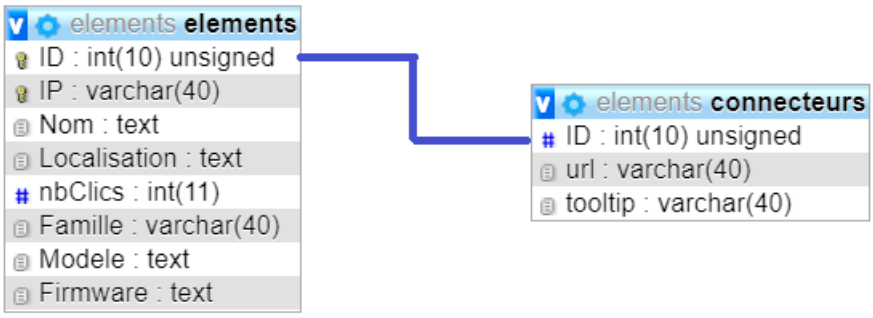


Schéma base de données

J'ai alors choisi d'utiliser 2 tables, une contenant les serveurs et les informations uniques s'y afférant (IP, Nom, Catégorie et Localisation) et une seconde contenant les types de connexions et les informations s'y référant. Le lien entre les 2 tables se fait grâce à l'ID du serveur fourni par la base de donnée.

Il a fallu en premier créer un outil permettant d'utiliser le PDO sans avoir à le réécrire à chaque fois.

```
function connexdb() {
  //paramétres de connexion à modifier
  $host = 'localhost'; // le chemin vers le serveur
  $db = 'elements'; // le nom de la base de donnée
  $user = 'root'; // nom d'utilisateur pour se connecter
  $pwd = ''; // mot de passe de l'utilisateur pour se connecter
  $dsn = "mysql:dbname=$db;host=$host";//creation du dsn pour mysql
  //utilisation de la base
  try {
    //Ouverture de l'accés à la base
    $pdo= new PDO($dsn, $user, $pwd);
        $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        $pdo->setAttribute(PDO::ATTR_EMULATE_PREPARES, false);
    // informer mysql du fait qu'on travaille en UTF-8
    $pdo->exec("SET NAMES 'utf8'");
    //echo"connectee a la base";
  } catch(PDOException $e){
    echo 'Erreur : '.$e->getMessage().'<br />';
    echo 'N° : '.$e->getCode();
    die('Echec de la connexion : '.$e->getMessage());
  } catch(Exception $e) {
    echo 'Erreur : '.$e->getMessage().'<br />';
    echo 'N° : '.$e->getCode();
    die('Erreur non liée à PDO : '.$e->getMessage());
  }
  return $pdo;
```

J’ai ensuite créé un outil permettant d’importer des fichiers issus d’un tableur dans la base de données afin de gagner du temps et de ne pas avoir à entrer un à un les serveurs dans la base de données. Cet outil a été codé en PHP.

3.4.2 Lien JAVASCRIPT <=> SQL

L’étape suivante a été la création de la vue côté client. J’ai utilisé principalement JAVASCRIPT pour construire cette vue.

Il faut en premier lieu récupérer les différentes catégories créées par les utilisateurs. Afin de construire un tableau contenant les serveurs, il faut récupérer la liste de ces serveurs côté client. Pour cela j’ai réalisé une API en PHP, le lien entre le JAVASCRIPT et le PHP est fait avec jQuery et les appels AJAX qui permet de récupérer un JSON. Je ne détaillerai que l’étape qui permet de récupérer la liste des serveurs d’une catégorie.

```
function sqlConnect(familly){ // Récupérer la liste des serveurs par catégorie
	listOfServeurs = []; // Vider la liste des serveurs côté client
	$.ajax({ // Connecteur AJAX jQuery
	url: 'app.php', // Appel de l'api
	type: 'GET',
	dataType: 'json',
	data: {action:'sqlConnectors', famille:familly}, // L'action et le paramètre à passer à l'API
	success: function(request){
	console.log('success SQL');
	request.forEach(function(elem){ // Pour chaque élements d'une catégorie
	listOfServeurs.push([elem[0], elem[1],elem[2],elem[3],elem[4],elem[5]]); // Mettre les données dans la liste des serveurs
	}); //0: IP, 1: Nom, 2: Localisation, 3: ID, 4: Famille, 5:Connecteurs
	},
	error: function(request){ // Afficher le retour de l'API en cas d'erreur
			console.log('fail SQL');
			console.log(familly);
			console.log(request);
	},
	complete: function(request){ // Complete ne garantit pas que l'action se soit bien passée juste qu'elle est terminée
		console.log('complete SQL');
		printCat(familly); // Afficher le tableau en HTML que la requète soit bien passée ou non
	}
	})
}
```

Dans cette fonction on voit la façon dont l'appel AJAX fonctionne. Elle appelle la fonction « sqlConnectors »dans « app.php » avec comme paramètre « familly » qui correspond à la catégorie de serveurs que l'on souhaite afficher. En cas de succès on récupère un objet JSON « request » Que l'on insère dans le tableau « listOfServeurs ».

Que la requête se soit bien ou mal passée, on construit le tableau, avec la fonction « printCat » qui sera expliquée par la suite.

Nous allons d'abord voir la fonction PHP derrière la requête.

```
function getSQLwithConnectors($_famille){ // Récupère tout les élements et connecteurs d'une catégorie
  $pdo = connexdb(); // Connexion à la BDD
  $_famille = strtoupper($_famille); // On passe le nom de la catégorie en majuscule
  $requeteSQL = "SELECT IP, Nom, Localisation, ID, famille FROM elements WHERE famille = '{$_famille}' ORDER by nbClics DESC"; // Requète SQL récupère tout les élements de la table elements d'une catégorie trié par nombres de clics
  $table = $pdo->query($requeteSQL); // Execution de la requète SQL
  $datas = $table->fetchAll(); // Insertion de la réponse dans la variable $datas
  foreach($datas as $key => $value){ // Pour chaques élements récupéré, on récupère les connecteurs dans la table connecteurs
      $requeteSQL = "SELECT elements.ip, connecteurs.url, connecteurs.ID, connecteurs.tooltip FROM connecteurs, elements WHERE elements.id = connecteurs.id AND elements.id = {$value[3]};"; // Requète SQL, Récupère les connecteurs (url et tooltip) par ID
      $table = $pdo->query($requeteSQL);    // Execution de la requète SQL
      array_push($datas[$key], $table->fetchAll());          // Pour chaque connecteurs on le mets à la suite de l'objet élements correspondant dans la variable $datas
  }
  echo json_encode($datas); // Renvoie de l'objet au Javascript
}
```

Dans cette fonction, on voit comment le PHP fait le lien avec la base de donnée avec des requêtes SQL et le JAVASCRIPT en renvoyant un objet de type JSON contenant les résultats de la requête.

Les utilisateurs m'ont ensuite demandé que les serveurs soient triés dans l'ordre du nombre de fois où l'on s'est connecté dessus. J'ai donc créé une variable dans la base de données qui s'incrémente à chaque clic sur un serveur. Et je demande à la base SQL de me fournir la liste triée en fonction de ce nombre de clics.

3.4.3 Vue client

La vue client est uniquement faite avec JAVASCRIPT, la mise en forme avec CSS3. Le code HTML est réduit à son strict minimum avec le menu et des « div » qui ne servent qu'à positionner les éléments.

```
function printCat(familly){ // Afficher la catégorie choisie

	var tableCat = document.getElementById("tables"); // recupère la div où sera imprimé le tableau
	if($('.Tableau').length>=1){	// Efface le tableau précédent possibilité d'afficher plus d'un tableau en changeant la condition de longueur
	$('.Tableau:nth-child(1)').remove();
	}
	var table = document.createElement("div"); // Créer une div par catégorie
	table.setAttribute("id", familly); // Nomme la div avec le nom de la catégorie
	table.setAttribute("class",'Tableau'); // Donne la classe Tableau
	TablesHtml = "<table class='table table-bordered' ><tr><th colspan='5'>"+familly+"</th></tr><th class='sortIP'>IP et Connect<a href=\"#\" onclick=\"sortByIp('"+familly+"',0);\"><img border=\"0\" alt=\"Tri\" src=\"./img/sort_up.png\" width=\"10\" height=\"10\"></a><a href=\"#\" onclick=\"sortByIp('"+familly+"',1);\"><img border=\"0\" alt=\"Tri\" src=\"./img/sort_down.png\" width=\"10\" height=\"10\"></a></th><th class='sortName'>Nom<a href=\"#\" onclick=\"sortByName('"+familly+"',0);\"><img border=\"0\" alt=\"Tri\" src=\"./img/sort_up.png\" width=\"10\" height=\"10\"></a><a href=\"#\" onclick=\"sortByName('"+familly+"',1);\"><img border=\"0\" alt=\"Tri\" src=\"./img/sort_down.png\" width=\"10\" height=\"10\"></a></th><th class='sortLoc'>Loc.<a href=\"#\" onclick=\"sortByLoc('"+familly+"',0);\"><img border=\"0\" alt=\"Tri\" src=\"./img/sort_up.png\" width=\"10\" height=\"10\"></a><a href=\"#\" onclick=\"sortByLoc('"+familly+"',1);\"><img border=\"0\" alt=\"Tri\" src=\"./img/sort_down.png\" width=\"10\" height=\"10\"></a></th><th></th><th></th></tr>"; // Créer le header du tableau avec les fonctions de tri

	var j=0; // Index du serveur dans Liste of Serveur
	listOfServeurs.forEach(function(elem){ // Pour chaque élements de la liste
	TablesHtml += "<tr><td class=\"tdIP\">"+listOfServeurs[j][0]; // Crée le champs avec l'adresse IP
	try{ // Evite de bloquer le script si aucun connecteur n'est affécté à l'adresse IP
		var nbOfConnectors = listOfServeurs[j][5].length; // Compte nombres connecteurs
	} catch(e){}
		for(k=0;k<nbOfConnectors;k++){ //Crée autant de connecteurs que besoin
			TablesHtml += "<div class=\"tdConn\">"
			var typeConnector = listOfServeurs[j][5][k][1].charAt(0); // Selectionne le premier caractere
				switch (typeConnector){ // Crée le connecteur en fonction du premier caractere (S=ssh, H=Http, R=RDP)
```

```
                                case "s" : // Cas SSH
                                TablesHtml += " <a title=\" "+listOfServeurs[j][5][k][3]+" \"
id=\""+j+"conn"+k+"\" onclick=\"incrementById("+listOfServeurs[j][3]+")\"
href=\""+listOfServeurs[j][5][k][1]+"\"><img border=\"0\" alt=\"Link "+k+"\"
src=\"./img/ssh.png\" width=\"25\" height=\"25\"></a>";
                                break;

                                case "r" : // Cas RDP
                                TablesHtml += " <a title=\" "+listOfServeurs[j][5][k][3]+" \"
id=\""+j+"conn"+k+"\" onclick=\"incrementById("+listOfServeurs[j][3]+")\"
href=\""+listOfServeurs[j][5][k][1]+"\"><img border=\"0\" alt=\"Link "+k+"\"
src=\"./img/rdp.png\" width=\"25\" height=\"25\"></a>";
                                break;

                                case "h" : // Cas HTTP/HTTPS
                                TablesHtml += " <a target=\"_blank\" title=\" "+listOfServeurs[j]
[5][k][3]+" \" id=\""+j+"conn"+k+"\" onclick=\"incrementById("+listOfServeurs[j][3]+")\"
href=\""+listOfServeurs[j][5][k][1]+"\"><img border=\"0\" alt=\"Link "+k+"\"
src=\"./img/web.png\" width=\"25\" height=\"25\"></a>";
                                break;

                                default: // Crée un lien texte si type de connecteur inconnu
                                TablesHtml += " <a target=\"_blank\" title=\" "+listOfServeurs[j]
[5][k][3]+" \" id=\""+j+"conn"+k+"\" onclick=\"incrementById("+listOfServeurs[j][3]+")\"
href=\""+listOfServeurs[j][5][k][1]+"\">Link "+k+"</a><br> ";
                                break;
                                }
                        TablesHtml += "</div>";
                        }
                TablesHtml += "</td><td class=\"tdNom\" >"+listOfServeurs[j][1]+"</td>"; // fermer la
ligne IP et Ajouter le Nom
                TablesHtml += "<td>"+listOfServeurs[j][2]+"</td>"; // Ajouter la localisation
                TablesHtml += "<td><a title=\"Modifier\" href=\"#\" class=\"tdMod\"
onclick=\"modifById("+listOfServeurs[j][3]+")\" ><img border=\"0\" alt=\"Modifier\"
src=\"./img/modif.png\" width=\"20\" height=\"20\"></a></td><td> <a title=\"Supprimer\"
href=\"#\" class=\"tdSuppr\" onclick=\"return confirme("+listOfServeurs[j]
[3]+",'"+listOfServeurs[j][0]+"');\"><img border=\"0\" alt=\"Supprimer\" src=\"./img/suppr.png\"
width=\"15\" height=\"15\"></a></td>"; // Créer les boutons modifier/supprimer
                TablesHtml += "</tr>";          // Ferme la ligne du tableau
                j++;
                });
                TablesHtml += "</table>";      // cloture le tableau
                table.innerHTML = TablesHtml; // Fabrique le tableau
                document.getElementById("tables").appendChild(table); // Affiche la div crée
                console.log('End print');
}
```

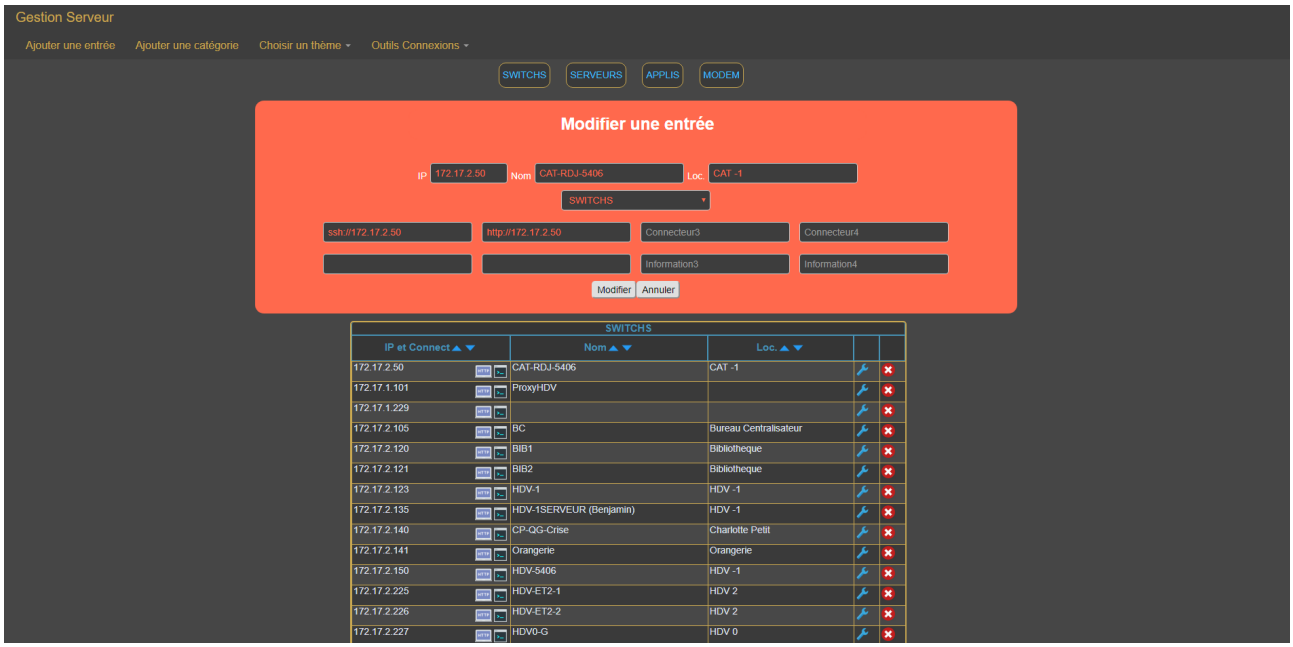
Cette fonction est la plus importante côté client, elle construit le tableau. On construit d'abord le titre du tableau avec les fonctions de tri par adresse IP, Nom et Localisation qui m'ont été demandées en fin de projet.

La fonction construit ensuite chaque ligne du tableau avec la même méthode. D'abord l'adresse IP, dans la même cellule on affiche des images cliquables en fonction du type de connexion. Pour ce faire, j'ai utilisé un « SWITCH-CASE » sur le premier caractère (S pour SSH, R pour RDP, H pour HTTP et HTTPS). On construit ensuite la cellule NOM, puis la cellule Localisation et enfin les cellules avec les liens pour modifier ou supprimer une entrée.

Aperçu de la vue client

3.4.4 Ajout Modification et suppression

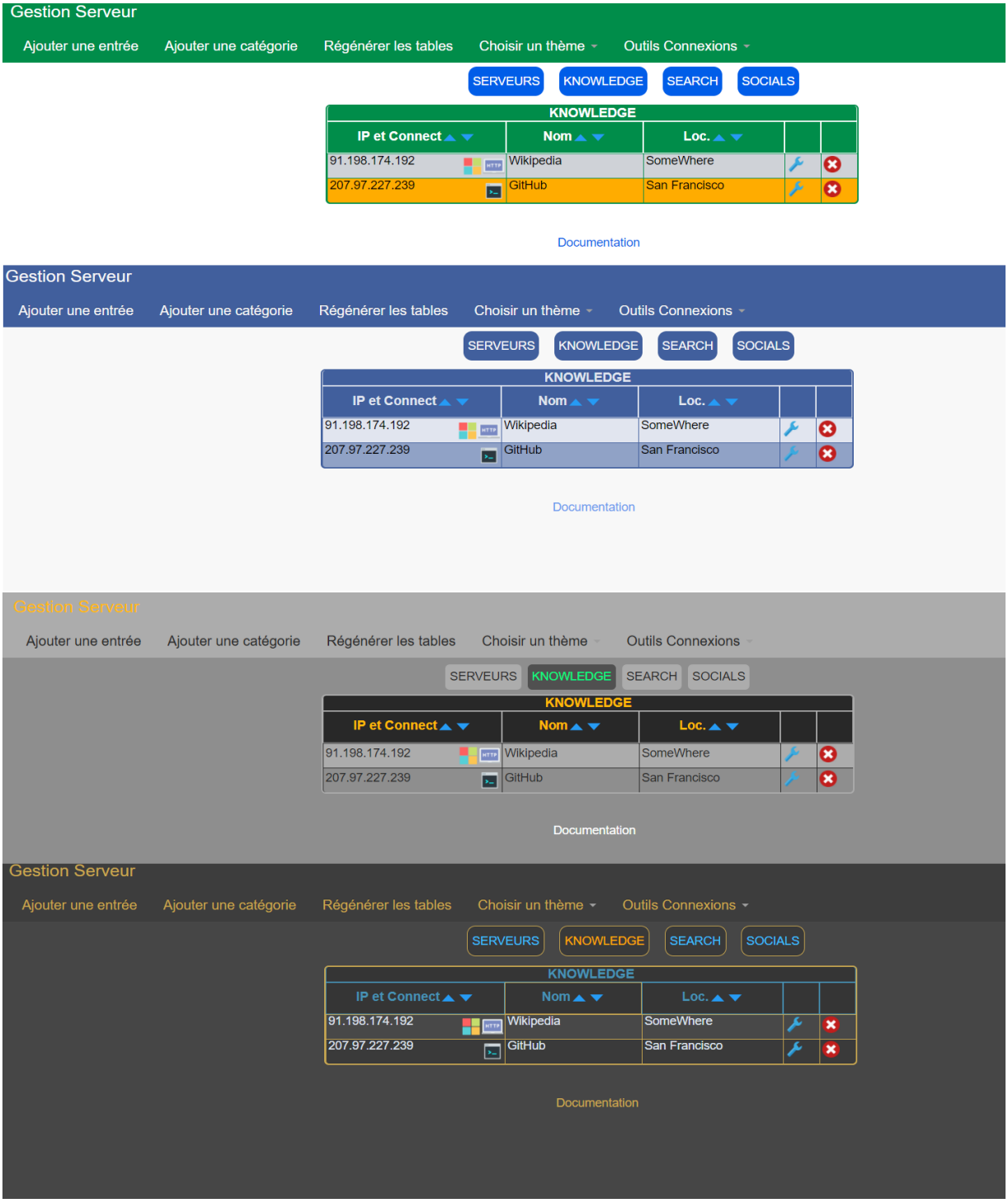
En plus de pouvoir afficher une liste de serveurs, il m’a été demandé qu’il soit possible de modifier, ajouter ou supprimer une entrée. J’ai crée pour une page PHP pour chacune de ces fonctions, cette page n’est jamais vu par l’utilisateur, dans un soucis de sécurité. J’utilise du JAVASCRIPT uniquement lors d’une modification afin d’afficher les informations à modifier.



Modification d'une entrée

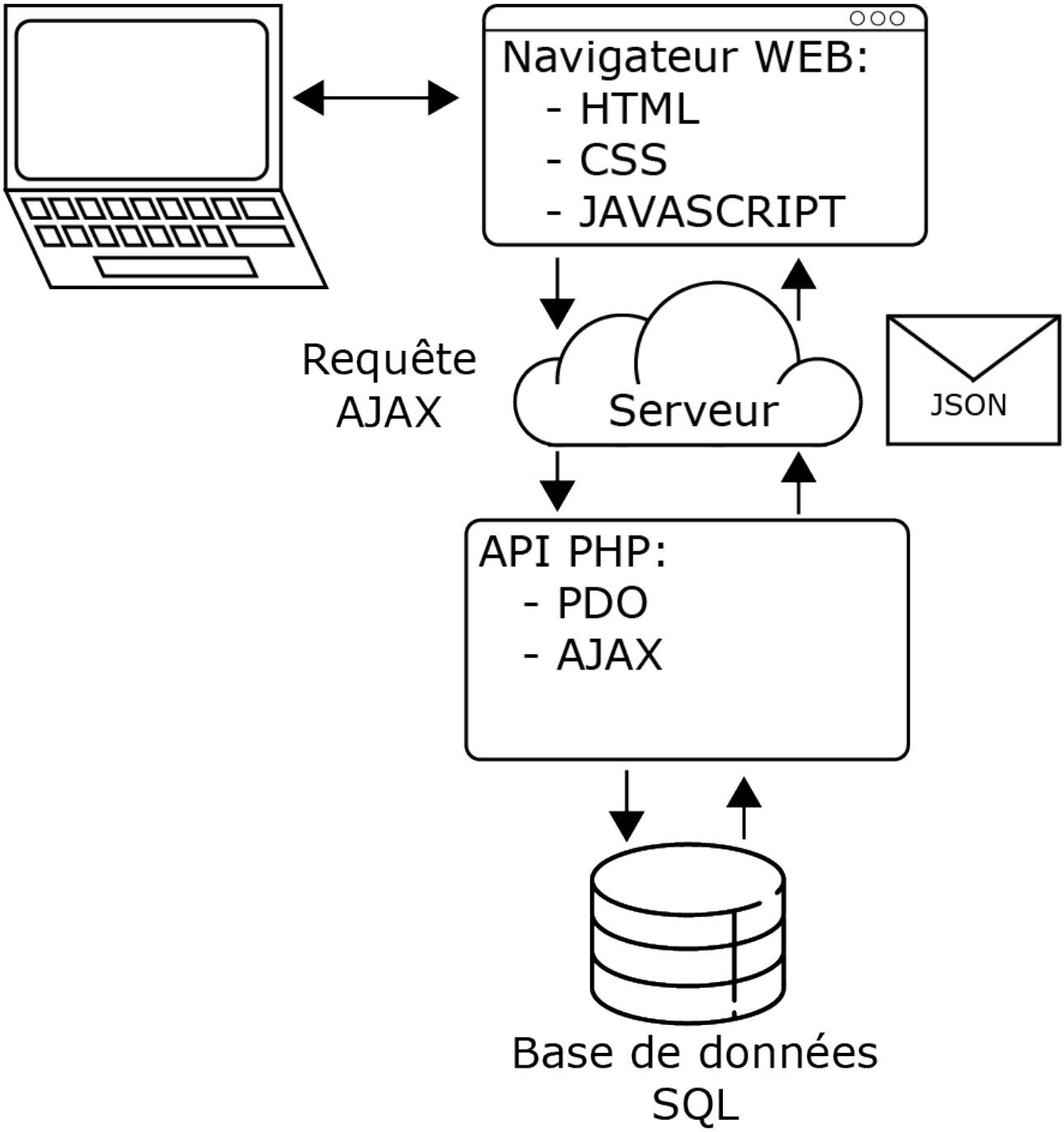
3.5 Thèmes

Il m’a été demandé de créer plusieurs styles visuels en fonction des préférences de chacun. J’ai donc créé plusieurs thèmes. Lorsque l’utilisateur se connecte à l’application il retrouve le dernier thème choisi grâce à un système de cookies.



Aperçu des thèmes

3.6 Synoptique de l’application



Synoptique de l'application

BILAN DU STAGE

4.1 Bilan personnel du stage

Ces quatre semaines de stage, que j'ai eu la chance de réaliser dans un service qui a su me faire confiance et me faire travailler en autonomie, furent enrichissantes au niveau personnel. J'ai pu mettre en œuvre mes capacités et en acquérir de nouvelles. Grâce à ce stage, j'ai découvert l'emploi vers lequel je me dirige. Je ne regrette en rien le choix de ce stage. J'ai éprouvé une certaine satisfaction dans la réalisation de mes tâches quotidiennes et ce stage a confirmé mon désir de poursuivre dans ce domaine.

Cette expérience m'a également permis de constater les contraintes du métier de développeur:

- La bonne compréhension des attentes des utilisateurs.
- Le doute et la remise en cause des choix effectués précédemment.
- La contrainte du temps et l'appréhension de la livraison dans les temps.

Ce stage m'a surtout permis de prendre confiance dans mes capacités à mener un projet de bout en bout et m'a conforté dans mon choix de réorientation professionnelle.

4.2 Bilan sur le projet

Durant les premiers jours, j’ai éprouvé des doutes sur ma capacité à mener à bien le projet. Dès que j’ai pris le temps de réfléchir à la méthode à adopter pour mener à bien le projet, progresser par petites itérations successives, mes doutes se sont dissipés. J’ai pu avancer rapidement tout en améliorant significativement mes connaissances en JAVASCRIPT, PHP et SQL.

J’ai aussi pu appréhender une problématique qui m’était jusque là inconnue, l’adéquation entre les besoins exprimés par l’utilisateur et les solutions à lui proposer. J’ai pu constater que les besoins ne sont pas toujours explicitement exprimés, c’est le retour vers l’utilisateur qui permet de valider une fonctionnalité.

GLOSSAIRE

KiTTY

KiTTY est un client SSH (Secure SHELL). Le protocole SSH est un protocole qui permet de faire des connexions sécurisées (chiffrées) entre un serveur et un client SSH. Dans le service où j'ai effectué mon stage, les utilisateurs utilisaient ce client.

Remote Desktop Protocol « RDP »

Remote Desktop Protocol (RDP) est un protocole qui permet à un utilisateur de se connecter sur un serveur exécutant Microsoft Terminal Services. C'est le protocole principal afin de se connecter à distance sur le poste d'un utilisateur utilisant Windows.

Serveur web Apache

Un serveur web est un logiciel permettant à des clients d'accéder à des pages web ou des ressources comme par exemple des images. Dans le cadre du projet fait en entreprise le serveur web utilisé était Apache. Apache est un serveur HTTP apparu en 1995. Il est distribué en licence libre Apache. Il est le plus populaire des serveurs web.

PHP: Hypertext Preprocessor

PHP est un langage de programmation libre, apparu en 1995, principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP. PHP est le langage Web dynamique le plus utilisé au monde, des sites comme Facebook ou Wikipédia l'utilisent.

API Application Programming Interface

Une interface de programmation est une façade clairement délimitée par laquelle un logiciel offre des services à d'autres logiciels. L'API que j'ai créé sert à se connecter à une base de données SQL via du code PHP.

PDO PHP Data Objects

PHP Data Objects est une extension définissant l’interface pour accéder à une base de données avec PHP.

AJAX Asynchronous JavaScript and XML

L’architecture informatique AJAX permet de créer une communication entre le navigateur de l’utilisateur et le serveur où se trouve l’application.

SQL Structured Query Language

SQL est un langage de requête structurée servant à exploiter des bases de données relationnelles. Le langage permet d’ajouter, de modifier ou de supprimer des données dans les bases de données. Un serveur SQL est une base de données utilisant le langage SQL

NotePad++

NotePad++ est un éditeur de texte léger créé pour l’édition de codes sources.

XAMPP

XAMPP est un ensemble de logiciels permettant de mettre en place facilement un serveur Web local.

GIT

Git est un logiciel de gestion de versions décentralisé.

jQuery

jQuery est une bibliothèque Javascript libre et multiplateforme créée pour faciliter l'écriture de scripts côté client dans le code HTML des pages web.