

# Rapport de stage :

## Développement d'un outil SIRH d'aide à la prise de décision

Staelen Rémi

10 Mai - 25 Juin

**Tuteur de stage** : Maxime ZANCHI

**Etablissement de formation** : EPSI Lille

**Entreprise d'accueil** : AllYouNeed, 28 rue Pierre Mauroy

## Remerciements :

En premier lieu, je voudrais remercier Maxime ZANCHI, directeur de l'agence *AllYouNeed* et maître de stage pour sa confiance, sa supervision et le partage des connaissances nécessaires pour évoluer dans le domaine de l'ingénierie informatique.

Je tenais ensuite à remercier Augustin MARIE et Julien VANHAUWAERT, qui sont d'autres stagiaires, également en B1, pour leur patience et leur professionnalisme.

De plus, je tiens à remercier M. MONTAGNE et Mme GROCKOWIAK du corps administratif d'EPSI Lille, d'avoir répondu à nos interrogations concernant le déroulement du stage et la rédaction du rapport lié à celui-ci.

Enfin, je souhaite remercier profondément ma mère, pour le soutien qu'elle m'a apporté durant la recherche de mon stage.

## Table des matières

Remerciements :.....	1
I. Introduction .....	4
A. La recherche.....	4
B. L'Entreprise.....	5
C. Les Missions .....	6
D. Début de stage .....	6
1. Formations.....	6
2. Quotidien et ressenti .....	7
E. PLAN.....	7
II. La création de la base de données « allyouneed ».....	8
A. Présentation.....	8
B. Démarche.....	8
C. Réalisation.....	10
1. Le dictionnaire de données .....	10
2. Le Modèle Conceptuel de données (MCD) .....	12
3. La base de données.....	14
4. Sécurité .....	18
III. La mise en place du Backend.....	19
A. Présentation.....	19
B. Démarche.....	20
C. Réalisation.....	21
1. La page d'enregistrement et de connexion .....	21
2. La liste des candidats et des entreprises.....	25
3. Fiche détaillée du Candidat.....	26
4. L'optimisation des échanges entre les fichiers .....	28
5. La migration de l'application en Symfony .....	30
IV. Utilisation d'API.....	31
A. Présentation.....	31
B. Démarche.....	31
C. Réalisation.....	33
1. API SMS : Twilio .....	33
2. API email : Mailjet.....	34
V. Évaluation des réalisations et des compétences mobilisées.....	36
A. Adéquation du travail .....	36
B. Compétences mises en œuvre.....	36

VI. Conclusion.....	37
VII. Glossaire .....	39
VIII. Bibliographie et Webographie.....	40
IX. ANNEXES .....	41
A. Annexe1 : Dictionnaire de données.....	41
B. Annexe 2 : Modèle Conceptuel de données .....	42
C. Annexe 3 : Page de création de comptes candidats.....	43
D. Annexe 4 : Page de connexion à l'application.....	44
E. Annexe 5 : Page de liste des candidats.....	45
F. Annexe 6 : Fiche détaillée du candidat .....	45

## I. Introduction

### A. La recherche

*« L'apprentissage est la seule chose que l'esprit ne puisse jamais, ne craindre jamais, et ne regrette jamais » - Léonard de Vinci*

C'est dans cette optique que j'ai commencé mes recherches de stage, en décembre 2020. Étant une personne avide d'apprendre, j'ai entrepris une démarche autodidacte pour développer mes bases en programmation dès 2018. Le fait de pouvoir créer virtuellement la quasi-totalité de ce que je désirais me faisait rêver. Puis, j'ai rejoint l'EPSI cette année, et je suis alors parti dans une démarche de recherche pour un stage de fin d'année afin de développer mes compétences préacquises et celles développées durant cette année scolaire.

Je voulais trouver un stage durant lequel je ne devrais pas seulement exprimer mes connaissances, mais également apprendre de nouvelles choses, que ce soit à propos de l'informatique, ou du monde du travail. De plus, étant passionné de programmation, je voulais offrir et développer mes capacités au sein d'une entreprise ou d'une équipe, ayant des connaissances dans ce domaine, et un stage était une occasion parfaite.

Ayant déjà une petite empreinte dans l'univers du digital, j'ai pu être mis en relation avec plusieurs entreprises durant mon mois de recherche. Je suis donc parti dans une démarche de multiples candidatures spontanées. Cependant, j'ai accusé plusieurs refus en raison de la situation sanitaire. Les entreprises ne savaient pas encore si elles seraient toujours en télétravail ou non au mois de mai ou juin, ce qui les obligeait à me fournir une réponse négative, en prévision des décisions gouvernementales. De plus, étant en première année, je ne suis pas la cible principale des entreprises en ce qui concerne les stagiaires. En effet, celles-ci ont tendance à vouloir quelqu'un de plus expérimenté, en deuxième ou troisième année, pour ensuite les prendre en alternance, et les embaucher à terme.

Cependant, j'ai alors décidé de demander à Maxime ZANCHI, intervenant pour le module *HEP Onboarding*, s'il était intéressé pour me prendre en tant que stagiaire. Je trouve que ses méthodes d'apprentissage sont cohérentes et concluantes, ce qui faisait de son entreprise une priorité dans mes choix. De ce fait, j'avais enfin trouvé un stage, dans lequel j'allais pouvoir me développer tant bien en tant qu'étudiant d'ingénierie informatique, mais également découvrir un avant-goût de la vie professionnelle. J'ai donc effectué un stage du 10 mai au 24 juin à *AllYouNeed* à Lille.

## B. L'Entreprise

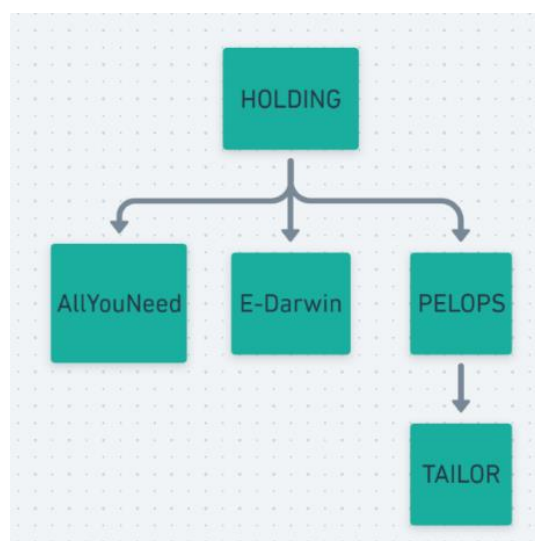
Le groupe *AllYouNeed* est une agence de recrutement de CDI et Freelance. Le groupe est composé de plusieurs départements : chaque département étant assigné à une profession :

- « SALES Marketing » pour les entreprises.
- « IT » pour tous les professionnels de l'informatique (réseau et système, programmation)
- « Top Management » pour les profils de prise de décisions (cadres)
- « Finance » pour ceux désirant travailler dans le domaine de la bourse
- « Freelance » qui est une branche récente, car le métier de free-lance en est en train de s'expandre au cours du temps

L'entreprise est située dans 1 seul espace, dans un immeuble de bureau. Concernant l'organisation du groupe *AllYouNeed*, elle se constitue ainsi :

- Le gérant : Maxime ZANCHI
- 2 CDI : Mélanie DOREE et Justine PIERSON
- Une alternante : Emma LE MOIGNE
- Un contrat pro: Madior FALL

Le groupe *AllYouNeed* n'est d'ailleurs pas la seule société dirigée par M. Maxime ZANCHI, comme montré sur le diagramme ci-dessous :



*Organisation des entreprises partenaires d'AllYouNeed*

La « Holding » est l'entité rattachant les 3 entreprises : *AllYouNeed*, *E-Darwin* et *Pelops*. Elle est celle qui va récupérer les capitaux, et va lui permettre de faire des remontées de fonds. *E-Darwin* est considérée comme étant une entreprise partenaire indépendante, même si elle appartient au même dirigeant. Elle va apporter un coaching professionnel aux candidats. *Pelops* quant à elle, est une société éditrice de logiciel, qu'un autre stagiaire en B1 a développé. Le logiciel se nomme *Tailor*.

L'idée centrale autour de ces entreprises est de remettre l'être humain au centre de toutes ses actions, en ayant des suivis réguliers, et des volontés transparentes.

## C. Les Missions

Durant ma période de stage dans l'entreprise *AllYouNeed*, j'ai eu l'occasion d'effectuer trois missions principales, en tant que développeur Back End, c'est-à-dire la relation entre le site internet et la base de données.

La première mission s'est déroulée en moins d'une semaine, je devais imaginer et créer la base de données, qui servirait à stocker les informations des entreprises et des candidats (en recherche d'emploi), mais également commencer à la sécuriser. Pour cela, j'ai effectué un dictionnaire de données connues en début de mission, ainsi qu'un MCD (Modèle Conceptuel de Données), que j'ai enrichi au fur et à mesure du stage, car nous ne connaissions pas toutes les tables et les données au début.

La deuxième mission s'est quant à elle déroulée du milieu de la première semaine jusqu'à la fin du stage. Nous devions, avec les autres stagiaires, créer une première version de l'outil *AllYouNeed*, où les candidats se connecteraient pour faire appel à l'agence pour leur trouver un emploi, mais où les entreprises pourraient également chercher de nouveaux salariés. De mon côté, je devais créer le Backend de l'application.

La troisième mission a débuté à la moitié de la création du site, nous (les stagiaires) devons rechercher diverses API pour pratiquer certaines actions que l'on ne pourrait pas faire sans : envoyer des SMS ou des mails, enregistrer et récupérer un calendrier Google, etc.

## D. Début de stage

### 1. Formations

Quelques semaines avant mon intégration au sein du groupe *AllYouNeed*, M. Maxime ZANCHI et moi avons fait un point sur mon niveau de connaissances en matière de développement web. L'objectif était de me former sur les différents

langages et technologies qui étaient importants dans le développement de la plateforme *SIRH*, imaginée par le dirigeant qui, par manque de temps, n'a jamais pu la développer lui-même. Sur mon accord et ma volonté d'apprendre et d'évoluer sur les technologies importantes pour moi, j'ai donc suivi une formation complète nécessaire au bon déroulement du stage qui vous sera présentée dans ce même compte rendu.

De ce fait, j'ai eu l'occasion durant ce stage de suivre plusieurs formations :

- « PHP / MySQL » pour la relation entre un site web et une base de données :16h
- « Symfony » pour simplifier le PHP : 20h
- « Gît et GitHub » pour le « versionning » avec un GitHub décentralisé : 4h

Ces formations m'ont permis d'entamer mon stage avec plus d'aisance et d'élargir mes connaissances. De plus, nous avions des outils à disposition comme un tableau pour visualiser les idées de chacun, un Trello pour dicter les tâches et un GitHub pour se partager les fichiers.

## 2. Quotidien et ressenti

Durant les jours de stage, en arrivant au bureau (9h30), nous faisons le point avec Maxime sur ce que nous étions en train de faire, ce que nous devrions travailler ensuite et ce dont nous avons besoin pour finaliser certaines actions. De plus, il nous arrivait de collaborer avec Florimond, stagiaire à *Pelops*, qui développait l'application *Tailor*, pour discuter sur une API exploitant les données de LinkedIn par exemple.

L'équipe était très accueillante. Ils ne nous mettaient pas de pression, demandaient régulièrement notre avis sur la planification des tâches, nous considéraient comme des membres de l'entreprise à part entière.

## E. PLAN

Tout au long de ce rapport, je reprendrai les différentes étapes clés qui ont permis de réaliser et de développer une version d'essai du logiciel *AllYouNeed*. À l'issue de ce compte rendu, j'apporterai ma vision ainsi que mon appréciation sur mon expérience personnelle et sur ce que j'ai acquis grâce aux différentes missions abordées durant ce stage.



## II. La création de la base de données « allyouneed »

### A. Présentation

La finalité de mon stage de fin d'année était de créer l'application AllYouNeed. En parallèle, il était indispensable de mettre en place une base de données reprenant un certain nombre d'informations vitales nécessaires au fonctionnement intégral de l'ensemble de l'application.

En effet, comment pouvons-nous proposer un candidat à une entreprise, si nous ne pouvons pas stocker et récupérer les informations de chacun, pour assurer la correspondance des profils ? C'est pour cela que nous devons absolument créer la base de données liée à cette application.

Le but de la construction de cette base de données est d'une part, stocker les informations des candidats et des entreprises et d'autre part, trier les candidats. Par exemple, si une entreprise a besoin d'un salarié ayant des compétences en développement mobile, on pourra alors demander à la base de données de nous fournir une liste de candidats ayant déjà fait du développement mobile. L'équipe AllYouNeed pourra alors contacter ces personnes pour savoir si elles sont intéressées par le poste. Ce processus permet à l'équipe de gagner du temps en leur évitant de devoir trier un par un les dossiers des utilisateurs pour déterminer lesquels ont les compétences requises et sont en plus intéressés par le poste. De ce fait, l'utilisateur se voit offrir une réponse à sa demande plus rapidement.

L'inconvénient majeur que j'ai rencontré au moment de la conception de cette base de données est que je n'avais pas connaissance des futurs ajouts que l'on apporterait à celle-ci. De ce fait, il y avait de grandes chances qu'au fur et à mesure de la création de l'application, je devais revenir modifier cette base de données (et ce fut le cas).

L'idéal aurait été que je sois en possession de toutes les données directement dès le début. Cependant, ce n'est pas si facile, car on ne peut pas savoir absolument toutes les informations avant de créer le site petit à petit, car c'est à ce moment que l'on se rend compte que l'on a oublié certains éléments, ou que d'autres besoins se créent.

### B. Démarche

Comme évoqué précédemment, si l'on ne connaît pas les informations dont on va avoir besoin, il est impossible de créer cette base de données. C'est pourquoi, avant de commencer sa conception, il fallait impérativement avoir connaissance des renseignements vitaux au bon fonctionnement de l'application.

Pour avoir une idée du type de données dont nous allions avoir besoin, j'ai listé avec M. Maxime ZANCHI les informations importantes connues lors de ce début de stage :

- Les candidats cherchant un poste devront s'inscrire en mentionnant :
  - Nom
  - Prénom
  - Téléphone
  - Statut
  - Adresse mail
  - Dernier poste occupé
  - Dernière entreprise
  - Mot de passe
  
- Les entreprises s'enregistrant par le biais d'un contact voulant trouver un salarié vont y renseigner :
  - Nom
  - Prénom
  - Téléphone
  - Poste occupé
  - Mot de passe
  - Et toutes les informations de l'entreprise.

Pour cela, dans nos fichiers, nous allons modifier les bouts de code où nous affichons des éléments sur la page web. À la place, nous effectuerons des requêtes Ajax vers le JavaScript. Une requête Ajax est une manière de faire passer des informations entre des fichiers à travers une page web. Elle permet d'effectuer des processus sans interrompre le chargement d'une page web.

Ensuite, il m'arrivait d'aller collecter d'autres informations nécessaires à certaines données si je ne les avais pas en ma possession : quel type de donnée est-ce ? Est-elle obligatoire ? Cela réduisait au possible les risques de mettre en place une mauvaise variable et donc cela éviterait de se retrouver avec des erreurs dans l'application lors de l'utilisation de cette base de données.

Cependant, il n'est pas du tout conseillé de créer une base de données directement, sans préparation préalable. Il y a une liste d'étapes à suivre.

Premièrement, il faut mettre en place un dictionnaire de données. Un dictionnaire de données est une liste où l'on va y référer la totalité des données que nous utilisons. En effet, un dictionnaire de données est le moyen le plus efficace pour avoir une vue de toutes les données en même temps, et de pouvoir connaître pour chacune d'entre elles :

- Son nom
- Sa signification
- Son type
- Sa longueur
- Sa nature
- Ses règles de calculs

Ensuite, il est préférable de créer un MCD. Un MCD est une manière d'imager une base de données, pour mieux la visualiser, et y inscrire les dépendances entre les tables. Cela peut être d'une grande aide en tant que support visuel si l'on n'arrive pas très bien à visualiser une base de données.

Enfin, quand l'on est sûr d'avoir toutes les informations nécessaires, on peut commencer à créer la base de données, sur un logiciel ou grâce à un service tiers.

## C. Réalisation

### 1. Le dictionnaire de données

Tout d'abord, comme énoncé plus haut, la première étape à la mise en place de cette base de données est la rédaction d'un dictionnaire. Pour cela, plusieurs logiciels nous sont accessibles. Tout d'abord, nous pouvons utiliser un outil de tableur comme *Google Sheets* ou encore *Microsoft Excel* (la création de tableaux est également possible sur *Microsoft Word*, mais moins pratique d'utilisation qu'un vrai tableur).



Logo de Microsoft Excel



Logo de Google Sheets

J'ai choisi d'utiliser *Microsoft Excel*, car c'est le tableur que j'utilise de préférence. Il peut être utilisé sans connexion internet contrairement à *Google Sheets*, et je n'ai pas l'obligation de stocker mes fichiers dans le Cloud. Je peux alors commencer à rédiger ce dictionnaire (Annexe 1).

	A	B	C	D	E	F	G
1	Nom de la table	Nom de la variable	Description	Type	Longueur	Nature	Règle de calcul
2							
3	Administrator						
4							
5		idadmin	Identifiant de l'administrateur	int	11	élémentaire	

*Exemple de ligne du dictionnaire de données*

Sur l'image ci-dessus, vous pouvez voir en haut, une ligne annonçant le contenu des cases d'intitulé de la colonne :

- Tout d'abord, il y a le nom des tables. Chaque ligne où se situe le nom d'une table est colorée en vert et les éléments de cette table sont décalés d'une case vers la gauche. Cela permet une meilleure visibilité, et on s'y retrouve mieux pour chercher une table spécifique. De plus, par souci de convention, le nom d'une table est toujours en anglais.
- Ensuite, il y a le nom de la variable. Celle-ci est également en anglais pour suivre les conventions. Sauf pour les identifiants (id\*\*\*) qui servent à différencier 2 éléments dans une même table, le nom des variables est suivi d'un « \_ » et du nom de la table. Par exemple : « name\_admin » est la variable où l'on va mettre le nom de l'administrateur. On utilise cette nomenclature lorsque le nom d'une variable apparaît dans 2 tables différentes (comme le prénom par exemple).
- De plus, la description de la variable est également citée. Cela a pour but que les personnes extérieures, développeurs ou non, puissent comprendre à quel élément est associé la valeur. Cela sera surtout utile pour quelqu'un nous remplaçant.
- Suivant la description, il y a le type. Le type fait référence à si l'objet est une chaîne de caractère (varchar), un nombre entier (int), un nombre flottant (float), un booléen (boolean) ou une date (date). Il existe bien d'autres types, mais ce sont les cinq principaux que l'on utilisera.
- Vient à cela la longueur de l'information. Par exemple, une chaîne de caractère a une longueur de base de 255 caractères maximum, mais on peut tout à fait modifier cela. Un autre exemple, la longueur d'un booléen est toujours de 1, car il n'y a qu'un seul nombre (0 ou 1).
- Ensuite, nous avons la nature de la variable. Les deux principales natures sont : « élémentaire » ou « calculé ». Une variable élémentaire signifie qu'on a juste besoin de la définir (comme un nom par exemple), tandis qu'une variable calculée va en prendre la valeur d'autres variables et les calculer selon la règle de calcul associée.

En enregistrant les informations de chaque champ pour chaque variable, nous avons enfin notre dictionnaire de données. Il est maintenant temps de passer au MCD (Modèle Conceptuel de Données).

## 2. Le Modèle Conceptuel de données (MCD)

Comme indiqué dans la présentation, un MCD (Annexe 2) est une représentation d'une base de données sous forme d'entités. Une entité est une table que l'on représente dans un MCD. Le MCD a pour but de simplifier et d'imager la base de données pour avoir une meilleure compréhension de celle-ci. Pour cela, de nombreux logiciels tiers existent comme *GitMind*, *Visual Paradigm*, ou encore *Looping*.



GitMind

Logo de GitMind



Logo de VisualParadigm



Logo de Looping

Entre ces trois logiciels gratuits, Looping a été mon choix, car nous l'avions déjà utilisé à plusieurs reprises, de ce fait, j'avais déjà des connaissances liées à ce logiciel. De plus, GitMind et VisualParadigm sont intéressants pour faire des MCD plus complexes. Or ici, le but n'est que de faire une représentation de la base de données pour s'y retrouver visuellement. Ce pour quoi il n'est pas nécessaire d'utiliser de gros logiciels. Enfin, Looping ne consomme qu'une infime partie de la mémoire vive, ce qui en fait donc un logiciel rapide et agréable d'utilisation.

Au départ, je n'avais que trois entités (table) dans ce MCD : *Administrator* (administrateur), *Company* (entreprise) et *Customer* (candidat). De plus, elles n'avaient aucun lien les unes envers les autres. Cependant, au fur et à mesure du stage, nous avons commencé à avoir d'autres tables, toutes liées à l'entreprise :

### - *Contact* (les contacts) :

Cette table a pour utilité de stocker les informations des contacts que l'on a avec cette entreprise. Comme la personne qui s'est inscrite au nom de celle-ci sur le logiciel par exemple, ou encore d'autres personnes que l'on aurait besoin de contacter, comme un salarié du service RH, ou encore le directeur.

On y conserverait :

- Son identifiant unique (ID)
- Son nom
- Son prénom
- Son numéro de téléphone
- Son email
- Sa place dans l'entreprise (le poste)
- Son mot de passe pour accéder à l'application

- Son moyen de contact. En effet, chacun a son propre choix de contact. Certaines personnes n'aiment pas être dérangées sur leur téléphone, d'autres préféreront les e-mails.
- L'identifiant (ID) de l'entreprise pour laquelle l'employé travaille

- *Mission* (les missions) :

C'est ici que l'on exposera les données liées aux missions. En effet, les entreprises veulent chercher des candidats pour atteindre leurs objectifs.

Il y aurait :

- L'identifiant unique de la mission (ID)
- Le manager de la mission
- Son poste
- L'intitulé de la mission
- Le résumé de la réunion avec le manager
- Le signé de la mission
- La date de création et de fermeture de la mission
- Le chiffre d'affaires de la mission

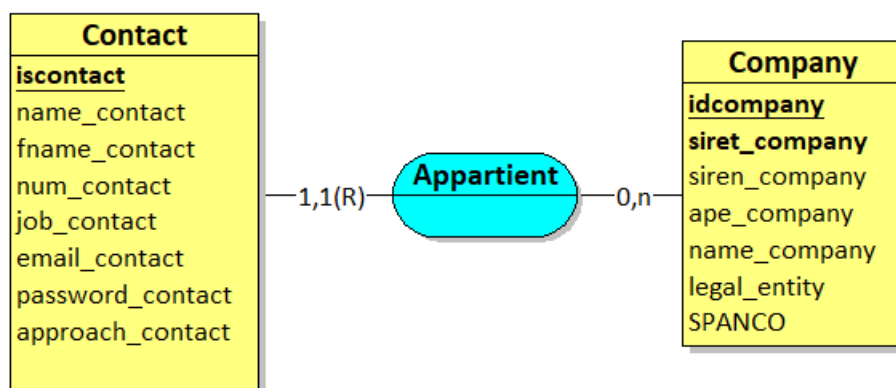
- *Revenue* (le chiffre d'affaires) :

Celle-ci a pour but de préserver les informations du chiffre d'affaires rapporté à *AllYouNeed* grâce aux missions que les candidats effectuent chez l'entreprise

Il y aurait :

- Le salaire annuel prévisionnel du salarié
- Le pourcentage que prend AllYouNeed sur ce salaire
- Les revenus prévisionnels que l'entreprise va donner à AllYouNeed
- Le total prévisionnel qu'AllYouNeed va recevoir

Ayant récupéré ces informations, j'ai ainsi pu étendre ce MCD, pour que l'on puisse bien distinguer les relations entre les tables.



*Exemple d'une relation entre 2 tables*

Sur l'exemple ci-dessus, on peut y voir l'entité (la table) *Contact* liée à l'entité *Company* (l'entreprise). Les entités ont toutes deux leurs variables. Celles en gras et soulignées indiquent que ce sont des clés primaires : une clé primaire a des caractéristiques similaires à un identifiant. En effet, il doit être unique et obligatoire.

L'élément uniquement en gras est une clé unique. Comme son nom l'indique, il ne peut pas y avoir deux fois le même « siret\_company » (le Siret de l'entreprise).

Enfin, on peut y voir des nombres entre les deux entités : ce sont des relations. Dans notre cas, prenons-le « 1,1(R) ». Que signifie-t-il ?

Il signifie qu'un « contact » est lié (ou dans notre cas, il « appartient ») à une et une seule « entreprise (company) ». Le « (R) » signifie une appartenance à l'entité « Company ». On peut y faire allusion à une salle de classe appartenant à une école : cette salle de classe ne peut pas appartenir à une autre école.

De l'autre côté, « 0, n » signifie qu'une entreprise peut avoir 0 ou plusieurs contacts. Il existe différentes relations, comme le « 1, n » qui exprime le fait qu'il peut y en avoir plusieurs, avec 1 au minimum.

Nous en avons donc fini avec notre MCD. Maintenant que nous avons fait la liste des variables de notre base de données et que nous connaissons les relations, nous pouvons alors commencer la création de notre base de données.

### 3. La base de données

Nous voici à la dernière étape de cette mission : la création de la base de données. Nous avons choisi de la faire en local, c'est-à-dire de stocker la base de

données sur un serveur que produit notre machine. Nous avons choisi de procéder de cette manière, car il ne s'agit que d'une première version de l'application et qu'il n'était donc pas nécessaire d'avoir le serveur mis en ligne.

Cependant, ici encore, de nombreux outils existent pour nous aider à remplir cette tâche. Les plus connus étant *Microsoft SQL Server*, ou encore *Xampp*.



Logo de Xampp

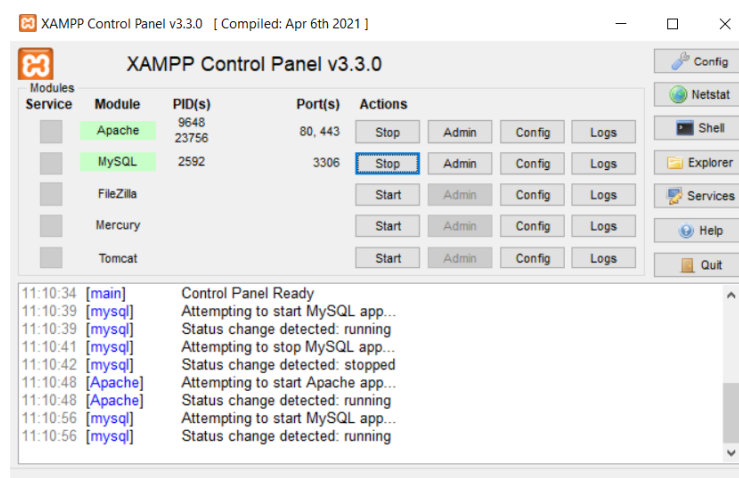


Logo de Microsoft SQL Server

*Microsoft SQL Server* est un logiciel pour créer et administrer des bases de données, tandis que *Xampp* est plutôt un ensemble d'outils permettant d'administrer un serveur et une base de données. En effet, lorsque l'on installe *Xampp*, le logiciel installe en même temps *Apache2* (ce qui va nous permettre de lancer un serveur sur notre machine), *PHP* (le langage qui permet l'interaction entre un site web et une base de données), *MySQL* (qui est le système de gestion de base de données) avec *Maria DB* (qui va nous permettre de créer des bases de données).

J'ai choisi d'utiliser *Xampp* car il nous fournit tous les outils nécessaires pour gérer des bases de données efficacement. En effet, j'ai à ma disposition une interface d'administration claire et efficace contrairement à *SQL Server* qui est compliquée à prendre en main, mais plus puissante que son rival.

Lorsque l'on installe *Xampp*, un menu de contrôle apparaît :



Menu de contrôle de Xampp

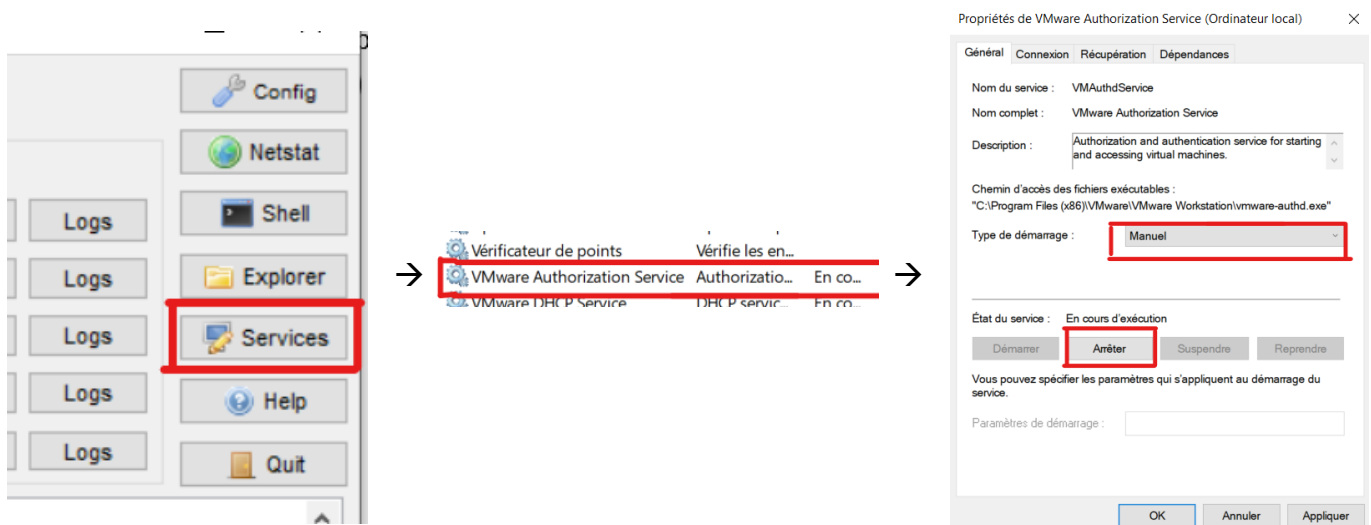


Sur ce menu, en dessous du titre « XAMPP Control Panel », on peut y voir le module, le PID, les ports, les actions et les options. Je ne sais pas quel est le « Service » devant « module », cela est toujours resté un carré grisé, et l'on ne peut pas interagir avec.

Lorsque l'on clique sur le bouton « start » dans « Actions sur la ligne du module que l'on veut utiliser, le nom se met en vert pour indiquer qu'il est bien en fonctionnement et un message apparaît tout en bas.

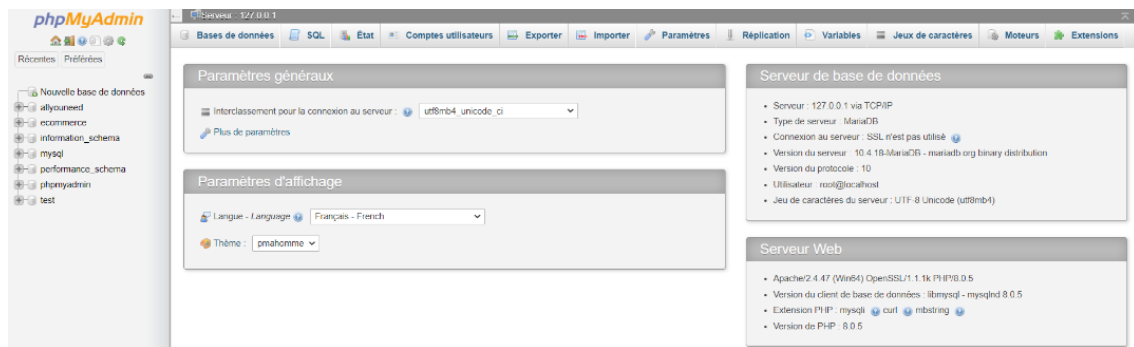
De plus, le(s) PID(s) et le(s) port(s) correspondant au processus s'affichent. Le PID est le numéro d'identification d'un processus. De plus, si l'on prend l'exemple « Apache » ici, le port 80 signifie qu'il est lié au HTTP (protocole pour naviguer entre les pages web), et le port 443 signifie le HTTPS (c'est le HTTP, mais avec une couche de sécurité en plus).

Cependant, il arrive que des logiciels en cours d'utilisation usent déjà les ports dont Apache a besoin par exemple. Pour remédier à cela, il faut aller dans « Services » (le 5<sup>e</sup> bouton en partant du haut dans la barre verticale à droite), chercher le programme qui bloque le port, puis l'arrêter manuellement (comme les images ci-dessous). Il m'est souvent arrivé que le logiciel VMware (pour créer des machines virtuelles) bloque les ports d'apache.



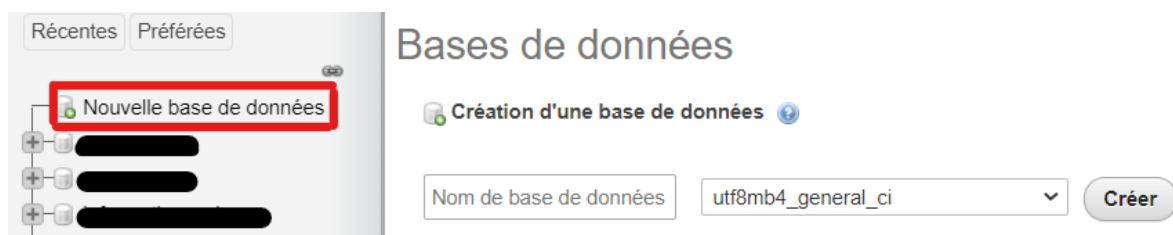
*Liste d'étapes pour stopper un programme empêchant apache de s'activer*

Après avoir arrêté le programme, on peut enfin relancer le module, il s'activera. Ensuite, il suffit d'aller sur un moteur de recherche, et d'écrire « localhost/phpmyadmin » et une page s'ouvrira et c'est ici que nous pourrons commencer la création de la base de données (image ci-dessous).



Page par défaut lors de la création de la base de données

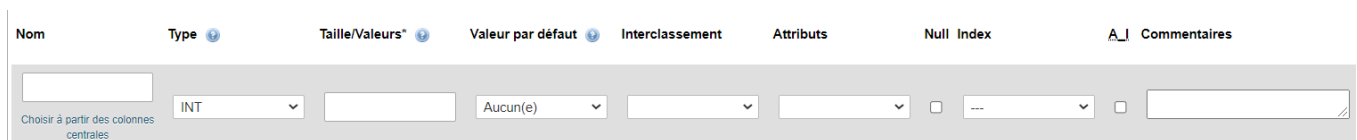
Cette page principale ne nous intéresse pas beaucoup, on y retrouve juste les informations des différents outils comme leur version par exemple. Cependant, pour créer une base de données, il faut d'abord cliquer sur « Nouvelle base de données » puis renseigner le nom (image ci-dessous).



Processus de création d'une base de données sous Xampp

Ici, « utf8mb4\_general\_ci » est le nom de l'encodage des données de la base. Il fait en sorte de pouvoir enregistrer et afficher les caractères spéciaux comme les « é » ou « à ».

Ensuite, comme il n'y a aucune table de créée, on nous propose de choisir un nom de table, et le nombre de colonnes dans celle-ci. Pour finalement nous rediriger sur un tableau :



Ligne d'initialisation d'une variable

C'est dans ce tableau qu'il faut insérer les données que nous voulons stocker dans cette table. Il faut y définir :

- Le nom de la variable
- Le type (INT pour nombre entier, VARCHAR pour une chaîne de caractères, BOOLEAN pour un booléen, DATE pour une date, etc.)
- La taille de la variable
- Sa valeur par défaut (par exemple, pour un booléen, on veut que sa valeur par défaut soit de 0 ou de 1)
- Son interclassement (son encodage)
- Ses attributs (s'il est binaire, ou si l'on veut la date actuelle par exemple)
- Si sa valeur est nulle
- Son index (primaire, unique ou rien)
- S'il est en auto-incrémentation (exemple : chaque fois qu'un élément sera ajouté dans cette table, l'identifiant augmentera de 1)
- Un commentaire, pour savoir à quoi correspond la variable

Après avoir rempli toutes les cases de chaque variable, on clique sur le bouton « exécuter » juste en dessous du tableau, et on recommence jusqu'à ce que toutes les tables soient créées, et que la base de données soit remplie.

De ce fait, nous avons à présent une base de données fonctionnelle et complète. De plus, étant en local et facilement modifiable, il sera aisé d'ajouter, de retirer ou de modifier des variables par la suite, pour permettre l'évolution de cette base de données dans le temps.

#### 4. Sécurité

Maintenant que nous avons terminé la mise en place, il faut penser à la sécurité de notre base de données.

Celle-ci est en local sur notre ordinateur, cela signifie que l'on ne risque pas grand-chose. De plus, nous ne sommes encore qu'aux prémices de l'application, et donc une attaque n'aurait pas grand effet, car rien d'important.

Cependant, si l'on veut avoir le plus sécurité possible, ce n'est pas pour tout de suite, mais pour avoir déjà un minimum de chances de se faire attaquer lorsque l'application et la base de données seront opérationnelles au grand public. C'est pour cela que l'on pense déjà à protéger notre système.

Tout d'abord, il faut changer le nom d'utilisateur et le mot de passe du compte administrateur ayant tous les droits. En effet, à la création d'une base de données, le nom de l'administrateur est « root », en plus de n'avoir aucun mot de passe. L'idéal étant de mettre tous les autres utilisateurs n'ayant que les droits de visionnage, pour plus de sécurité.

De plus, pour ne pas que les données que nous envoyons et recevons soient visibles par n'importe qui, il est important de créer une clé SSH, qui va nous permettre de crypter les données d'envois. Cependant, il est important de le faire sur la machine où est utilisée la base de données. Nous attendions d'avoir un serveur commun, pour la stocker et la protéger, mais nous n'en avons pas eu l'occasion. La base de données étant établie, je pouvais me lancer dans la conception du Backend de l'application.

### III. La mise en place du Backend

#### A. Présentation

La deuxième mission à réaliser était la mission principale de ce stage, ce pour quoi j'étais là. En effet, pendant que je finissais de créer la base de données, mes collègues stagiaires, qui étaient présents pour la partie visuelle du site web avaient déjà commencé la production de celui-ci. Je devais donc maintenant, faire interagir l'application, avec cette base de données. « Backend » est un terme désignant les éléments ne faisant pas partie de la face visible d'un site ou d'une application.

Cette application a plusieurs intérêts, dont un par type de client. Tout d'abord, il permettra aux candidats cherchant une mission, à renseigner leurs informations pour être plus tard contactés par l'équipe *AllYouNeed*. À la suite de cela, une fiche candidat sera créée, pour être envoyée à une ou plusieurs entreprises.

De plus, les entreprises cherchant des salariés pour effectuer une mission pourront également se créer un compte à travers un contact, pour que là encore, l'équipe *AllYouNeed* puisse les contacter pour plus d'informations sur leur demande.

Enfin, du point de vue d'un administrateur, on aura d'un côté la liste de tous les candidats inscrits, et de l'autre la liste de toutes les sociétés en contact avec *AllYouNeed*.

Puis on pourra voir les informations détaillées de chaque personne, la liste des missions d'une entreprise, ses chiffres d'affaires (dû aux missions) et ses contacts.

Elle aura pour but la simplification des échanges entre *AllYouNeed* et ses clients, mais également que la recherche des candidats soit plus efficace.

## B. Démarche

Pour développer le Backend, il fallait souvent être en communication avec les stagiaires s'occupant de la création du visuel de l'entreprise. En effet, le moindre changement de fonction ou de variable dans un fichier peut mener à une erreur de requête vers la base de données, ou peut faire en sorte que l'application n'affiche pas ce que l'on souhaite, et peut même planter.

Pour cela, j'ai commencé à développer en premier lieu la page d'enregistrement. Lorsque le candidat remplissait ses informations et s'enregistrait, elles étaient envoyées et enregistrées dans la base de données. Puis, sur la page de connexion, le programme vérifiait si le mot de passe renseigné était le bon. Si oui, il entrait dans l'application, sinon, on lui informait que son mot de passe n'était pas celui associé à son email.

Ensuite, nous avons les sections des listes : une section pour la liste des candidats, et une section pour la liste des entreprises. Lorsque l'on se connecte en tant qu'administrateur, la liste des candidats affiche tous les candidats inscrits dans la base de données, et la liste des entreprises, les affiche toutes également. L'administrateur peut alors les supprimer, les modifier, ou en ajouter. Cependant, lorsque je me connecte en tant que candidat, je n'ai pas accès à la liste des entreprises, et je ne vois que moi, dans la liste des candidats, la liste devient une visualisation de mon compte. Cela fonctionne de la même façon pour les entreprises. Enfin, dans les informations détaillées de l'entreprise, nous aurons les missions, contacts et chiffre d'affaires de ceux-ci.

Cependant, au début de la mission, nous n'avions pas assez communiqué entre nous, les stagiaires, sur notre manière de procéder, et notamment par rapport aux noms des variables que nous utilisons. Le nom de mes variables était en anglais. Je les ai initialisés dans cette langue selon la réglementation qui pousse à écrire les variables en anglais, pour ne pas avoir de problème avec les accents, pour suivre avec la langue principale des langages de programmation qui est l'anglais.

Comme les écrire dans cette langue est une réglementation, je pensais que tout le monde allait écrire ses variables en anglais. Cependant, une personne a décidé d'écrire toutes ses variables en français, et comme son code était un peu avancé, il ne voulait pas les changer, et a donc préféré les traduire directement dans son code.

Une autre problématique qui est souvent revenue au début de cette mission était que nous n'avions pas d'espace GitHub, et ne savions pas nous en servir. De ce fait, chaque fois qu'une personne faisait une importante modification, elle devait l'envoyer sur le groupe discord « AllYouNeed » que nous avons créé, et les autres devaient récupérer ce fichier et remplacer les anciens fichiers par leur nouvelle version. C'était assez compliqué et aussi confus, ce pour quoi nous avons alors fait une formation d'environ 4h sur l'utilisation de GitHub. GitHub est une plateforme de stockage et de partage de code, de projets.

Nous avons alors pu mettre notre projet dans un GitHub centralisé, où nous pouvions récupérer et envoyer nos modifications, qui se faisaient automatiquement sur GitHub, ce qui rendait le partage de modifications assez simple.

Enfin, pour la réalisation de cette mission, j'ai utilisé le langage qui m'était imposé, c'est à dire PHP, car c'est le langage qui s'allie le mieux avec HTML et CSS, pour la partie visuelle. Afin d'être sûr d'être à jour et d'avoir un bagage assez conséquent de connaissances en PHP, j'ai effectué une formation de 20h sur ce langage. J'étais donc prêt à me lancer dans la création du Backend de l'application *AllYouNeed*.

Au début de cette mission, nous avons mélangé le HTML et le PHP, car le langage PHP permet de modifier ce que l'on voit à l'écran. De plus, avec cette méthode, il était extrêmement pratique de changer l'aspect du site. Cependant, cela avait de plus gros inconvénients que nous pensions. En effet, il nous était alors plus compliqué de se retrouver dans le code du visuel du site. Ce pour quoi nous avons alors décidé de fragmenter et de détacher des codes. J'en parlerai en détail à la fin de la réalisation de la mission.

## C. Réalisation

### 1. La page d'enregistrement et de connexion

Tout d'abord, avant de pouvoir entrer dans l'application et voir les informations, il faut créer la page d'enregistrement. Les autres stagiaires avaient déjà créé le visuel de la page (Annexe 3), je n'avais plus qu'à la lier à la base de données. Pour cela, il faut d'abord créer un lien, entre notre code, et cette base. Ce lien sera introduit grâce à la fonction « `mysqli_connect` », qui va directement se connecter à la base de données :

```
$conn = mysqli_connect($db_host, $db_user, $db_password, $db_database, $db_port); // Connexion à la base de données
```

*Code de connexion à la base de données*

Sur cette capture d'écran, on peut voir que dans la fonction « `mysqli_connect` ». J'y insère des éléments. Ces éléments font référence à tout ce dont on a besoin pour se connecter à une base de données. Comme le nom de la base, le nom de l'utilisateur, son mot de passe, l'hôte du serveur (l'adresse IP), et le port de l'hôte. Si tous les éléments correspondent et que le programme arrive à se connecter à la base de données, on continue la lecture du script. Sinon, on informe que le programme n'a pas réussi à se connecter à la base de données.

Nous avons donc dès à présent la capacité de nous connecter à notre base de données. Nous allons donc l'écrire sur toutes les pages en langage PHP. Cependant, il faut avouer qu'un même code mis plusieurs fois dans des fichiers différents n'est pas très pratique, car si l'on a besoin de modifier une information dans cette partie, il faudra alors changer celle-ci sur chaque page. Pour remédier à cela, nous allons faire un fichier à part, appelé « conn\_bd », où nous allons y écrire notre script, puis nous allons l'importer dans chaque fichier, avec cette ligne :

```
include './conn_db.php';
```

*Insertion d'une page de code, dans une autre.*

Avec « include » puis l'endroit où se trouve le fichier, on importe celui-ci dans la page, ce qui évite les répétitions.

La page de connexion de la base de données enfin terminée, je peux alors me pencher sur la page de création de comptes, dont le visuel a été fait par les autres stagiaires. Il y a 3 différentes pages d'enregistrement : celle de l'administrateur (que nous avons faite pour ajouter plus rapidement un administrateur, mais qui sera supprimée par la suite), celle des entreprises et celle des candidats. Nous avons séparé ces pages, car le candidat ne doit pas rentrer les mêmes informations que l'entreprise. Je ne vais expliquer que le fonctionnement de l'enregistrement du candidat, car la méthode est la même pour l'administrateur et l'entreprise.

Tout d'abord, le candidat rentre ses informations dans les champs qui lui sont attribués. Puis, il va cliquer sur le bouton « envoyer ». À ce moment-là, le programme va regarder s'il reçoit le mail et le mot de passe (car les autres champs sont gérés par le JavaScript, langage utilisé par les autres stagiaires). S'il reçoit ces informations, il faut qu'il s'assure que personne d'autre n'a le même mail, car un mail est unique. Pour cela, il va lancer le code suivant :

```
$sql = "SELECT * from contact where email_contact='" . addslashes($_GET['mail']) . "'";
$resultat = mysqli_query($conn, $sql);
if ($resultat == FALSE) { // S'il y a une erreur dans la requête sql
    echo 'Erreur d\'execution de la requête';
} elseif (mysqli_num_rows($resultat) == 1) { // Si il y a déjà l'adresse mail dans la database --> le compte existe déjà
    echo 'cette email est déjà utilisé';
} elseif (mysqli_num_rows($resultat) == 0) { //S'il n'y a pas la meme adresse mail dans la table
```

*Capture d'écran d'une vérification de mail*

Le programme va regarder si l'email existe déjà dans la table *contact* en lançant une requête SQL (ici, les 2 premières lignes de l'image ci-dessus).



Ensuite, s'il y a eu une quelconque erreur dans la requête, le programme nous avertit. Sinon, il va regarder s'il récupère un résultat, et s'il n'en trouve aucun, c'est qu'il n'y a pas déjà l'adresse email dans la table *contact*. De ce fait, il va alors regarder dans les autres tables si l'adresse utilisée n'est pas déjà attribuée. S'il ne trouve rien dans chaque table, c'est que l'email n'a pas encore été enregistré dans la base de données, nous pouvons alors passer à l'enregistrement.

En premier lieu, pour éviter les injections SQL, c'est-à-dire le fait d'arriver à modifier notre base de données directement depuis le site, en rentrant des valeurs étranges dans les champs, nous allons mettre toutes les informations récupérées dans une fonction nommée « *htmlspecialchars* », comme nous le montre l'image ci-dessous :

```
$nom = htmlspecialchars($_GET['nom']);
```

*Exemple d'utilisation de la fonction « htmlspecialchars »*

Cette fonction fait en sorte de mettre tout ce qu'il y a dans la variable, en caractère de texte.

Après avoir fait cela pour toutes les variables, nous devons hacher le mot de passe. Hacher un mot de passe a pour but de ne pas l'afficher tel quel dans la base de données. De ce fait, si l'on se fait pirater, les intrus verront les mots de passe, mais ne comprendront pas ce qu'ils veulent signifier.

```
$2y$10$687qT8Pj7tC9pGXhUWoN0.9Qz5e/bZrwkPPR34nS7xvVFUth1o7SW  
$2y$10$1efAbDsgJu2bVv0vjYSu4e0VDuVJ0grq4NCS0v5hZeovIE09bhC8G
```

*Exemple de 2 mots de passe hachés*

Pour cela, il faut utiliser une fonction nommée « *password\_hash* » comme montrée ci-dessous :

```
$pass_hash = password_hash($mdp, algo: PASSWORD_DEFAULT);
```

*Exemple d'utilisation de la fonction « password\_hash »*



Nous pouvons alors inscrire notre utilisateur, en envoyant ses données dans la table *customer* dédiée aux candidats.

```
$sql = "INSERT INTO customer (name_customer, fname_customer, etc) values ('" . $nom . "', '" . $prenom . "', '" . etc . "')";
$resultat = mysqli_query($conn, $sql); //envoi la requête dans la base de données
```

*Requête permettant d'ajouter un candidat dans la base de données*

Puis comme à chaque requête, le programme va pouvoir nous renvoyer si la requête a fonctionné. Si aucun problème n'a été détecté, il nous fait savoir que l'utilisateur a bien été enregistré, sinon, il nous informe que la requête n'a pas abouti.

Viens ensuite la page de login, car à quoi bon vous enregistrer, si vous ne pouvez pas vous connecter sur l'application ?

Le système est simple : tout d'abord, comme pour l'enregistrement, le programme va récupérer les données transmises dans les champs de connexion (Annexe 4), et va chercher dans toute la base de données si l'email y est renseigné ou non. S'il ne le trouve pas, l'utilisateur va être notifié que son compte n'a pas encore été créé. En revanche, s'il trouve l'email envoyé, cela veut dire qu'un compte existe, et va donc vérifier si le mot de passe est le bon.

Pour cela, rappelons que le mot de passe inséré dans la base de données est haché, si l'on regarde directement le mot de passe de la base de données tel qu'il est, il n'est pas du tout le même que celui renseigné. Il existe une solution à cela : la fonction « `password_verify` ».

```
$mdp = $row['password_customer']; // "password_customer" est le nom de la variable du mot de passe dans la base de données
if (password_verify($_POST['mdp'], $mdp)) { // "mdp" est le nom de la variable du mot de passe que le client envoie pour se connecter
```

*Utilisation de la fonction « `password_verify` »*

Dans cette fonction, on y renseigne le mot de passe haché (ici, dans la base de données), et celui que l'on veut faire correspondre. La fonction va retourner « `true` » si ce sont les mêmes mots de passe, et « `false` » s'ils ne le sont pas. Si les mots de passe correspondent, cela signifie que le mail et le mot de passe renseignés sont tous les 2 bons, et l'utilisateur est donc connecté. Selon la table où se trouve l'email, nous allons assigner un type à l'utilisateur, pour savoir s'il est un administrateur, un candidat, ou contact d'une entreprise.

```
$_SESSION['mdp_customer'] = $_POST['mdp_customer'];
$_SESSION['type'] = "customer"; // On détermine le type de l'utilisateur
```

*Assignation d'un type à l'utilisateur venant de se connecter*

Maintenant que nous arrivons à nous connecter, il faut donner la capacité à l'utilisateur de se déconnecter. Pour cela rien de plus simple, lorsque l'utilisateur cliquera sur le bouton dédié à la déconnexion le programme effectuera ce code :

```
session_start(); //Lance la session dans laquelle on se trouve
session_destroy(); // Détruit cette session
```

*Script de déconnexion*

Le programme va alors lancer la session de l'utilisateur connecté, et va la supprimer juste après. Car si l'on utilise la fonction « session\_destroy » sans avoir au préalable récupéré la session de la personne connectée, alors il ne peut pas savoir quelle session supprimer. Maintenant que les scripts de connexion, enregistrement et de déconnexion sont créés, nous pouvons passer aux listes.

## 2. La liste des candidats et des entreprises

Ici, comme pour la page d'enregistrement, le code est sensiblement le même que ce soit pour la liste des candidats, ou la liste des entreprises. Cependant, les informations que le programme va aller chercher ne sont pas les mêmes si l'utilisateur est un candidat, entreprise, ou administrateur.

En effet, lorsqu'un administrateur se connecte au site, il voit la liste de tous les candidats dans la rubrique *Candidat* (Annexe 5) ou toutes les entreprises dans la rubrique *Entreprise*. Mais au contraire, lorsqu'un candidat ou une entreprise se connecte, ils ne verront seulement que leurs propres informations, et ne pourront pas voir tous les autres comptes de l'application.

Pour cela, nous allons utiliser les « rôles », initiés plus haut. Le programme reconnaîtra alors le rôle de l'utilisateur, et affichera les listes en conséquence. Bien sûr, s'il ne reçoit pas de rôle, il sera demandé à l'utilisateur de se connecter.

```
if ($_SESSION['type'] == "admin") { // = Si l'utilisateur est un admin
```

*Vérification du type de l'utilisateur*

Si l'utilisateur est bien un administrateur, le programme va alors afficher tous les clients, ou toutes les entreprises grâce à cette boucle :

```
while ($row = mysqli_fetch_assoc($resultat)) {
    echo '<tr>'; //Ouvre une nouvelle ligne dans le tableau sur le site
    echo '<td>$row["fname_customer"]</td>'; //affiche le nom
    echo '<td>$row["name_customer"]</td>'; // le prénom
    echo '<td>$row["num_customer"]</td>'; // le numéro de téléphone
    echo '<td>$row["email_customer"]</td>'; //l'adresse email
    echo '<td>$row["creation_date"]</td>'; //La date de création du compte
    echo '<td>$row["score_customer"]</td>'; //Le score du client
    echo '<td><a href= "../html/fiche_candidat.html">Fiche détaillée</a></td>'; // Le lien vers la fiche détaillée
    echo '</tr>'; // ferme la ligne
}
```

#### *Boucle d'affichage des informations du candidat*

Cependant, si l'utilisateur n'est pas un administrateur, le script va alors faire une requête à la base de données pour ne récupérer que les informations de la personne connectée. Pour cela, dans la requête, nous allons préciser que nous voulons chercher un utilisateur précis :

```
$sql = "SELECT * FROM customer where idcustomer='" . $_SESSION['id_user'] . "'"; // Ne demande l'information que de l'utilisateur qui s'est connecté
$resultat = mysqli_query($conn, $sql); //Envoie la requête dans la base de données
```

#### *Requête pour trouver les informations de l'utilisateur actuel*

Nous récupérons donc l'identifiant de l'utilisateur, qui le rappelons-le, est unique, et l'ajoutons dans la requête, qui fera donc office de filtre. Puis, nous afficherons ses informations. C'est là le même processus que pour afficher la liste des candidats, excepté qu'ici, il n'y en a qu'un seul.

Maintenant que nous arrivons à afficher tous les candidats et entreprises, il est nécessaire de développer le code qui servira à en ajouter, modifier, ou encore supprimer. Tout ceci sera utilisé grâce à la fiche détaillée du candidat (Annexe 6).

### 3. Fiche détaillée du Candidat

Comme son nom l'indique, la fiche détaillée d'un candidat affichera toutes les informations de celui-ci. De plus, on pourra depuis cette page, en ajouter ou en modifier un. La suppression se fera dans le même fichier, mais elle sera activable dans la liste générale.

Le but de cette fiche est d'avoir une représentation précise de la personne, qui elle est, ce qu'elle a fait et veut faire, ses compétences, et tout ce qui tourne autour d'elle. Seul l'administrateur a la possibilité de voir cette page.

Lorsque l'on remplit les informations dans cette fiche détaillée, le fichier JavaScript va envoyer un message à notre programme, nous informant si c'est la 1re fois que l'on remplit cette page ou si c'est une modification d'une page déjà créée.

```
elseif ($_GET['action'] == 'ajouter') { // Si le javascript nous dis qu'on ajoute un client
    $sql = "SELECT * FROM customer where email_customer='" . $_GET['mail'] . "'"; //Recherche si l'email est déjà dans la base de données
    $resultat = mysqli_query($conn, $sql); // Envois la requête à la base de données
    if ($resultat) {
```

*Script où l'on reçoit une demande d'ajout*

L'action d'ajouter un candidat étant reçue, on vérifie d'abord que le candidat que l'on souhaite ajouter ne figure pas déjà dans la base de données. Comme à notre habitude, le programme nous notifiera si la requête a échoué, ou si le candidat existe déjà. S'il n'existe pas dans cette base de données, le programme va l'ajouter grâce à cette ligne :

```
$sql = "INSERT INTO customer (name_customer, fname_customer, etc) VALUES ('" . $name . "', '" . $fname . "', 'etc')"; //Fait la requête ajoutant le client dans la base de données.
$resultat = mysqli_query($conn, $sql); //Envois la requête
```

*Enregistrement d'un nouveau candidat depuis la liste détaillée*

Le candidat étant maintenant enregistré, si nous modifions quelques-unes de ses informations, la page JavaScript va nous avertir que l'action est une modification, et non un ajout :

```
elseif ($_GET['action'] == 'modifier') {
```

*Demande de modification du candidat*

La modification, étant sollicitée, nous allons faire une requête pour modifier un candidat selon son identifiant :

```
$sql = "UPDATE customer SET
    name_customer='" . $name . "', //changement du nom
    fname_customer='" . $fname . "', //changement du prénom
    etc
    WHERE idcustomer='" . $idcustomer . "'"; //Faire ces changements sur le client avec un identifiant particulier
```

*Modification d'un candidat*

Sur l'image ci-dessus, nous modifions un candidat à l'aide de son identifiant. En effet, si l'on demande juste à la base de données de modifier le nom, le prénom, sans préciser à qui il faut les modifier, tous les candidats vont alors se retrouver avec les informations que nous avons saisies. Une fois encore, l'identifiant sert de filtre, pour dire à notre requête à quel candidat en particulier il faut modifier ces données.

L'ajout et la modification effectuée, il ne reste plus que la suppression du candidat. Cette partie a été la plus facile à produire, car elle ne demande que l'identifiant du candidat pour savoir lequel supprimer, car sinon, tous les candidats seraient supprimés de la base de données.

```
elseif ($_GET['action'] == 'supprimer') { //Si le javascript nous renvoie qu'il faut supprimer un client
    $sql = "DELETE FROM customer WHERE idcustomer=" . $_GET['idcustomer'] . " "; // Requête pour supprimer le client grâce à son identifiant
```

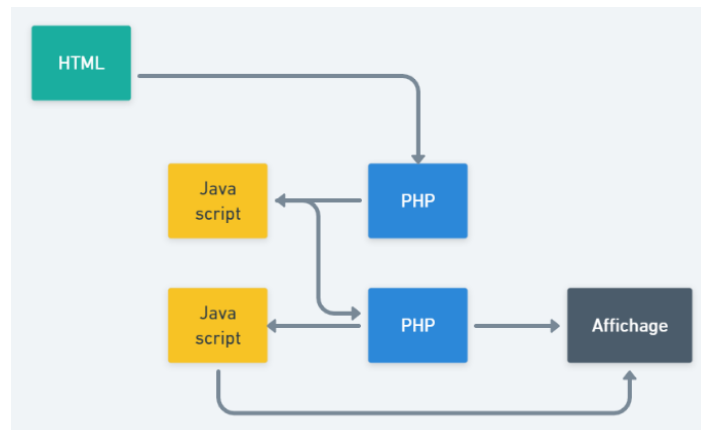
#### *Suppression d'un candidat*

Comme à notre habitude, dans la requête envoyée à la base de données, nous filtrons cette requête avec un identifiant pour rechercher et supprimer un candidat précis. La fiche du candidat détaillée étant finie, celle de l'entreprise, ou du moins son fonctionnement est presque identique à celle du candidat. Les seules différences étant les noms des variables.

Cependant, au cours de ce projet, nous nous sommes rendu compte qu'afficher des éléments depuis le PHP n'était pas la méthode la plus optimisée, et nous avons donc commencé à modifier la structure du code, pour arranger tout ça.

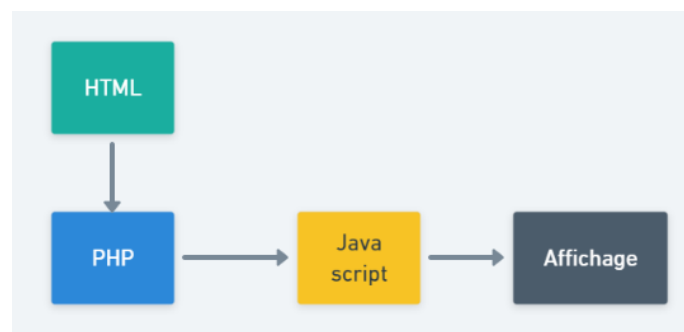
## 4. L'optimisation des échanges entre les fichiers

Une bonne optimisation est importante. Cela permet de réduire le temps de latence de chargement d'une page web ou d'une application d'une durée parfois non négligeable. C'est pourquoi nous avons pris notre temps pour mettre en pause la progression du projet et pour nous pencher sur l'amélioration de ce qui s'y trouvait déjà. Car si nous continuions de développer l'application comme nous le faisons à ce moment-là, il y aurait eu plus de modifications à faire par la suite. En effet, l'application faisait des répétitions de requêtes entre le PHP et le JavaScript, comme illustrées dans l'image ci-dessous :



*Représentation de la structure des échanges entre les fichiers avant l'optimisation*

Le but de cette optimisation est de faire en sorte qu'il n'y ait qu'un échange par fichier, comme schématisé dans l'image si dessous :



*Représentation de la structure des échanges entre les fichiers après l'optimisation*

Pour cela, dans nos fichiers, nous allons modifier les bouts de code où nous affichons des éléments sur la page web. À la place de ceux-ci, nous effectuerons des requêtes Ajax vers le JavaScript. Une requête Ajax est une manière de faire passer des informations entre des fichiers à travers une page web. Elle permet d'effectuer des processus sans interrompre le chargement de celle-ci.

Dans ce but, chaque fois que nous devrons afficher des informations dans notre application, nous les ferons passer à travers des requêtes Ajax, qui seront sous forme de tableaux de cette forme :

```

$table = array( // On initialise le tableau
    'error' => true, // Si l'on veut remonter une erreur dans la requête
    'message' => 'Erreur d\'exécution de la requête1', // Message d'erreur
); // Fermeture du tableau
$table_encode = json_encode($table, flags: JSON_INVALID_UTF8_IGNORE); // Transforme le tableau pour faire en sorte qu'il puisse passer en Ajax
echo $table_encode; // Envoie le tableau au javascript à travers la requête ajax
  
```

*Exemple de tableau que nous enverrons à travers une requête Ajax*

Après avoir entré les informations que nous voulons faire passer dans ce tableau, il faut à présent le transformer pour qu'il soit capable d'être communiqué à un fichier JavaScript grâce à la requête Ajax. C'est dans ce but que l'on utilisera la fonction "json\_encode". On y placera à l'intérieur de celle-ci, le tableau en question puis la façon dont on veut l'encoder. On envoie ensuite ce tableau grâce à l'utilisation de "echo". De ce fait, le tableau s'envoie au JavaScript à travers les requêtes Ajax, et le JavaScript s'occupera de l'afficher. Au terme de ces changements, j'ai reçu comme directive d'essayer d'intégrer Symfony à notre application.

## 5. La migration de l'application en Symfony

Symfony est un ensemble d'outils et de composants logiciels, aussi appelé « framework ». Il permet entre autres de faciliter le développement d'un site web, en automatisant la création de pages, formulaires et requêtes vers la base de données. Nous avons besoin d'utiliser Symfony, car il permet de sécuriser plus facilement une application ou un site web, mais également d'avoir une maintenance plus rapide. Facilitant la compréhension et la modification d'un fichier.

Pour cela, j'ai effectué une formation de vidéo d'à peu près 20h, m'apprenant comment utiliser ce framework. Cependant, en suivant la formation, le formateur utilisait Symfony avec le langage PHP orienté objet. Un langage orienté objet est une manière d'utiliser un langage informatique différente de la méthode « classique » (dont le terme précis est « procédural »). C'est-à-dire de haut en bas de la page, comme on pourrait lire un livre.

La formation en PHP suivie en début de stage faisait mention de l'utilisation orientée objet, mais pas aussi poussée que ce que Symfony exigeait. De ce fait, je devais souvent mettre les vidéos en pause, pour comprendre ce que le formateur disait, mais également pour rattraper son rythme. N'ayant qu'un seul écran, je ne pouvais suivre la formation et coder en même temps. La formation censée durer 20h s'est transformée en formation d'une semaine.

Cependant, après avoir fini cette formation, il ne me restait pas beaucoup de temps avant la fin du stage. De ce fait, ayant rencontré beaucoup de problèmes dont je mettais du temps à résoudre, car n'ayant pas assez d'expérience d'utilisation de Symfony, je n'ai malheureusement pas réussi à l'utiliser dans notre application. Mais avant cela, nous avons reçu une autre mission : la recherche d'API.

## IV. Utilisation d'API

### A. Présentation

Durant la création du site, M. Maxime ZANCHI a demandé à toute l'équipe de faire des recherches sur des API. Une API est une solution informatique qui a la capacité de lier des services et des applications. Cela permet entre autres d'aller chercher et de récupérer des informations ou des services d'une application chez des entreprises tiers.

Il existe des API pour tous types de services : envoyer des messages par téléphone et par email, faire un système de paiement bancaire, créer ou modifier un calendrier et bien d'autres.

L'idée derrière cette recherche était de pouvoir automatiser un envoi de messages, de mails, et de mise en place de réunions dans le calendrier Microsoft et Google à travers l'application. Lorsque l'utilisateur s'inscrit sur l'application, il pourrait recevoir un mail signifiant que son compte a bien été créé. Ou alors, le jour de l'anniversaire d'un candidat, il pourrait recevoir un message sur son téléphone de la part d'*AllYouNeed*, lui souhaitant son anniversaire, ou encore un message d'encouragement lorsqu'on sait qu'il aura un entretien dans la journée. Cela pourrait grandement renforcer les liens entre le candidat et son agent, ce qui n'est presque pas le cas, dans la plupart des entreprises de recrutements.

Cependant, aucun d'entre nous n'avait déjà utilisé une API. Nous ne savions pas comment fonctionnait ce service. Pour cela, nous avons dû faire des recherches.

### B. Démarche

N'ayant aucune connaissance en matière d'API, je décide alors de faire des recherches sur le net. J'y cherche principalement des informations sur quelles API utiliser pour les services dont on a besoin. Nous avons 3 services à vouloir utiliser dans l'application :

- L'envoi de messages sur un téléphone
- L'envoi de mail
- La génération d'évènements dans un calendrier

Les API sont très utilisées dans le domaine de l'informatique. Donc nous pensions que nous trouverions assez d'informations rapidement pour les utiliser. Néanmoins, il y avait tellement d'API différentes pour le même service, que nous ne savions pas laquelle choisir. De plus, d'une API à l'autre, le langage utilisé n'était pas



le même. Je cherchais donc une API à utiliser avec le langage *PHP*, pour l'intégrer dans notre application.

Ayant des difficultés avec certaines documentations, car les trouvant trop compliquées pour un néophyte comme moi. Je décide de rechercher un tutoriel d'une API de messagerie en *PHP* sur la plateforme de vidéo *YouTube*. Je trouve alors une vidéo d'une personnalité dont les compétences sont reconnues. Je décide de suivre le tutoriel pour comprendre comment cette API fonctionne. L'API de messagerie dont j'ai choisi d'utiliser vient de *Twilio*. *Twilio* est une entreprise de services téléphoniques pour développeur et entreprise, il envoie des messages et photos par SMS et peut également être capable d'appeler quelqu'un. J'ai choisi celui-ci, car étant reconnu comme un bon service téléphonique, la façon dont on peut l'utiliser est extrêmement aisée. On a également la possibilité de tester les envois de messages vers une seule personne, le tout gratuitement, tandis que la plupart des autres services nous demandent de payer avant même de pouvoir les tester.

De même avec l'API pour l'envoi d'emails. Cette fois-ci, un tutoriel à un service d'email nommé *Mailjet* était déjà intégré dans une formation. Elle faisait office de supplément. Ayant fait des recherches sur d'autres API d'emails, j'ai choisi de garder *Mailjet* car un tutoriel précis nous était mis à disposition. De plus, comme pour *Twilio*, il nous était possible d'envoyer des emails de tests gratuitement, pour vérifier que cela fonctionnait bien. Enfin, ce qui m'a fait décider de choisir celui-ci, c'est le fait que nous pouvons créer des visuels directement sur le site web, le passer à travers notre programme pour l'envoyer par email. Ce système peut paraître anodin, mais cela permet à des commerciaux ou des designers de créer le visuel du mail, sans écrire une seule ligne de code. Enfin, nous avons également une page destinée aux statistiques, pour voir qui a ouvert le mail, qui a cliqué sur le lien contenu dans celui-ci, les mails ayant été placés dans les courriers indésirables, etc.

Cependant, un des stagiaires ne trouvant aucune information technique sur une API pour créer un événement dans le calendrier de *Microsoft* me demande son aide. En recherchant sur le Net, il est vrai que seule la documentation officielle de *Microsoft* semblait à jour, et expliquait comment l'utiliser. Malgré cela, cette documentation n'était pas très pratique, car il y avait beaucoup de termes spécifiques à *Microsoft* que nous ne comprenions pas très bien. De plus, il semblerait qu'il n'y ait pas une API spéciale pour le calendrier, mais plutôt pour tout le système *Microsoft* (*Word*, *Excel*, *Notes*, *Calendar*, *PowerPoint*), dont le calendrier est une toute petite partie, ce qui ne nous simplifie pas la tâche. Pour utiliser l'API, il fallait se connecter à un site spécial de *Microsoft*, se connecter avec notre compte entreprise puis y inscrire une clé de compte et d'entreprise. Cependant, nous n'avons jamais réussi à les récupérer, car les endroits où la documentation nous demandait d'aller n'existaient plus. De ce fait, après une journée et demie de recherches, nous avons mis de côté les recherches sur cette API, et par manque de temps, n'avons pas pu y retourner pour continuer les recherches.

Les API que nous avons trouvés et réussis à faire fonctionner ne sont pas encore dans l'application, car cela aurait demandé encore une ou deux semaines de travail. Je vais rédiger un résumé simplifié de leur fonctionnement, dans la prochaine partie réalisation.

## C. Réalisation

### 1. API SMS : Twilio

Comme énoncé précédemment, *Twilio* est un service d'envois de messages, de photos et d'appels téléphoniques. De ce fait, pour pouvoir envoyer des messages, il faut un numéro de téléphone. Pour cela, nous n'avons pas besoin d'utiliser notre numéro personnel, *Twilio* en fournit un pour nous. Il y a 2 sortes de numéros. Les numéros payants, et gratuits. Les numéros gratuits sont basiques : ils nous sont attribués, nous ne pouvons envoyer des messages à des utilisateurs ayant entré leur numéro sur le site internet de *Twilio*, pour accepter de recevoir des SMS. De plus, la personne recevant le message verra systématiquement « Twilio » comme nom d'expéditeur, et la première phrase du message sera « Ceci est un message envoyé par Twilio ». De l'autre côté, nous pouvons choisir quel numéro payant nous voulons, nous pouvons envoyer des messages à n'importe qui, changer le nom de l'expéditeur par le nom de l'entreprise et écrire n'importe quel message. Cependant, comme nous utiliserons ce service comme test pour un temps, nous allons rester sur le modèle gratuit.

Le code pour utiliser cette API ne se trouve pas sur le site de *Twilio*. Il faut aller sur un lien externe et télécharger un dossier depuis GitHub et extraire celui-ci à la racine de notre projet. Deux éléments importants se trouvent dans ce code.

En premier lieu, il faut d'abord se connecter à notre compte depuis notre programme. Pour cela, il faut utiliser 2 informations : notre numéro de compte, et notre identifiant d'authentification. Car oui, nous ne pouvons pas utiliser l'API s'il ne sait pas avec quel numéro envoyer le message.

Ensuite, il faut créer un nouveau message, qui va être envoyé au destinataire :

```
$client->message->create( //Indication que l'on veut envoyer un message
    '+336XXXXXXX', //Numéro du destinataire
    [
        'from' => '+336XXXXXXX', //Numéro de l'expéditeur
        'body' => "Ceci est un message de test !", //Message que l'on veut envoyer
    ]
);
```

*Exemple d'utilisation de Twilio pour envoyer un message*

Pour créer et envoyer ce fameux message, il faut, comme montré si dessus, prévenir au programme que l'on veuille envoyer un message avec la fonction « create ». Ensuite, il faut indiquer le numéro de la personne à qui l'on veut envoyer un message, car sinon l'API ne saura pas à qui l'envoyer. Vient à cela notre numéro de téléphone sur le site *Twilio* (ce n'est pas notre numéro de téléphone personnel, mais bien le numéro avec lequel nous allons envoyer les messages automatiquement). Enfin, comme nous voulons envoyer un message, il faut y indiquer le message que nous voulons envoyer.

De ce fait, si par exemple, nous effectuons cela après que l'utilisateur s'est inscrit, il recevra alors un message automatisé, lui annonçant la bienvenue dans l'application. L'intégration de l'API pour envoyer des messages étant effectuée, nous pouvons maintenant passer à celle des emails.

## 2. API email : Mailjet

Comme pour *Twilio*, *Mailjet* est une entreprise fournissant un service d'envois de messages, mais ici par mail. Mais contrairement à celui-ci, ce service va utiliser l'email que l'on a inscrit lors de notre inscription. Lorsque l'on crée un compte, *Mailjet* nous demande si nous voulons utiliser le service à travers une API pour les développeurs, ou à travers des modèles modifiables pour ceux n'ayant pas de connaissances en développement web. En choisissant l'API, *Mailjet* nous affiche alors les directives pour l'utilisation de son API.

En Premier lieu, il faut télécharger un dossier qui va nous servir à utiliser ce service. Pour cela, il faut utiliser une certaine commande dans le terminal :

```
composer require mailjet/mailjet-apiv3-php
```

*Commande pour installer un dossier Mailjet*

Sur cette commande, vous pouvez y voir « composer », « require » et le nom du dossier que nous voulons utiliser. *Composer* est un logiciel tiers, permettant, comme dans notre cas, d'installer des dossiers ou bibliothèques que nous pouvons trouver sur le web, pour améliorer notre application. « require » est une fonction de *Composer*, qui veut tout simplement dire « télécharger ».

Lorsque cela est fait, nous pouvons utiliser ce que *Mailjet* nous fournit alors, le code pour envoyer des emails. Il est extrêmement similaire à celui que nous avons utilisé pour *Twilio*. Tout d'abord, nous devons indiquer notre numéro de compte, ainsi

que notre clé privée, disponible dans notre espace de compte. Ensuite, il faut indiquer le destinataire, l'expéditeur, et le contenu du message :

```
'Messages' => [
  [
    'From' => [
      'Email' => "remi.staelen@epsi.fr", //Email de l'expéditeur
      'Name' => "Remi staelen" // Nom de l'expéditeur
    ],
    'To' => [
      [
        'Email' => "augustin@marie@epsi.fr", //Email du client
        'Name' => "Augustin Marie" // Nom du client
      ]
    ],
    'Subject' => "Envois d'email à travers Mailjet !", //Objet du mail
    'TextPart' => "Ceci est un email de test ! L'as tu reçu ?", //Contenu du mail
  ]
]
```

*Script d'envois de mail à l'aide de l'API*

Sur l'image si dessus, nous pouvons voir que nous devons d'abord y inscrire le nom l'email et le nom de la personne envoyant le mail, puis celle de la personne le recevant. Enfin, il faut y indiquer l'Objet du mail (sur l'image, la partie « Subject »), pour éviter qu'il ne se retrouve dans les courriers indésirables et enfin le contenu du message (ici, la partie « TextPart »).

Si par exemple nous sommes dans une plus grande entreprise, et qu'une personne chargée du service marketing en vient à créer un visuel de mail, et l'a enregistré sur le site de *Mailjet*, ce visuel a alors un identifiant. Nous pouvons prendre celui-ci et l'indiquer à notre code à la place de la section « TextPart », qui va alors utiliser notre visuel en tant que message.

Nous avons alors dès à présent les connaissances pour être capables d'envoyer des SMS et emails automatiquement aux candidats.

## V. Évaluation des réalisations et des compétences mobilisées

### A. Adéquation du travail

Lors de notre stage, M. Maxime ZANCHI nous demandait parfois de présenter ce que nous avions fait, notre avancement. Il nous disait souvent que nous faisons du bon travail. Cela nous motivait encore plus de continuer dans cette voie, et nos résultats en ont été impactés en positif.

En effet, même si ce que nous faisons n'était que la 1<sup>re</sup> version de l'application, nous étions en train de donner vie à un réel projet, qui sera utile à l'entreprise. Dans le futur, cette application permettra aux employés d'*AllYouNeed* de travailler plus efficacement, mais également aux entreprises d'avoir un outil, permettant de se rapprocher de ses employés.

Cependant, toute l'idée de cette application et de comment elle fonctionne, sort de l'esprit de son fondateur. De ce fait, nous avons parfois un peu de difficulté à visualiser le produit final, car nous n'avons pas de réel cahier des charges, ni d'écrits ou de croquis sur ce projet. Il aurait été plus pratique, je pense, de mettre sur papier toutes les idées liées à l'application, pour en faire un document dont nous nous baserions.

### B. Compétences mises en œuvre

Tout au long de ce stage, en travaillant en groupe et sur de nouvelles technologies, j'ai utilisé, au cours de ces 7 semaines de nombreuses compétences.

En premier lieu, mes connaissances en *PHP* m'ont permis d'être assez efficace sur le plan développement. Car ayant reçu une formation avant ce stage, pour me refaire une piqure de rappel sur cette technologie, j'ai pu me lancer dans le développement de l'application assez rapidement, et sans problèmes apparents. Bien sûr, il y avait quelquefois où le programme était plus difficile à construire, mais j'ai réussi à utiliser encore une fois mes compétences en algorithme pour m'en sortir.

De plus, ayant eu la chance de travailler avec des personnes compréhensives et communicatives, nous avons pu travailler en équipe de manière efficace, où chaque problème rencontré était vite résolu. Du fait de notre vitesse de communication, mais également par la justesse d'une information.

Les missions réalisées m'ont permis d'accroître mon efficacité en production. En effet, au fil du temps, on m'a appris des méthodes, que ce soit logiciel, ou psychologique, pour aller plus vite dans ce que je faisais. J'ai changé plusieurs fois d'environnement de développement, jusqu'à trouver celui qui me correspond le mieux : ceux de JetBrains. De plus, j'ai appris à travailler avec la méthode Pomodoro. Pomodoro est une méthode de travail où l'on se concentre pendant 45min, puis on fait une pause de 15 minutes. Cela permet de ne pas être fatigué trop rapidement, ce qui

m'arrivait assez souvent, aux alentours de 15, et ce qui baisser drastiquement ma productivité à partir de cette heure-ci. En adoptant cette méthode de travail, j'ai pu améliorer ma manière de travailler.

Pour travailler comme Concepteur-Intégrateur DevOps, je devrais accentuer mes recherches dans divers langages. Je connais certes les langages de développement web (HTML, CSS, PHP, Javascript), mais je n'ai pas suffisamment de connaissances dans d'autres technologies. Le langage orienté objet est également une des compétences sur lesquels je devrais me pencher, car c'est la façon la plus idéale pour programmer, car cette manière de faire est très pratique, pour réutiliser des fonctions par exemple.

Ensuite, je devrais développer ma communication avec les autres. Comme dit plus haut, je sais travailler en équipe, prendre leur point de vue en considération dans ce que je fais, mais je n'arrive pas encore à prendre la parole et à diriger un groupe vers une voie précise. Je n'ai que suivi cette voie pour le moment, je ne l'ai pas tracée. C'est une des compétences dont je travaillerai le plus, dans l'intervalle avec le stage de l'année prochaine.

## VI. Conclusion

En conclusion, cette expérience professionnelle m'a été bénéfique de bien des façons.

En premier lieu, nous avons été très bien accueillis. Le personnel était à notre écoute, comme si nous étions des employés à part entière de l'entreprise. On ne nous demandait pas de tâches ingrates comme l'image que l'on pourrait se faire de stagiaires de premières années. Au contraire, ils nous ont mis au cœur du projet de l'application *AllYouNeed*, nous faisant totalement confiance. Et je pense que cela a pu se faire ressentir dans notre façon de travailler et dans les résultats que nous avons produits. De plus, nous avons un panel d'outils à disposition pour réfléchir à la conception de nos missions : tableau, formations, gestion de projet en ligne, GitHub centralisé. Il est dommage de ne pas avoir pu avoir un 2<sup>e</sup> écran, cela aurait sans doute augmenté notre efficacité.

Ensuite, avec toutes les missions que nous avons effectuées, je n'ai pas du tout l'impression d'avoir perdu mon temps, où d'avoir fait des missions basiques, comme ce que l'on nous disait en milieu d'année. J'ai eu, au contraire, l'impression de participer à la création de quelque chose de concret, qui répond à un réel besoin, et non de quelque chose de superficiel.

Les missions réalisées m'ont permis de prendre conscience des réels besoins d'une entreprise. De plus, j'ai pu y apprendre les langages et leurs utilisations dans un projet informatique. On n'utilise jamais un langage seul pour tout développer de A à Z. C'est un mélange de plein de langages différents que l'on utilise pour ce qu'ils font de

mieux et qui se complètent. Ceux à quoi l'on ajoute l'utilisation d'API et de framework pour un fonctionnement optimal.

J'y ai aussi développé un sens de la recherche. En effet, il y avait beaucoup de choses dont je ne savais pas encore de grandes connaissances. Pour remédier à cela, il fallait simplement que je trouve et comprenne une documentation sur ce que j'étais en train de chercher. Cela peut paraître aisé en première vue, mais il est beaucoup plus compliqué de lire des documentations sur ce que l'on ne comprend pas. Par exemple, ce serait comme lire un dictionnaire en ne sachant pas bien lire au départ. À force d'entraînement et de volonté, on y arrive, mais le chemin pour y parvenir n'a pas été de tout repos. Mais grâce à cela, on en ressort avec des connaissances plus élargies et étant donné que la plupart des documentations fonctionnent de la même manière, si l'on sait en lire en comprendre une, on sait le faire avec presque la totalité d'entre elles.

Ce stage a également contribué à nous initier au travail d'équipe en entreprise. On pourrait facilement croire que ce n'est pas différent du travail d'équipe lors de certains travaux de groupe à l'EPSI. Cependant, ici, chacun d'entre nous a un travail propre, et une responsabilité pour la mission qu'il doit mener. Bien sûr, on s'aide entre nous si quelqu'un a besoin d'aide, car il est bloqué, ou qu'il manque de temps. Mais il y a également cette responsabilité où si l'on est en retard, cela impacte tout le projet.

Pour terminer ce rapport, je souhaite m'exprimer sur les prochaines expériences professionnelles que je serai amené à faire dans le futur. J'aimerais, si possible, faire un stage ayant un lien avec le « Big Data » (Le Big Data est l'ensemble des données, nécessitant plusieurs machines pour les traiter) ou la data science en général (la data science est un terme employé pour désigner le fait d'utiliser des données acquises, pour répondre à des besoins d'une entreprise), car je souhaite, à terme, travailler dans le domaine de l'intelligence artificielle, qui se repose sur le principe de récupérer des données et de les traiter.

L'intelligence artificielle est un domaine assez récent, qui est voué à se développer de plus en plus vite, jusqu'à toucher la quasi-totalité des entreprises. En effet, la masse de données qui transite autour du globe augmente petit à petit, ce qui a pour conséquence directe que nous avons de plus en plus de données à traiter. De plus, au plus une intelligence artificielle a de données à sa disposition, au plus elle sera efficace, et donnera des résultats précis. C'est pour cela que dans le futur, l'intelligence artificielle aura une si grande masse de données à sa disposition, que sa sollicitation et son utilisation seront presque obligatoires si une entreprise veut résoudre des besoins, ou se développer efficacement.

C'est la raison pour laquelle j'aimerais effectuer prochain stage dans ce domaine, pour développer mes connaissances, et pour découvrir comment fonctionne ce métier.

## VII. Glossaire

**AllYouNeed :** Nom du groupe dans lequel j'ai effectué mon stage, liée aux sociétés E-Darwin et Pelops.

**API :** Ensemble normalisé de méthodes servant de façade par laquelle un logiciel offre des services à d'autres logiciels.

**Base de données :**

Stocke des données sous forme de tables, pouvant être reliée entre elles.

**Candidat :** Personne cherchant un poste dans une entreprise.

**Composer :** Logiciel de gestions de bibliothèques de dossiers et fichiers dans un projet.

**Dictionnaire de données :**

Un dictionnaire de données est une liste où est regroupée l'entièreté des données. On y inscrit également toutes les informations de chaque élément.

**E-Darwin :** Société partenaire d'AllYouNeed, proposant des services de coaching professionnels.

**Framework :** Ensemble d'outils et de composants informatiques à la base d'un logiciel ou d'une application.

**HTTP :** HyperText Transfert Protocol est un protocole qui a pour but de transmettre des documents entre les navigateurs web.

**Injection SQL :**

Ensemble de méthodes exploitant une faille de sécurité d'une application interagissant avec une base de données.

**MCD :** Le modèle conceptuel de données est une manière d'imager une base de données, il permet de la représenter sous forme blocs, avec des relations entre chacun d'eux.

**Pelops :** Société partenaire d'AllYouNeed, qui est en train de développer l'application Tailor.

**Requête Ajax :**

Méthode permettant d'effectuer des requêtes au serveur web et de modifier une page web, sans devoir recharger la page.

**Siret :** Code Insee permettant l'identification d'un établissement ou d'une entreprise.

**Requête SQL :**

Interrogation d'une base de données pour faire de multiple action.



## VIII. Bibliographie et Webographie

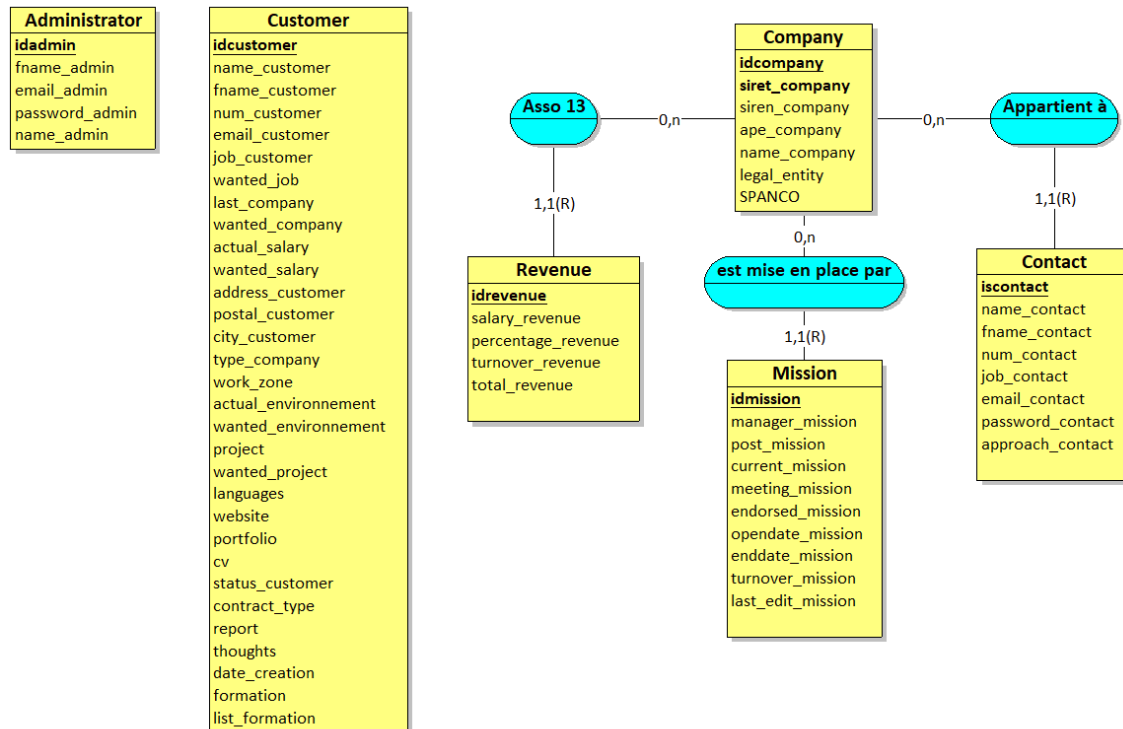
- Formations des langages informatiques utilisés :  
Udemy: <https://www.udemy.com/>  
Openclassroom:  
<https://openclassrooms.com/fr/courses/5489656-construisez-un-site-web-a-l-aide-du-framework-symfony-5>  
<https://openclassrooms.com/fr/courses/1665806-programmez-en-orientee-objet-en-php>
- Utilisation de Twilio:  
<https://www.twilio.com/docs/sms/send-messages>
- Utilisation de Mailjet:  
<https://dev.mailjet.com/email/guides/getting-started/>

## IX. ANNEXES

### A. Annexe1 : Dictionnaire de données

A	B	C	D	E	F	G
<b>Administrator</b>						
	idadmin	Identifiant de l'administrateur	int	11	élémentaire	
	name_admin	nom de l'administrateur	varchar	255	élémentaire	
	fname_admin	prénom de l'admin	varchar	255	élémentaire	
	email_admin	email de l'admin	varchar	255	élémentaire	
	password_admin	password de l'admin	varchar	255	élémentaire	
<b>Customer</b>						
	idcustomer	Id du candidat	int	11	élémentaire	
	name_customer	nom du candidat	varchar	255	élémentaire	
	fname_customer	prénom du candidat	varchar	255	élémentaire	
	num_customer	numéro de tel du candidat	varchar	255	élémentaire	
	email_customer	email du candidat	varchar	255	élémentaire	
	job_customer	poste du candidat	varchar	255	élémentaire	
	wanted_job	poste voulu du candidat	varchar	255	élémentaire	
	last_company	dernière entreprise occupée	varchar	255	élémentaire	
	wanted_company	entreprise souhaitée	varchar	255	élémentaire	
	actual_salary	salairé actuel	varchar	255	élémentaire	
	wanted_salary	salairé voulu	varchar	255	élémentaire	
	address_customer	adresse du candidat	varchar	255	élémentaire	
	postal_customer	code postal du lieu de domicile	varchar	255	élémentaire	
	city_customer	ville du lieu de domicile	varchar	255	élémentaire	
	type_company	type d'entreprise souhaité	varchar	255	élémentaire	
	work_zone	zone de travail souhaité	varchar	255	élémentaire	
	actual_environment	environnement actuel	varchar	255	élémentaire	
	wanted_environment	environnement voulu	varchar	255	élémentaire	
	project	project déjà effectués	varchar	255	élémentaire	
	wanted_project	projets souhaités	varchar	255	élémentaire	
	languages	Langues maîtrisées	varchar	255	élémentaire	
	website	site web du candidat	varchar	255	élémentaire	
	portfolio	portfolio du candidat	varchar	255	élémentaire	
	cv	CV du candidat	varchar	255	élémentaire	
	status_customer	status du candidat	varchar	255	élémentaire	
	contract_type	type de contrat souhaité	varchar	255	élémentaire	
	report	Précisions à dire sur le candidat	varchar	255	élémentaire	
	thoughts	Ce que l'on pense du candidat	varchar	255	élémentaire	
	date_creation	Date de création de la fiche du candidat	date		élémentaire	
	formation	Si le candidat veut une formation ou non	bool	1	élémentaire	
	list_formation	Liste des formations	varchar	255	élémentaire	
<b>Company</b>						
	idcompany	Id de l'entreprise	int	11	élémentaire	
	siret_company	Siret de l'entreprise	int	11	élémentaire	
	siren_company	siren de l'entreprise	int	11	élémentaire	
	ape_company	ape de l'entreprise	varchar	255	élémentaire	
	name_company	nom de l'entreprise	varchar	255	élémentaire	
	legal_entity	Raison sociale de l'entreprise	varchar	255	élémentaire	
	SPANCO	Le spanco de l'entreprise	int	11	élémentaire	
<b>Contacts</b>						
	idcontact	Id du contact	int	11	élémentaire	
	idcompany	Id de l'entreprise du contact	int	11	élémentaire	
	name_contact	nom du contact	varchar	255	élémentaire	
	fname_contact	prénom du contact	varchar	255	élémentaire	
	num_contact	numéro de tel du contact	varchar	255	élémentaire	
	job_contact	Poste du contact	varchar	255	élémentaire	
	email_contact	email du contact	varchar	255	élémentaire	
	password_contact	mot de passe du contact	varchar	255	élémentaire	
	approach_contact	Le suivi opéré du contact	varchar	255	élémentaire	
<b>Mission</b>						
	idmission	Id de la mission	int	11	élémentaire	
	idcompany	Id de l'entreprise lié à la mission	int	11	élémentaire	
	manager_mission	Manager de la mission	varchar	255	élémentaire	
	post_mission	Le poste dans la mission	varchar	255	élémentaire	
	current_mission	Détails de la mission actuelle	varchar	255	élémentaire	
	meeting_mission	Entretien avec le client	varchar	255	élémentaire	
	endorsed_mission	Signé de la mission	varchar	255	élémentaire	
	opendate_mission	Date de la création de la mission	date		élémentaire	
	enddate_mission	Date de la clôture de la mission	date		élémentaire	
	turnover_mission	Chiffre d'affaires de la mission	int	11	élémentaire	
	last_edit_mission	date de la dernière modification de la mission	date		élémentaire	
<b>Revenue</b>						
	idrevenue	Id du chiffre d'affaire	int	11	élémentaire	
	salary_revenue	Salairé annuelle	int	11	élémentaire	
	percentage_revenue	Pourcentage	int	11	élémentaire	
	turnover_revenue	Chiffre d'affaires préventioennel	int	11	élémentaire	
	total_revenue	Total préventioennel	int	11	Calculé	Salairé annuel + Chiffre d'affaires préventioennel

## B. Annexe 2 : Modèle Conceptuel de données



## C. Annexe 3 : Page de création de comptes candidats

## Creation du compte

### Informations personnelles

Prenom :

Nom :

Telephone :

Status actuel :

Adresse mail :

Vérification adresse mail :

### Dernier poste occupé

Laissez vide si vous n'avez pas occupé de poste dernièrement.

Fonction :

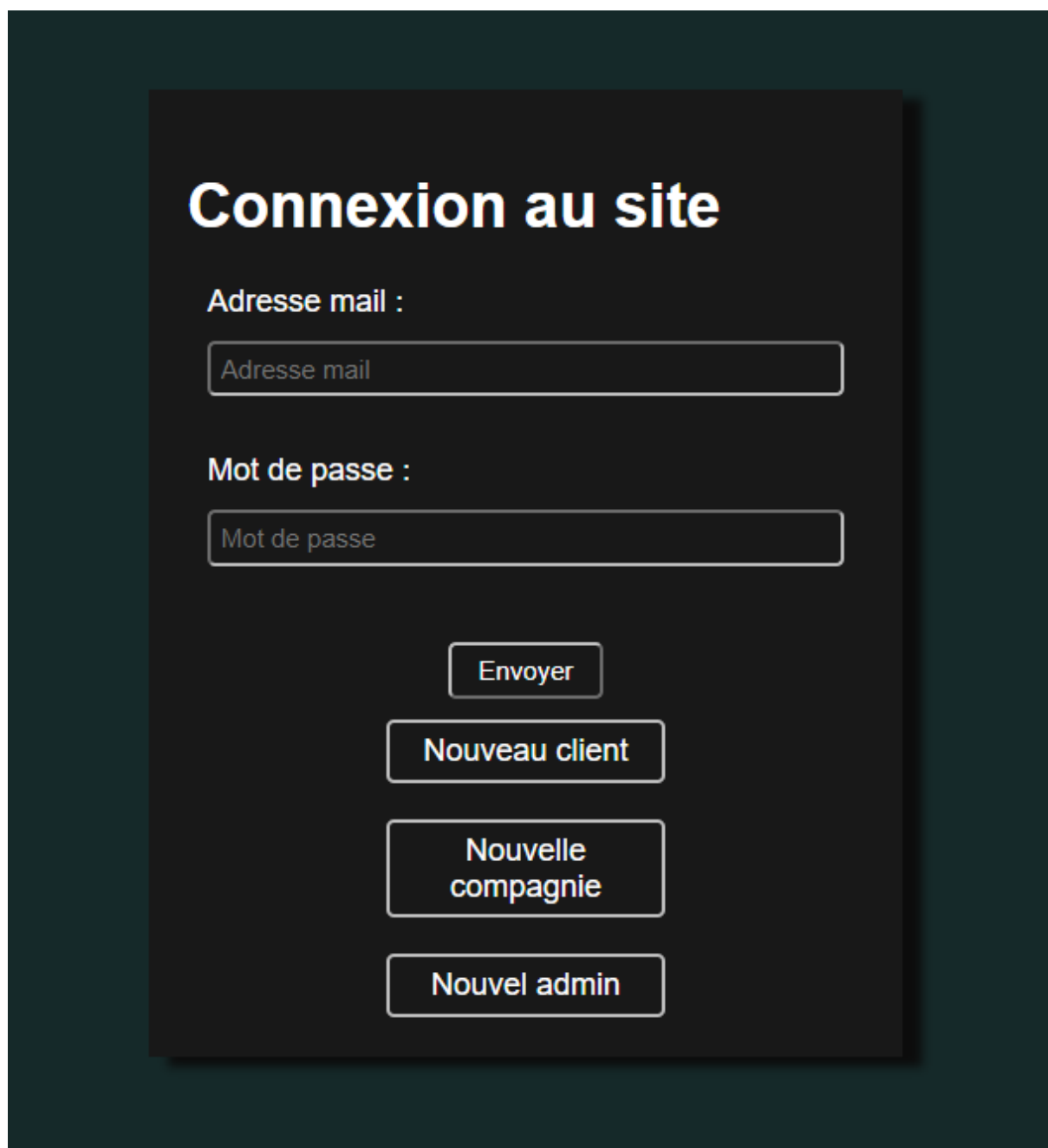
Nom de l'entreprise :

### Sécurité

Mot de passe :

Vérification mot de passe :

## D. Annexe 4 : Page de connexion à l'application



## Connexion au site

Adresse mail :

Mot de passe :

Envoyer

Nouveau client

Nouvelle  
compagnie

Nouvel admin

## E. Annexe 5 : Page de liste des candidats

Candidats							Entreprises	Log out
-	Prenom	Nom	Telephone	Adresse mail	Inscrit depuis	Avis	Accéder a la fiche	
	Marie	Gabriel	33628578965	g.marie2002@laposte.net	10/9/2020	★★★★★	<a href="#">Fiche détaillée</a>	
	Michel	Dupont	33625496781	d.michel1984@laposte.net	6/9/2021	★★★★★	<a href="#">Fiche détaillée</a>	
	Marie	Camille	33628717401	c.marie189@laposte.net	14/6/2021	★★★★★	<a href="#">Fiche détaillée</a>	
	Marie	Hubert	336287456895	h.marie1882@hotmail.fr	15/6/2021	★★★★★	<a href="#">Fiche détaillée</a>	
<div>Nouveau candidat</div>								

## F. Annexe 6 : Fiche détaillée du candidat

NOM

POSTE ACTUEL

SALAIRE

PRENOM

POSTE SOUHAITE

SALAIRE SOUHAITE

PORTABLE

DERNIERE ENTREPRISE

ORIGINE DU CONTACT

MAIL

ENTREPRISES SOUHAITEES

CP

VILLE

ENVIRONNEMENT TECHNIQUE ACTUEL

ENVIRONNEMENT TECHNIQUE SOUHAITE

LES PLUS GROS PROJETS

LES PROJETS SOUHAITES

LANGUES

SITE WEB

PORTFOLIO

CV

STATUT DU CANDIDAT

TYPE DE CONTRAT

COMPTE RENDU D'ENTRETIENS

AVIS DE L'AGENT

26/6/2021

Etat du formulaire

Avis sur le profil

enregistrer