# Compte rendu projet Générateur de CV en ligne

## Introduction

Notre objectif est de réaliser un site web permettant à un utilisateur de générer son CV en ligne. Cependant pour pouvoir enregistrer ses informations il lui faudra un compte. Pour gérer cela, nous devons mettre en place une base de données qui nous permettra de gérer la connexion à son compte mais aussi de stocker les différentes informations relatives à son CV. La dernière étape consistera à la mise sous format PDF de son cv. Pour qu’un générateur de cv soit utile, il nous semblera important de considérer son aspect esthétique mais aussi et surtout son aspect simple et pratique d’utilisation.

Table des matières

[Compte rendu projet Générateur de CV en ligne 1](#_Toc55681752)

[Introduction 1](#_Toc55681753)

[I. Architecture de la base de données 2](#_Toc55681754)

[1) Les grandes parties de la base 2](#_Toc55681755)

[2) L’organisation des sous-tables, et leur intérêt 2](#_Toc55681756)

[3) Le fonctionnement des jointures 3](#_Toc55681757)

[II. Le site web 4](#_Toc55681758)

[1) L’architecture 4](#_Toc55681759)

[2) La conversion en pdf du CV 6](#_Toc55681760)

[III. Le lien entre le site et la base de données 7](#_Toc55681761)

[1) Introduction 7](#_Toc55681762)

[2) Le cryptage des mots de passe 7](#_Toc55681763)

[3) Le système d’authentification 7](#_Toc55681764)

[4) Suppression et ajout de données 7](#_Toc55681765)

[5) Filtres, listes et saisie prédictive 8](#_Toc55681766)

[Annexe 1 : Schéma vue du concepteur 9](#_Toc55681767)

[Annexe 2 : Diagramme entité-association 9](#_Toc55681768)

## Architecture de la base de données

### Les grandes parties de la base

On a décidé d’architecturer notre base autour d’une table compte retenant les informations inhérentes au propriétaire du compte. Par soucis d’optimisation, on a décidé d’établir un ID lié à chaque compte qui est la clé primaire de cette table bien que l’adresse mail est censé être unique. Pour être sûr que deux comptes avec le même mail ne soient créés, on a réglé son attribut UNIQUE. Ainsi, la table compte contient toutes les informations nécessaires à la création d’un compte et les informations importantes liées àl’utilisateur.



Autour de la table compte se placeront de grands groupes de tables correspondant chacun aux grandes parties des élément d’un CV. Cela nous permet ainsi plus de clarté et surtout de respecter les différentes normes pour la création d’une base de données. Ces cinq tables permettent de faire l’intermédiaire entre le compte et les entités comme les écoles et les entreprises. Ce sont des tables qui vont contenir la ‘matière’ des CV, c’est-à-dire les éléments à sélectionner par l’utilisateur.

Pour résumer, dans un premier niveau nous avons la table compte qui comprend toutes les données d’un utilisateur, dans un second niveau nous avons toutes les tables de jointure qui servent à faire l’intermédiaire entre le compte et les entités du CV se trouvant au troisième niveau.

### L’organisation des sous-tables, et leur intérêt (voir ANNEXE 1)

La première sous-table est celle des écoles. Nous avons créé cette table école car elle correspond à une entité principale d’une expérience scolaire. Nous avons décidé que les attributs que devaient comporter une école était tout d’abord son nom, ainsi que son adresse pour la situer, son niveau d’études et finalement son secteur. Ces quatre attributs permettent de définir précisément une école.



La seconde table s’inscrit aussi dans le cadre d’une expérience scolaires car elle correspond au nom du diplôme réalisé dans une école en particulier. Ce nom de diplôme sera affiché à côté des informations de l’école correspondante dans le CV.



La troisième sous-table entreprise contient les informations essentielles d’une entreprise, c’est dire son nom, son secteur et son adresse. Elle est connectée à la table des expériences professionnelles. Nous avons décidé de créer cette table dans le but d’afficher sur le CV les informations nécessaires pour un recruteur.



Ensuite, nous avons les tables langues et niveau qui comporte chacune d’elles un id et un nom. Les deux tables étant reliés à la table expérience linguistiques, elles vont de pair. C’est-à-dire qu’une expérience linguistique se compose d’une langue associée à un niveau.



L’avant dernière et la dernière table sont celles des qualifications et des loisirs, elles sont identiques à celles des langues et des niveaux avec un id et un nom. Nous avons décidé de choisir ce modèle pour ces quatre tables car il nous permettait de faire de créer des liste de saisie lors de la rentrée d’information par l’utilisateur.



### Le fonctionnement des jointures

Comme nous l’avons dit précédemment, la structure de notre base de données s’organise de la manière suivante. La table compte, cinq tables connectées à celle-ci, qui représentent les grandes parties du CV et qui sont des tables de jointures, puis sept autres tables qui correspondent aux entités principales, connectées aux tables intermédiaires. Nous avons utilisé des tables de jointures pour optimiser les performances lors de l’exécution du programme. En effet le temps de récupération de valeurs indexées et nommées sera plus rapide qu’une sous-requête. La deuxième raison pour laquelle nous avons utilisé ces tables de jointures est pour avoir une table organisée, lisible et simple d’utilisation.

Nous allons maintenant expliquer comment fonctionne chaque table de jointures.

La table « expérience\_scolaire » contient un id qui joue le rôle de clé primaire. Elle compose une clé étrangère nommée « id\_étudiant » qui est reliée à la clé primaire id de la table compte. Chaque table de jointures contient cette clé étrangère reliée à l’id de la table compte. Ensuite, l’id\_ecole est une clé étrangère reliée à l’id de la table école qui est sa clé primaire. L’id\_diplôme suit la même logique en étant la clé étrangère de l’id de la table diplôme. Cette table comporte aussi deux attributs qui sont les dates d’années et de sortie d’une école. Ainsi grâce à cette structure, chaque expérience scolaire correspond à un étudiant donné, à une école et à un diplôme en particulier et aux dates d’entrées et de sortie de ceux-ci.



La table « expérience\_professionnelles » à un id pour clé primaire. Comme énoncé plus haut, elle est reliée à la table compte avec l’id de celle-ci. Ensuite, elle se compose d’une clé étrangère liée à la clé primaire de la table entreprise, et des deux attributs de date d’entrée et de sortie de l’expérience. Donc chaque expérience professionnelle est reliée à un seul compte, et comporte une entreprise avec les dates correspondantes.



La troisième table de jointure est différente des deux précédentes car sa clé primaire est une clé étrangère de celle de la table compte. Elle contient également deux clés étrangères reliées aux deux id respectifs des tables langues et niveau. De ce fait, une expérience linguistique est caractérisée par une langue et un niveau et appartient à un compte.



Les deux dernières tables sont identiques, elles ont toutes deux des clés étrangères de l’id de la table compte pour clé primaire. Elles ont ensuite une clé étrangère chacune qui est reliée à l’id de la table de l’entité correspondante, c’est-à-dire le type de qualification et le type de loisirs. De la sorte, une qualification et un loisir sont reliés chacun à un compte et à un type de qualification ou de loisirs.



## Le site web

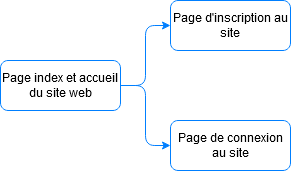
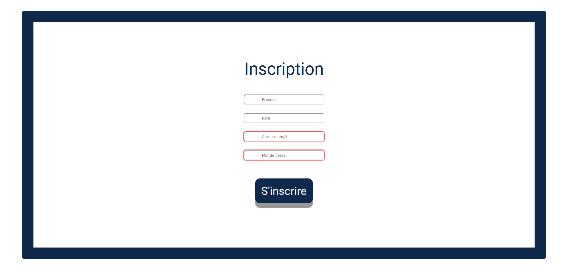
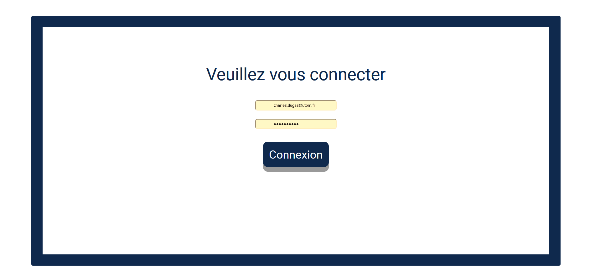
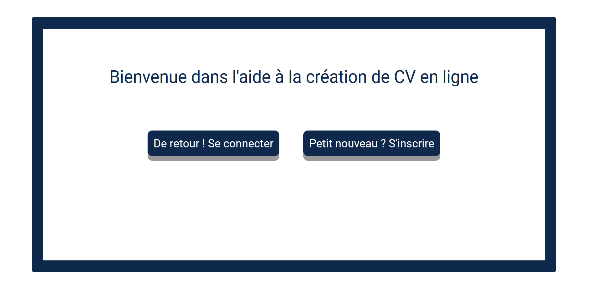
### L’architecture

#### De manière globale

Il nous a fallu réfléchir en amont avant même la création du site pour mettre au point l’architecture globale du site web car il fallait gérer le lien entre la base donnée, les éléments affichés à l’écran et la praticité d’utilisation du site par l’utilisateur. On a ainsi séparé le site web en deux grandes parties, les pages d’accueil et les pages liées au compte. Cela nous a permis d’utiliser schéma de page web pour plusieurs pages à la fois, c’est-à-dire une mise en page css commune ce qui nous a simplifier la création de ces pages et permet de mieux identifier les différentes pages au sein du site.

#### Les pages d’acceuil

L’accès au site mène à une page d’accueil très simple redirigeant vers les différentes options de connexion. On a choisi par simplification de la procédure de création du compte de se limiter aux informations essentielles de celui-ci (mail, nom, prénom, mot de passe).



Le lien entre les pages d’accueil

#### La page principale du compte

Nous avons fait le choix d’afficher de manière claire et directe toutes les informations du profil dès la première page. Pour éviter toute confusion à cause du surplus d’information, on a fait attention de bien regrouper toutes les informations par catégories. Cela se traduit en css par la création d’un grid-container global (bloc de 1er ordre) et à l’intérieur de celui-ci pleins de « mini-containers » (bloc de 2nd ordre) contenant eux-mêmes d’autre containers (bloc de 3ème ordre). Le tout est géré en flexbox pour la praticité d’utilisation mais aussi un centrage aisé des éléments mais surtout un affichage « responsive ». C’est un aspect que l’on a vraiment voulu soigner, ainsi avons aussi utilisé de manière exclusive des unités relatives pour atteindre ce but.



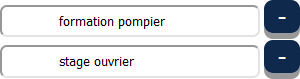
Bloc de 1er ordre

Bloc de 2nd ordre

Bloc de 3ème ordre

Les blocs de 2nd ordre qui affichent les informations du compte seront tous organisés de la même manière.

#### Le lien entre les pages du site



Informations provenant de la bdd

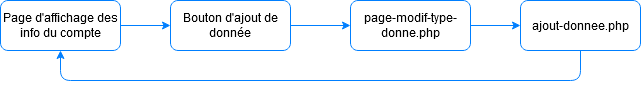
Titre de la catégorie

Bouton d’ajout de donnée

Bouton de suppression

La page de présentation des informations du compte est la page principale, celle qui va centraliser toutes les informations et toutes les fonctionnalités partent de cette page. Nous avons conçu le site web selon un schéma précis et qui a été appliqué quelque soient les catégories d’information.

* A chaque ligne d’information est associé un bouton qui va venir supprimer la ligne de donnée correspondante dans la base.
* Dans chaque catégorie d’information, un bouton d’ajout de donnée menant vers les autres pages du site est présent, l’enchainement des pages se fera toujours suivant ce schéma :



* Les pages du type ‘page-modif-type-donnee.php’ contiennent le formulaire d’ajout de donnée correspondant à la catégorie d’information à modifier.
* ‘ajout-donnee.php’ contiennent l’ensemble des requetes php pour ajouter ces données dans la base de donnée.

### La conversion en pdf du CV

#### Introduction

Convertir une page web en document pdf n’est pas aussi simple qu’il n’y parait, en effet alors qu’un pdf va utiliser des positions absolues pour repérer les informations, une page web elle va afficher les informations html à l’aide d’un css, ce qui aura pour effet de rendre la disposition des informations à l’écran relative à la taille de la fenêtre. De plus les dimensions d’une page web sont très différentes d’un document pdf qui est généralement l’A4. Nous aurions pu alors décider de créer une page web dédiée regroupant les informations nécessaires pour constituer le cv et demander la conversion avec les modules du navigateur. Cependant le rendu était très moyen, les dimensions mal respectées et une disposition aléatoire des éléments. Il nous a donc fallu trouver un autre moyen.

#### Utilisation de FPDF

On s’est donc tourné vers une bibliothèque libre et open source nommée ‘FPDF’ qui permet de générer un document pdf. Ainsi à l’intérieur du site web, existe une page nommée ‘imprimer-en-pdf.php’, rédigée entièrement en php, et qui permet de générer le CV directement en pdf.

Pour ce faire, pas de disposition relative, uniquement des positions absolues réparties sur un axe x et y. Ainsi pour chaque zone de texte que l’on veut créer, il faut lui préciser des coordonnées.

Le PDF contient deux types d’informations.

* ‘Le squelette’ : l’ensemble des informations qui sont communes à chaque CV, c’est-à-dire le titre des catégories ainsi que les éléments esthétiques de la page tels le cadre bleu.
* Les éléments générés dynamiquement, ce sont des informations récupérées à l’aide de requêtes SELECT dans la base de données.

Pour rendre la programmation plus facile et le programme plus lisible, on a mis en place une classe ‘PDF’ qui regroupe toutes les fonctions. Chaque fonction permet l’affichage d’une catégorie d’information du cv. Ainsi chaque fonction contient une requête SQL SELECT qui va venir récupérer les informations nécessaires et sont toutes affichées à l’aide d’une boucle ‘while’.

Pour être sûr que les différentes catégories d’informations soient toujours bien agencées les unes par rapport aux autres malgré les quantités de données variable, chaque fonction exécutée prend en paramètre une variable ‘hauteur’ et la transmet à la fonction suivante.

Toutes ces fonctions sont ainsi appelées les unes après les autres dans le bon ordre pour assurer la bonne disposition des informations.

## Le lien entre le site et la base de données

### Introduction

Pour assurer toutes les fonctionnalités du site web, il a fallu faire communiquer le site web avec la base de données. Pour incorporer ces données dans les pages, il a fallu utiliser le langage php qui permet d’afficher des données récupérées par le biais de requêtes SQL adressées à la base.

### Le cryptage des mots de passe

Pour sécuriser la base de données, nous avons décidé de crypter les mots de passe lorsque nous les insérions dans la base. Nous avons utilisé la fonction crypt qui prend en paramètre le mot de passe rentré par l’utilisateur lors de son inscription. Le cryptage permet de rendre le mot de passe illisible depuis la base de données. Pour ensuite comparer ce mot de passe maintenant crypté dans la base avec celui que saisi l’utilisateur lors de sa connexion, c’est plus compliqué. En effet, regarder si le mot de passe de la base est égal au mot de passe crypté saisi ne fonctionne pas. Il faut donner à la fonction crypt le mot de passe qui vient d’être saisi et celui de la base (déjà crypté). C’est seulement après cela que l’on peut comparer le mot de passe retourné par la fonction crypt avec celui venant d’être saisi et ainsi procéder à l’authentification.

### Le système d’authentification

Nous avons réfléchi au moyen d’authentifier l’utilisateur et le garder connecté malgré les allers et retours sur les pages du site. Pour donner suite au passage sur la page de connexion (‘page\_connexion.php’), avant que l’utilisateur ne soit redirigé soit dans la page d’accueil (‘compte.php’) ou bien à nouveau sur la page de connexion, les résultats du formulaire de connexion sont récupérés sur une page ‘login.php’. L’email et le mot de passe saisis sont récupérés et comparés avec les adresses email et les mots de passe présents dans la base de données. Si le mot de passe correspond à celui de la base de données et que l’adresse email est identique, alors l’utilisateur est connecté et dirigé vers la page d’accueil. Dans ce cas, une session est alors lancée avec la fonction session\_start() et nous initialisons une variable $\_SESSION avec l’email de l’utilisateur, ce qui nous permettra d’effectuer toutes les requêtes même dans des pages différentes. Sinon il est redirigé vers la page de connexion à nouveau. Les variables de session sont un peu particulières car ce sont des variables stockées dans le navigateur de manière sécurisé, ainsi une fois la connexion effectuée nous pouvons récupérer le mail de l’utilisateur avec la variable de session associée. Cependant pour éviter tout problème de confidentialité, il nous a fallu aussi créer une page qui se chargera de la suppression de la variable de session et de la fermeture de la session en cours, c’est la page ‘deconnexion.php’ qui s’en charge.

### L’affichage des données sur la page d’accueil

Dans chacun des blocs de 2nd ordre se trouve un bloc de 3ème ordre qui se charge d’afficher les informations liées à son compte de manière dynamique. Ainsi, dans chacune des catégories nous lançons des requêtes pour récupérer ces informations. Pour ce nous devons mettre au point des requêtes qui font appel à des jointures, en effet nous cherchons les informations des tables de 2ème catégories lié à un utilisateur de la table compte, cependant dans les tables de 2èmes catégories, ne sont mises que des identifiants, il faudra alors refaire une jointure avec les tables contenant la vraie valeur correspondant à ces identifiants.

De plus ne sachant à l’avance le nombre d’information à récupérer, nous faisons appel à une boucle while qui affiche les valeurs tant qu’il y en a.

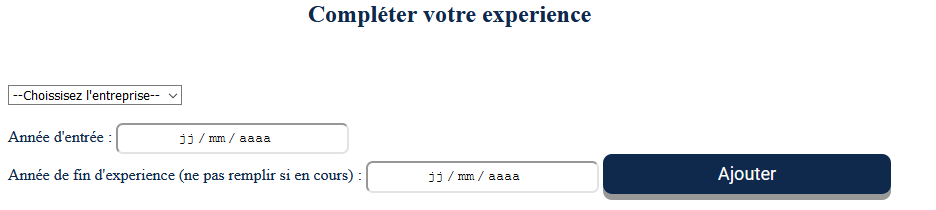
Pour afficher ces valeurs, nous avons choisi de les mettre dans un formulaire. Chaque ligne correspondra à un formulaire, comprenant les informations qui seront affichées dans des zones de texte non modifiables et un bouton servira à la suppression de la ligne. Cela nous permet lors de l’appui sur le bouton supprimer d’envoyer directement les informations qui serviront à la requête de suppression, ce qui est très pratique.

### Suppression de données

Une fois le bouton de suppression cliqué, on est redirigé vers une page de type ‘modification\_d’information.php’, cette page reconnait que l’on souhaite supprimer des valeures grâce à l’attribut ‘type=suprr’ envoyé par méthode $\_GET. On récupère alors les informations envoyées par le formulaire, et nous lançons la requête de suppression.

### Ajout de données

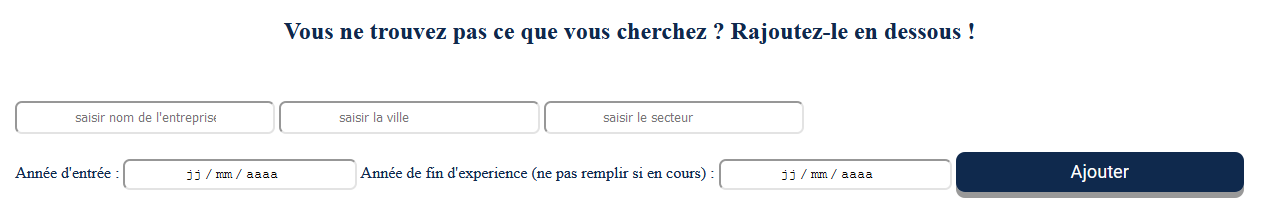
Pour ajouter des données, il faudra passer par le bouton d’ajout qui nous redirige vers des pages du type ‘modification\_d’information.php’. Cependant cette fois on la page reconnait que l’on est là pour ajouter des données et affiche la page ainsi que les formulaire d’ajout de donnée. Les données des formulaires rempli sont ensuite récupérées dans des pages (ajout\_....php) pour ensuite les ajouter dans la base de données. Les informations sont insérées dans la base en étant formatées pour qu’elles aient toutes la même typographie à l’aide de la fonction ucwords(strtolower()) qui met en minuscules puis la première lettre en majuscule. Une fois l’ajout effectué, l’utilisateur est redirigé vers la page d’accueil où il peut constater son profil avec les modifications apportées à l’instant.



### Filtres, listes et saisie prédictive

Pour faciliter la saisie des informations par l’utilisateur, nous avons mis en place dans les formulaires des listes déroulantes, un système de filtre et des champs de saisie avec saisie prédictive.

Cependant pour chacune des catégories nous n’avons pas retenu les mêmes méthodes, au détail près que toutes les pages de modification contiennent un formulaire d’ajout d’information dans la base de données, si celle-ci n’est pas assez complète.



 De plus pour faciliter la vie aux utilisateurs les formulaires d’ajout manuel sont dotés saisie prédictive qui permet à l’utilisateur de voir les informations correspondant à sa saisie au fur et à mesure de celle-ci. Elles fonctionnent grâce à des ‘datalist’ qui sont générées de la même manière que les listes déroulantes. En effet des requêtes récupèrent toutes les informations dans la base et une liste est générée. Elles comportent un argument id qui correspond avec l’argument ‘list’ du champ de texte dans le formulaire.

Dans les pages de modification de scolarité nous avons choisi de fonctionner par filtre et liste de donnée du fait du grand nombre de données à traiter et des nombreuses combinaisons possibles.

Dans le formulaire pour modifier sa scolarité qui partage son fonctionnement avec la page de modification d’expérience professionelle, afin de rechercher facilement une école, nous avons mis en place un système de filtre permettant de trier les écoles par villes, secteurs et niveaux d’études. Lorsque l’utilisateur choisi d’appliquer un filtre, le bouton submit associé à ce formulaire renvoie au haut de cette même page avec un argument type égal à ‘filtre’. Ce type permet déjà de différencier le formulaire de suppression et l’ajout de données avec les types ‘ajout’ et ‘suppr’. Nous avons donc créé une condition qui vérifie si le type est égal à ‘filtre’, si c’est le cas nous récupérons les valeurs des trois filtres du formulaire. Ensuite le code reprend son cours. Seulement lorsqu’il affiche la liste déroulante permettant de choisir les écoles, en fonction des filtres appliqués (que la ville, la ville et le secteur, …) des requêtes avec ces filtres comme conditions dans le WHERE sont faites et affichent les écoles correspondantes dans la liste. Cette liste est générée par du code php qui renvoie pour chaque résultat de la requête une option de plus dans la liste avec comme valeur le nom de l’école.

Dans les pages d’ajout de qualification et de langue nous avons privilège un système de liste simple du fait de l’impossibilité de tri comparé aux pages précédentes.

La page de modification de loisir, du fait des immenses possibilités et variantes possible, nous avons choisi de n’utiliser qu’une saisie prédictive.

# Annexe 1 : Schéma vue du concepteur

# Annexe 2 : Diagramme entité-association