



# Générateur Cv

```
{function Generateur_Cv() {  
    Database();  
    Ajout();  
    connect();  
    cv();  
})();}
```

Présenter par :  
Lamiri yasmine  
Ben rhaiem safa

# Introduction

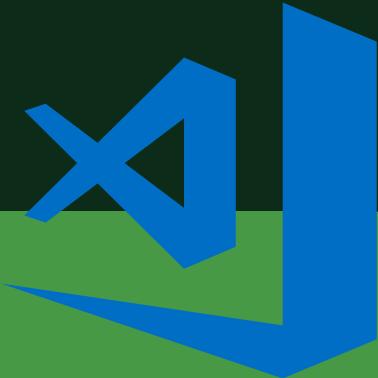


Notre projet consiste en un générateur de CV en ligne, développé en utilisant PHP comme langage principal. L'objectif de ce projet est de fournir aux utilisateurs un moyen simple et efficace de créer leur CV en remplissant un formulaire en ligne.

Cette solution permet aux utilisateurs de saisir leurs informations personnelles, leurs expériences professionnelles, leur formation académique et d'autres détails pertinents, puis de générer automatiquement un CV bien formaté en sortie.



# Technologies Utilisées



- **PHP** : Utilisé pour la logique back-end, le traitement des données et la génération de contenu dynamique.
- **HTML et CSS** : Utilisés pour concevoir et structurer les pages web, ainsi que pour le style et la mise en page.
- **MySQL** : Base de données relationnelle utilisée pour stocker les informations utilisateur et les CV générés.
- **WAMP** : Environnement de développement local qui intègre Apache, MySQL et PHP pour faciliter le développement et les tests.



# Fonctionnalités Principales



- 1. Inscription et Connexion** : Les utilisateurs peuvent créer un compte. Ils peuvent ensuite se connecter pour accéder à leur espace.
- 2. Formulaire de CV** : Un formulaire intuitif permet aux utilisateurs de saisir leurs informations.
- 3. Génération de CV** : Une fois le formulaire rempli, les données sont traitées côté serveur avec PHP pour générer un CV au format HTML.
- 3. Stockage des Données** : Toutes les informations utilisateur et les CV générés sont stockés de manière sécurisée dans une base de données MySQL.

# Flux d'exécution du Code PHP

## *Connexion à la Base de Données*

Database.php : Code pour la connexion à la base de données MySQL en utilisant PDO.

### 1. Déclaration de la classe BaseDeDonnees :

-Cette classe encapsule les fonctionnalités liées à la connexion et à l'utilisation d'une base de données MySQL.

### 2. Propriétés de la classe :

**:serveur, utilisateur, motDePasse, baseDeDonnees** : Ces propriétés définissent les paramètres de connexion à la base de données.

### 3. Constructeur (\_\_construct()) :

-Initialise une connexion à la base de données en utilisant les informations fournies.

-Configure le mode d'erreur pour générer des exceptions en cas d'erreur lors de l'exécution des requêtes SQL.

### 4. Méthodes de la classe :

**-fermerConnexion()** : Ferme la connexion à la base de données.

**-query(\$sql)** : Exécute une requête SQL et retourne le résultat.

# DATABASE.PHP

```
Database.php      connexion.html      connect.php      ajout.php      GEN-CV.html      cv.php      inscription.html

<?php
class BaseDeDonnees {
    private $serveur = 'localhost';
    private $utilisateur = 'root';
    private $motDePasse = '';
    private $baseDeDonnees = 'bd_cv';
    public $connexion;

    public function __construct() {
        try {
            $this->connexion = new PDO("mysql:host={$this->serveur};dbname={$this->baseDeDonnees}", $this->utilisateur, $this->motDePasse);
            $this->connexion->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        } catch (PDOException $e) {
            die("Erreur de connexion à la base de données : " . $e->getMessage());
        }
    }

    public function fermerConnexion() {
        $this->connexion = null;
    }

    public function query($sql) {
        return $this->connexion->query($sql);
    }
}
?>
```

# *inscription d'utilisateur*

Ajout.php : Page traitant la soumission du formulaire d'inscription, insérant les données dans la base de données

1.Inclusion du fichier contenant la classe de connexion à la base de données : Le fichier Database.php est inclus pour utiliser la classe BaseDeDonnees par require('Database.php');

2.Traitement des données du formulaire : Les données soumises par le formulaire HTML sont récupérées à l'aide de la superglobale `$_POST`.

3.Préparation et exécution de la requête d'insertion SQL : Une requête SQL d'insertion est préparée avec les données soumises et exécutée à l'aide de PDO dans le variable `$stmt`

4.Affichage des résultats : Un message de réussite ou d'échec est affiché dans une condition en fonction du résultat de l'exécution de la requête.

# AJOUT.PHP

```
1 <?php
2 require('Database.php');
3
4 $bd = new BaseDeDonnees();
5
6 if ($_SERVER["REQUEST_METHOD"] == "POST") {
7     $u = isset($_POST['nom']) ? $_POST['nom'] : '';
8     $e = isset($_POST['email']) ? $_POST['email'] : '';
9     $p = isset($_POST['mot_pass']) ? $_POST['mot_pass'] : '';
10
11    $stmt = $bd->connexion->prepare("INSERT INTO inscription(nom, email, mot_pass) VALUES (:nom, :email, :mot_pass)");
12    $stmt->bindParam(':nom', $u);
13    $stmt->bindParam(':email', $e);
14    $stmt->bindParam(':mot_pass', $p);
15
16    if ($stmt->execute()) {
17        echo "<div><h3>Vous êtes inscrit avec succès</h3><p>Cliquez ici pour vous <a href='connexion.html'>connecter</a></p></div>";
18    } else {
19        echo 'Requête non exécutée';
20    }
21
22    $stmt->closeCursor();
23} else {
24    echo "Erreur : Les données ne sont pas correctes.";
25}
26
27 $bd->fermerConnexion();
28 ?>
29
```

# **Authentification d'utilisateur**

connect.php: permettant de vérifier les informations d'identification des utilisateurs avant de leur permettre l'accès à certaines fonctionnalités ou ressources.

## 1. Inclusion du fichier contenant la classe de connexion à la base de données

## 2. Démarrage de la session :

- La fonction **session\_start()** est appelée pour démarrer ou reprendre une session.

- 3. Définition de la classe Authentification :

- Cette classe encapsule les fonctionnalités d'authentification d'utilisateur.

- 4. Méthode connexion() :

- Cette méthode vérifie les informations d'identification soumises par l'utilisateur en exécutant une requête SQL **\$requete** pour rechercher une correspondance dans la base de données.

- vérification par **rowCount()** Si une correspondance est trouvée, l'utilisateur est redirigé vers une page spécifiée. Sinon, un message d'erreur est affiché.

## 5.Traitement du formulaire soumis :

- Les données soumises par le formulaire sont récupérées à l'aide de la superglobale **`$_POST`**.
- Une instance de la classe Authentification est créée **`$auth`** et sa méthode **`connexion()`** est appelée avec les données soumises

## 6.Fermeture de la connexion à la base de données :

- Une fois l'authentification terminée, la connexion à la base de données est fermée

**`$bd->fermerConnexion();`**

# CONNECT.PHP

```
1 <?php
2 require('Database.php');
3 session_start();
4
5 class Authentification {
6     private $bd;
7
8     public function __construct(BaseDeDonnees $bd) {
9         $this->bd = $bd;
10    }
11
12     public function connexion($email, $motDePasse) {
13         $requete = "SELECT * FROM inscription WHERE email=:email AND mot_pass=:motDePasse";
14         $stmt = $this->bd->connexion->prepare($requete);
15         $stmt->bindParam(':email', $email);
16         $stmt->bindParam(':motDePasse', $motDePasse);
17         $stmt->execute();
18
19         if ($stmt->rowCount() > 0) {
20             header("Location: GEN-CV.html");
21             exit();
22         } else {
23             echo "Identifiants invalides. Veuillez réessayer.";
24         }
25     }
26
27 if ($_SERVER["REQUEST_METHOD"] == "POST") {
28     $email = $_POST['email'];
29     $motDePasse = $_POST['mot_pass'];
30
31     $bd = new BaseDeDonnees();
32     $auth = new Authentification($bd);
33     $auth->connexion($email, $motDePasse);
34
35     $bd->fermerConnexion();
36 }
37 ?>
```

# Affichage du dernier CV enregistré

cv.php: Cela permet aux utilisateurs de visualiser rapidement les informations les plus récentes soumises.

## 1. Inclusion du fichier Database.php :

## 2. Définition de la classe CV :

- Cette classe contient des méthodes pour ajouter un CV à la base de données et récupérer le dernier CV enregistré.

## 3. Constructeur de la classe CV :

- Le constructeur prend un objet de type BaseDeDonnees en argument et initialise la propriété \$baseDeDonnees de la classe Méthode ajouterCV :
  - Cette méthode est utilisée pour ajouter les données d'un CV à la base de données.
  - Les données sont extraites du tableau \$donnees passé en argument et insérées dans une requête SQL d'insertion.

## 5. Méthode getDernierCV :

- Cette méthode récupère le dernier CV enregistré dans la base de données en exécutant une requête SQL.
- Les données du dernier CV sont renvoyées sous forme de tableau associatif.

## 6. Création de l'instance de la classe BaseDeDonnees :

- Une instance de la classe BaseDeDonnees est créée en fournissant les informations de connexion à la base de données.

## 7. Crédit de l'instance de la classe CV :

- Une instance de la classe CV est créée en passant l'instance de la classe BaseDeDonnees créée précédemment en argument.

## 8. Traitement des données soumises par le formulaire :

- Si des données sont soumises via la méthode POST, la méthode ajouterCV de l'objet \$cv est appelée pour ajouter le CV à la base de données.

## 9. Récupération des données du dernier CV :

- Les données du dernier CV enregistré sont récupérées en appelant la méthode getDernierCV de l'objet \$cv.

## 10. Fermeture de la connexion à la base de données :

- Une fois que toutes les opérations sur la base de données sont terminées, la connexion à la base de données est fermée.

# CV.PHP

The screenshot shows a code editor with multiple tabs open. The tabs include: Database.php, connect.php, ajout.php, connexion.html, GEN-CV.html, cv.php (which is the active tab), and inscription.php. The cv.php tab contains the following PHP code:

```
<?php
require('Database.php');

class CV
{
    private $baseDeDonnees;

    public function __construct(BaseDeDonnees $baseDeDonnees)
    {
        $this->baseDeDonnees = $baseDeDonnees;
    }

    public function ajouterCV($donnees)
    {
        $prenom = $donnees['prenom'];
        $nom = $donnees['nom'];
        $dateNaissance = $donnees['date-naissance'];
        $langue = $donnees['langue'];
        $email = $donnees['email'];
        $telephone = $donnees['phone'];
        $titreExperience = $donnees['titre-exp'];
        $entreprise = $donnees['entreprise'];
        $descriptionExperience = $donnees['experience_description'];
        $diplome = $donnees['education_degree'];
        $ecole = $donnees['ecole'];
        $descriptionEducation = $donnees['education_description'];
        $competences = $donnees['skills'];

        $sql = "INSERT INTO cv (prenom, nom, date_naissance, email, phone, langue, titre_exp, entreprise,
            experience_description, education_degree, ecole, education_description, skills)
            VALUES ('$prenom', '$nom', '$dateNaissance', '$email', '$telephone', '$langue', '$titreExperience', '$entreprise',
            '$descriptionExperience', '$diplome', '$ecole', '$descriptionEducation', '$competences')";

        $resultat = $this->baseDeDonnees->query($sql);

        if ($resultat) {
            echo "Données ajoutées avec succès à la base de données.";
        } else {
            echo "Erreur d'insertion dans la base de données : " . $this->baseDeDonnees->errorInfo();
        }
    }
}
```

```
40 }  
41 }  
42  
43  
44     public function getDernierCV()  
45     {  
46         $resultat = $this->baseDeDonnees->query("SELECT * FROM cv ORDER BY id DESC LIMIT 1");  
47  
48         if ($resultat) {  
49             $ligne = $resultat->fetch(PDO::FETCH_ASSOC);  
50             return $ligne;  
51         } else {  
52             return null;  
53         }  
54     }  
55  
56 }  
57  
58 $baseDeDonnees = new BaseDeDonnees('localhost', 'root', '', 'bd_cv');  
59 $cv = new CV($baseDeDonnees);  
60  
61 if ($_SERVER['REQUEST_METHOD'] === 'POST') {  
62     $cv->ajouterCV($_POST);  
63 }  
64  
65 $dernierCV = $cv->getDernierCV();  
66 $baseDeDonnees->fermerConnexion();  
67 ?>  
68
```