****

**CampusGuide : Plateforme intelligente d’orientation universitaire pour les élèves post-bac au Niger**

**Felix Aguin Aoga Penouel**

**720230045**

**Licence 2 en Intelligence Artificielle**

**African Development Universalis**

**Digital Transformation and IT**

**Superviseur** : Mr. Mounkaila Amadou Mahamadou

**Mai 2025**

# ****Résumé du Projet****

**CampusGuide** est une plateforme web conçue pour transformer l’orientation universitaire des élèves post-bac au Niger, en répondant à un besoin crucial d’accès centralisé, fiable et interactif à l’information sur les établissements d’enseignement supérieur. Dans un contexte où les solutions existantes comme Karatou Post Bac ou Campus Niger sont souvent jugées peu ergonomiques, incomplètes et peu engageantes, CampusGuide propose une alternative moderne et intuitive qui place l’élève au cœur du processus d’orientation. La plateforme permet aux utilisateurs de rechercher dynamiquement des universités ou des filières, d’accéder à un espace personnel où ils peuvent gérer leur profil, consulter des fiches détaillées sur les établissements (incluant la description, l’histoire, la localisation, les frais, les médias, les débouchés, et des brochures téléchargeables), et visualiser un classement des écoles selon différents critères. Dès le chargement de la page, des suggestions intelligentes sont proposées, améliorant l’expérience utilisateur. Un espace administrateur sécurisé a été mis en place pour gérer les universités, les filières et les utilisateurs, tout en affichant des statistiques essentielles sur la plateforme. CampusGuide est développé avec le Framework Laravel (version 8.83.29) et une base de données MySQL, garantissant une architecture fullstack robuste, évolutive et performante. Bien que certaines fonctionnalités soient en cours de développement, comme le quiz intelligent de recommandation, le suivi des candidatures, les notifications personnalisées, la messagerie interne, le forum, l’interface multilingue et les témoignages d’anciens élèves, leur intégration progressive est prévue dans les versions futures. À moyen terme, CampusGuide prévoit également une version mobile, une interface entreprise pour le suivi des talents, des profils publics pour les étudiants et une page dédiée aux opportunités professionnelles. Ce projet se positionne comme un outil stratégique pour guider efficacement les futurs étudiants nigériens dans leur parcours académique, renforcer la visibilité des talents locaux et valoriser durablement les acteurs de l’enseignement supérieur au Niger.

# Table de Matière :

[Résumé du Projet i](#_Toc199697495)

[Table de Matière : ii](#_Toc199697496)

[Table des Figures : iv](#_Toc199697497)

[Table des Tableaux iv](#_Toc199697498)

[I. Présentation du projet 1](#_Toc199697499)

[I.1 Contexte 1](#_Toc199697500)

[I.2 Problème à résoudre 1](#_Toc199697501)

[I.3Utilisateurs visés 1](#_Toc199697502)

[I.4 Importance et utilité du système 1](#_Toc199697503)

[I.5 Portée du projet 2](#_Toc199697504)

[I.6 Objectifs 2](#_Toc199697505)

[I.7 Contraintes 2](#_Toc199697506)

[II. Analyse des besoins (Phase Analyse) 3](#_Toc199697507)

[II.1 Exigences fonctionnelles 3](#_Toc199697508)

[II.2 Exigences non fonctionnelles 3](#_Toc199697509)

[II.3 Témoignages utilisateurs (Attentes exprimées) 3](#_Toc199697510)

[II.4 Diagramme de cas d’utilisation 4](#_Toc199697511)

[II.5 Matrice MoSCoW 5](#_Toc199697512)

[II.6 Planning du projet (Diagramme de Gantt) 5](#_Toc199697513)

[III. Conception du système 6](#_Toc199697514)

[III.1 Maquettes UI/UX (Wireframes) 6](#_Toc199697515)

[III.2 Architecture système 6](#_Toc199697516)

[III.3 Schéma de base de données (ERD) de CampusGuide 7](#_Toc199697517)

[III.4 Stack technologique 7](#_Toc199697518)

[IV. Mise en œuvre (Phase Développement) 9](#_Toc199697519)

[IV.1 Technologies et outils utilisés 9](#_Toc199697520)

[IV.2 Étapes du developpement frontend/backend : 9](#_Toc199697521)

[IV.3 Fonctionnalités implémentées 9](#_Toc199697522)

[IV.4 Extraits de code clé 11](#_Toc199697523)

[IV.5 Problèmes rencontrés et Résolutions 16](#_Toc199697524)

[V. Tests (Phase Test) 17](#_Toc199697525)

[V.1 Types de tests 17](#_Toc199697526)

[V.2 Plan de test 17](#_Toc199697527)

[V.3 Résultats des tests + corrections 18](#_Toc199697528)

[V.4 Retours utilisateurs 18](#_Toc199697529)

[VI. Déploiement 19](#_Toc199697530)

[VI.1 Hébergement 19](#_Toc199697531)

[VI .2 Étapes d’installation / configuration 19](#_Toc199697532)

[VI.3 Défis ou limites rencontrées 19](#_Toc199697533)

[VI.4 Suggestions pour un déploiement future 19](#_Toc199697534)

[VII. Conclusion & Travaux futurs 20](#_Toc199697535)

[VII.1 Bilan de Mon projet 20](#_Toc199697536)

[VIII. Références 21](#_Toc199697537)

[IX. Annexes 22](#_Toc199697538)

[IX.1 Annexe A 22](#_Toc199697539)

[IX.2 Annexe B 23](#_Toc199697540)

# Table des Figures :

[Figure 1: Diagramme de Cas d’utilisation 4](#_Toc199697541)

[Figure 2: Diagramme de Grantt 5](#_Toc199697542)

[Figure 3: Page d’inscription 6](#_Toc199697543)

[Figure 4: ERD CampusGuide 7](#_Toc199697544)

[Figure 5: Page de Connexion 9](#_Toc199697545)

[Figure 6: Page Universitaire 10](#_Toc199697546)

[Figure 7: Espace Personnel 10](#_Toc199697547)

[Figure 8: Page Administrateur 22](#_Toc199697548)

[Figure 9: Structure Table Utilisateurs 23](#_Toc199697549)

[Figure 10: Structure Table Universites 23](#_Toc199697550)

[Figure 11: Struture Table Filieres 24](#_Toc199697551)

[Figure 12: Struture Table Pivot Propose 24](#_Toc199697552)

# Table des Tableaux

[Tableau 1: Tableau de MOSCOW 5](#_Toc199625048)

# Présentation du projet

## I.1 Contexte

CampusGuide est né d’un constat : au Niger, les élèves post-bac rencontrent des difficultés à trouver des informations fiables et centralisées sur les établissements universitaires, les filières, et les débouchés professionnels. Cela crée une mauvaise orientation scolaire pouvant conduire à un désintérêt ou à l’abandon des études.

## I.2 Problème à résoudre

Comment permettre aux étudiants nigériens de faire des choix d’orientation post-bac éclairés, en mettant à leur disposition une plateforme intuitive et intelligente regroupant toutes les données essentielles sur les filières et établissements universitaires disponibles ?

## I.3Utilisateurs visés

Les principaux utilisateurs sont :  
- Les élèves post-bac  
- Les parents souhaitant accompagner leurs enfants  
- Les conseillers d’orientation  
- Les établissements supérieurs souhaitant valoriser leur offre

## I.4 Importance et utilité du système

CampusGuide est une plateforme conçue pour centraliser toutes les informations utiles à la prise de décision des élèves après le bac. Elle met à disposition des fonctionnalités intelligentes, comme effectuer une recherche des universites, un classement dynamique des universites du pays et un espace dédier a l’utilisateurs afin de voir ces universites mis en favoris ainsi que les universites et filieres adapter à son profil, afin d'accompagner chaque élève dans son parcours d’orientation. Grâce à cette approche, la plateforme contribue à une meilleure correspondance entre le profil des étudiants et les parcours universitaires disponibles, réduisant ainsi les erreurs d’orientation et favorisant la réussite académique.

## I.5 Portée du projet

CampusGuide est une plateforme web accessible aussi bien depuis un ordinateur que sur mobile, via un simple navigateur. Elle offre une interface intuitive et fluide, intégrant un moteur de recherche performant pour les filières et les universités, ainsi qu’un espace personnalisé pour chaque établissement regroupant brochures, débouchés, et informations pratiques. La plateforme propose également un classement des universités pour aider à la comparaison, ainsi qu’un espace personnel dédié à chaque utilisateur, lui permettant de modifier et mettre à jour son profil, sauvegarder ses établissements favoris.

## I.6 Objectifs

* Offrir une orientation académique personnalisée.
* Centraliser toutes les informations nécessaires à la prise de décision post-bac dans une plateforme unique
* Mettre à disposition un espace numérique personnaliser pour les utilisateurs et aux parents.
* Valoriser les universités et écoles locales en améliorant leur visibilité auprès des futurs étudiants

## I.7 Contraintes

* Difficulté d’accès à des bases de données officielles ou à jour pour les universités locales

# Analyse des besoins (Phase Analyse)

## II.1 Exigences fonctionnelles

* Création de compte utilisateur
* Connexion sécurisée
* Recherche intelligente de filières et universités
* Fiches détaillées pour chaque établissement
* Système de classement des établissements
* Ajout aux favoris
* Espace personnel pour chaque utilisateur
* Téléchargement de brochures PDF
* Espace Administrateur pour enrôler les universités automatiquement

## II.2 Exigences non fonctionnelles

* **Performance** : chargement rapide des pages
* **Sécurité** : protection des données personnelles (authentification sécurisée, confidentialité)
* **Accessibilité** : interface responsive compatible PC et mobile
* **Fiabilité** : système stable avec faible taux d’erreurs
* **Extensibilité** : possibilité d’ajouter des fonctionnalités dans le futur

## II.3 Témoignages utilisateurs (Attentes exprimées)

* Accès centralisé à l’information sur les universités et les filières, sans avoir à consulter plusieurs sites ou se déplacer physiquement.
* Interface simple et claire permettant de trouver rapidement les établissements correspondant à ses critères.
* Outil de recommandation ou d’aide à la décision pour les élèves qui ne savent pas encore quoi choisir après le bac.
* Possibilité d’échanger avec d’autres étudiants ou d’anciens élèves pour mieux comprendre les débouchés et la vie universitaire.

## II.4 Diagramme de cas d’utilisation

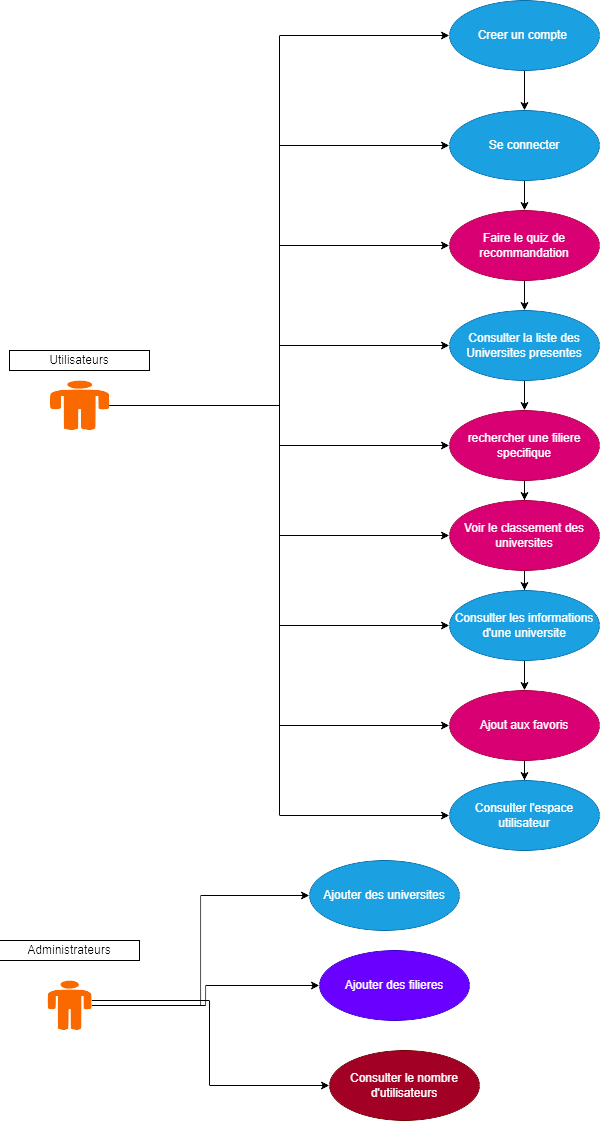
Ci-joint mon diagramme d’utilisation :

Figure 1: Diagramme de Cas d’utilisation

## II.5 Matrice MoSCoW

Tableau 1: Tableau de MOSCOW

| **Priorité** | **Fonctionnalité** |
| --- | --- |
| **Must** | Recherche universités, affichage des universités, fiches détaillées, inscription/connexion, recherche des filières, compte utilisateur, description d’université, une interface attrayante, |
| **Should** | Classement, favoris, espace dédier à l’utilisateur, |
| **Could** | Notifications, témoignages d’anciens élèves, système de suivi, historique |
| **Won’t** | Application mobile native (pour l’instant) |

## II.6 Planning du projet (Diagramme de Gantt)

Une image contenant texte, capture d’écran, Police, nombre

Le contenu généré par l’IA peut être incorrect.Ci-joint mon diagramme de Grantt

Figure 2: Diagramme de Grantt

# Conception du système

## III.1 Maquettes UI/UX (Wireframes)

Pour concevoir l’expérience utilisateur de la plateforme **CampusGuide**, des maquettes ont été réalisées à l’aide de **Figma**.

* **Page d’accueil**.
* **Page de recherche d’université ou de filière**
* **Profil utilisateur**
* **Espace dédié à chaque université**
* **Espace Utilisateur**
* **Page de classement universitaires**

Une image contenant texte, capture d’écran, Poubelle, conception

Le contenu généré par l’IA peut être incorrect.

Figure 3: Page d ’inscription

## III.2 Architecture système

Le système suit une architecture classique en 3 couches :

* **Client** (navigateur web) : HTML, CSS, JavaScript
* **Serveur** (backend PHP avec Laravel) : Gestion de la logique métier, authentification.
* **Base de données** (MySQL) : Stockage des utilisateurs, universités, filières...etc.
* **Schéma simplifier de l’architecture :** Client (Browser) ⇆ Backend (Laravel) ⇆ Base de données (MySQL)

## III.3 Schéma de base de données (ERD) de CampusGuide

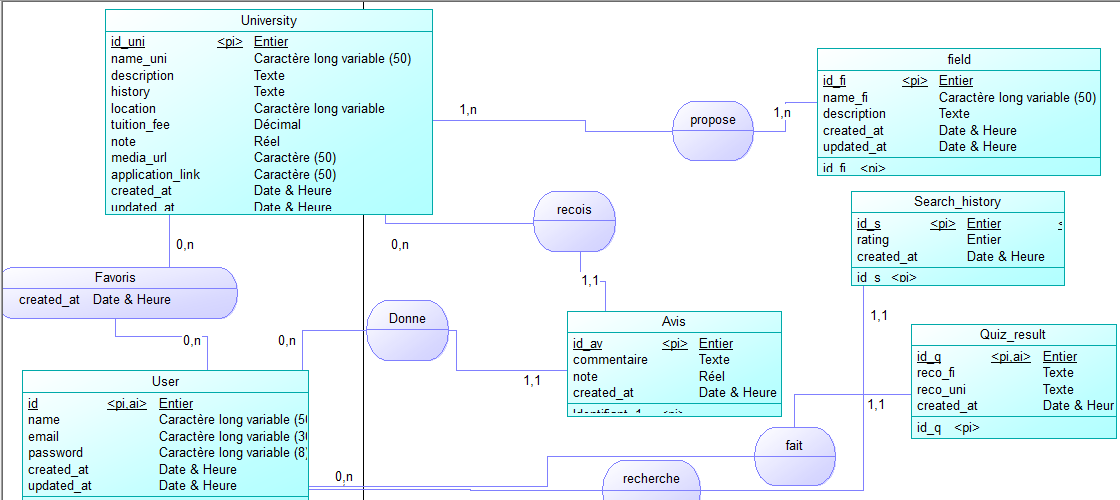


Figure 4: ERD CampusGuide

## III.4 Stack technologique

* **Frontend** : HTML5, CSS, JavaScript
* **Backend** : PHP (avec Laravel )
* **Base de données** : MySQL
* **Outils** : Figma(maquettes), Draw.io (diagrammes), GitHub (gestion de version), Power AMC (ERD), Canvas (Diagramme de Grant).
* **Serveur** : Hébergement Web PHP/MySQL
* **Sécurité** : Hashage des mots de passe, validations serveur/client

# Mise en œuvre (Phase Développement)

## IV.1 Technologies et outils utilisés

Le développement de CampusGuide s’est appuyé sur Laravel 8 avec Blade pour le backend, PHP 8.3 avec Composer pour la gestion des dépendances, et MySQL pour la base de données. Visual Studio Code a servi d’environnement de développement, GitHub pour le versionnage, et Opera GX pour les tests frontend.

## IV.2 Étapes du developpement frontend/backend :

**Frontend** :

Les interfaces ont été d’abord conçues puis intégrées via Blade dans Laravel. Les fichiers statiques ont été organisés proprement et l’ensemble a été adapté pour un affichage responsif sur différents écrans.

**Backend** :

Le projet Laravel a été initialisé avec la modélisation de la base de données. L’authentification a été mise en place, suivie du développement des routes, contrôleurs et modèles. La connexion entre frontend et backend a été assurée, avec des phases régulières de tests et débogage.

## IV.3 Fonctionnalités implémentées

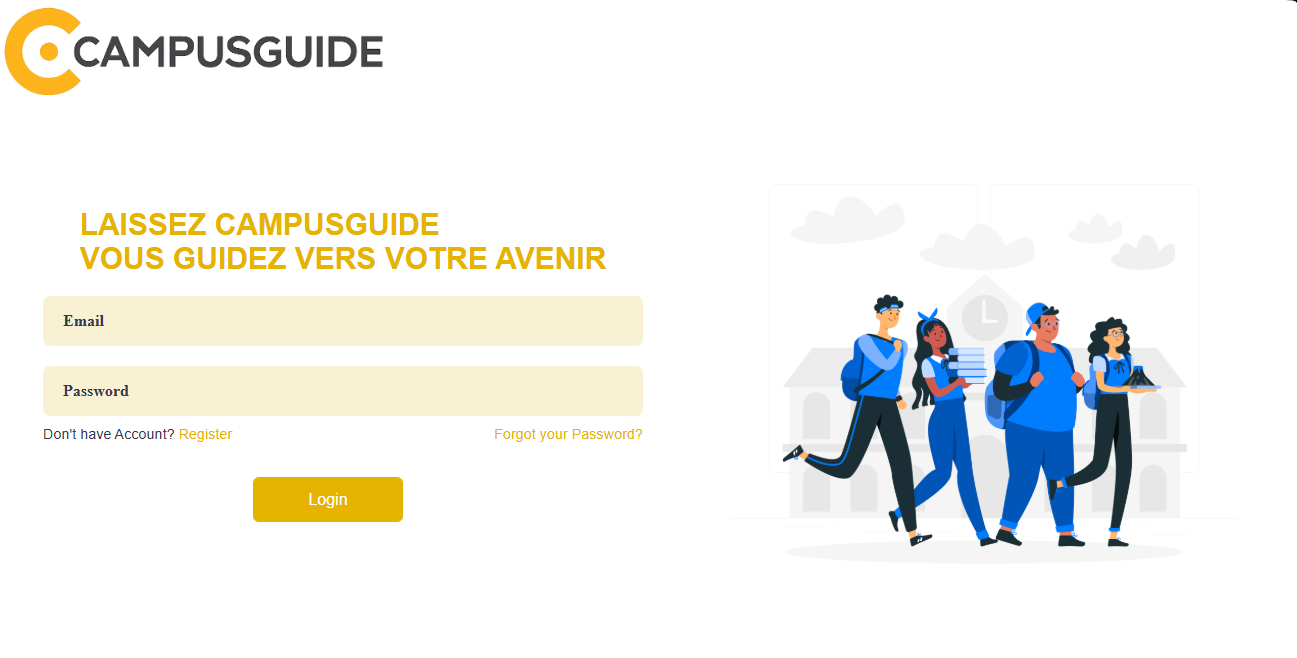
**Connexion / Inscription**

Figure 5: Page de Connexion

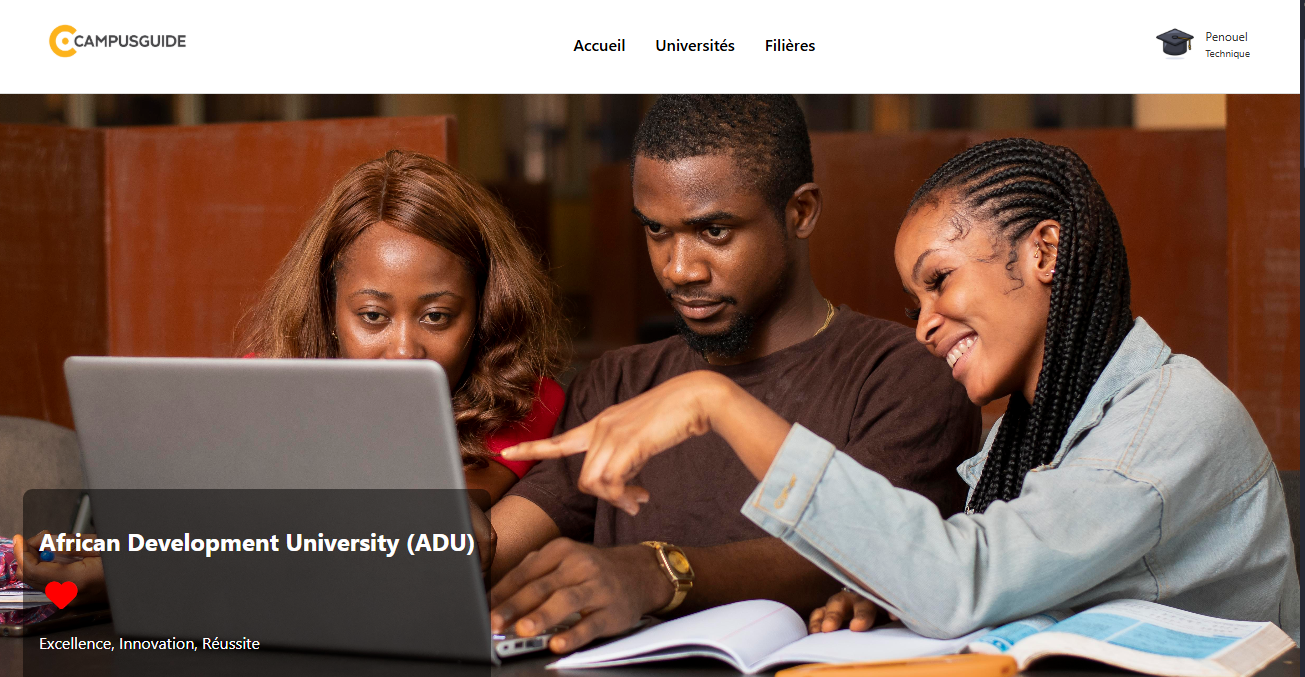
**4. Page université**.

Figure 6: Page Universitaire

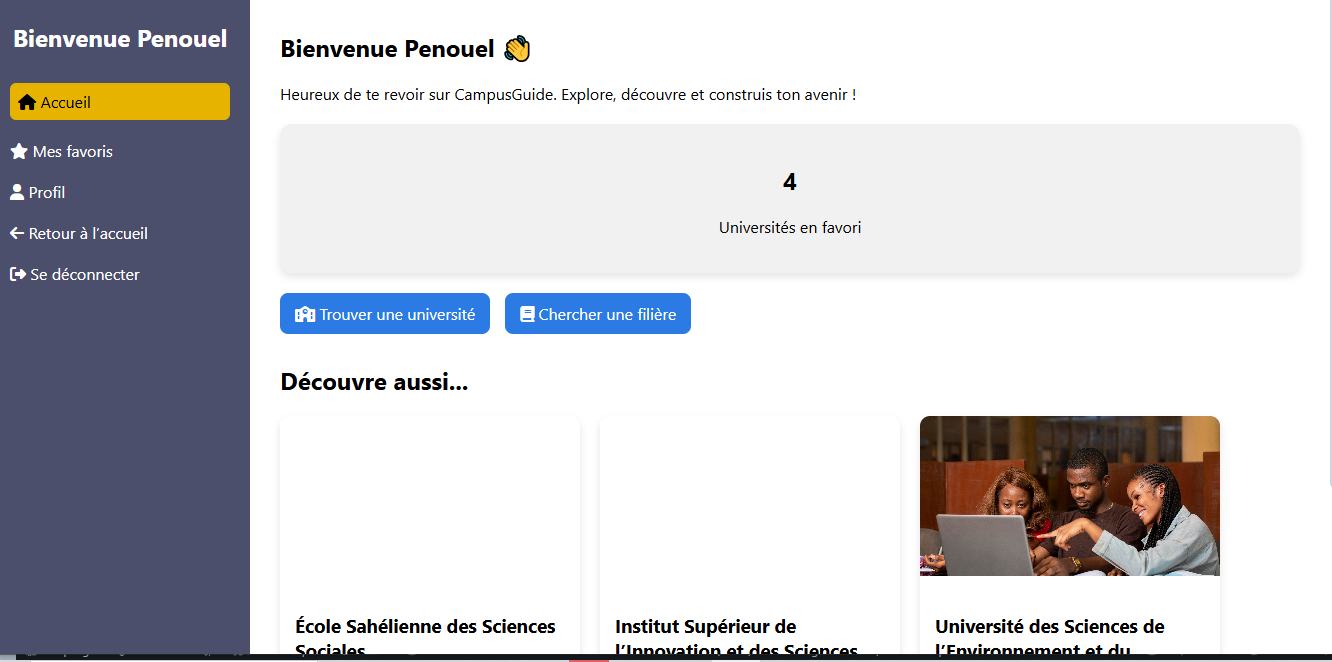
**2. Espace personnel**

Figure 7: Espace Personnel

## IV.4 Extraits de code clé

**Les routes avec un middleware afin de s’assurer que l’utilisateur s’est inscrit afin de lui donner acces a certaines fonctionnalités** :  
Route::middleware(['auth'])->group(function () {

    Route::get('/home', [UniversityController::class, 'index']);  // Cette route est protégée par auth

    Route::get('/University', function () {

        return view('auth.University');

    });

    Route::get('/field', function () {

        return view('auth.field');

    });

    Route::get('/dashboard', function () {

        return view('auth.dashboard');

    Route::middleware(['auth', 'isAdmin'])->prefix('admin')->name('admin.')->group(function () {

    Route::get('/dashboard', [AdminController::class, 'dashboard'])->name('dashboard');

    Route::post('/university/store', [AdminController::class, 'storeUniversity'])->name('university.store');

    Route::post('/field/store', [AdminController::class, 'storeField'])->name('field.store');

});

Route::middleware(['auth'])->prefix('admin')->group(function () {

    Route::put('/user/{id}/role', [AdminController::class, 'toggleRole'])->name('admin.user.toggleRole');

    Route::delete('/user/{id}', [AdminController::class, 'deleteUser'])->name('admin.user.delete');

});

Route::post('/favorites/toggle/{university}', [FavoriteController::class, 'toggle'])->name('favorites.toggle')->middleware('auth');

Route::middleware(['auth'])->group(function () {

    Route::get('/search-universities', [UniversitySearchController::class, 'index'])->name('universities.search');

   Route::get('/ajax-search-universities', [UniversitySearchController::class, 'ajaxSearch'])->name('universities.ajaxSearch');

});

Route::middleware(['auth'])->group(function () {

    Route::get('/search-fields', [FieldSearchController::class, 'index'])->name('fields.search');

    Route::get('/ajax-search-fields', [FieldSearchController::class, 'ajaxSearch'])->name('fields.ajaxSearch');

    Route::get('/filiere/{id}', [FieldSearchController::class, 'showUniversities'])->name('field.universities');

});

Route::middleware(['auth'])->group(function () {

    Route::get('/student/dashboard', [StudentController::class, 'dashboard'])->name('student.dashboard');

    Route::delete('/student/favorite/{id}', [StudentController::class, 'deleteFavorite'])->name('student.deleteFavorite');

    Route::put('/student/profile', [StudentController::class, 'updateProfile'])->name('student.updateProfile');

});

**Ceci est un modèle qui reflète les colonnes présentes dans ma table afin de récupérer et de stocker les donnees:**

class User extends Authenticatable

{

    use HasApiTokens, HasFactory, Notifiable;

    /\*\*

     \* The attributes that are mass assignable.

     \*

     \* @var array<int, string>

     \*/

    protected $fillable = [

        'name',

        'email',

        'password',

        'is\_admin',

        'bac\_type',

        'favorite\_subject',

        'interest\_area',

    ];

    /\*\*

     \* The attributes that should be hidden for serialization.

     \*

     \* @var array<int, string>

     \*/

    protected $hidden = [

        'password',

        'remember\_token',

    ];

Le code ci dessous est mon contrôleur par exemple FavoriteController qui verifie et fait les conditions nécessaires afin qu’un utilisateur puisse ajouter puisse ajouter en favoris une université:  
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

use App\Models\Favorite;

use Illuminate\Support\Facades\Log;

class FavoriteController extends Controller

{

    public function toggle($universityId)

    {

        try {

            $user = auth()->user();

            $exists = $user->favorites()->where('university\_id', $universityId)->exists();

            if ($exists) {

                $user->favorites()->where('university\_id', $universityId)->delete();

                $status = 'removed';

            } else {

                $user->favorites()->create(['university\_id' => $universityId]);

                $status = 'added';

            }

            return response()->json([

                'status' => $status,

                'university\_id' => $universityId

            ]);

        } catch (\Exception $e) {

            Log::error('Erreur dans FavoriteController@toggle : ' . $e->getMessage());

            return response()->json([

                'status' => 'error',

                'message' => 'Une erreur est survenue. Veuillez réessayer.'

            ], 500);

        }

    }

}

## IV.5 Problèmes rencontrés et Résolutions

* Première utilisation de Laravel donc c’était un peu compliquer mais avec la lecture de la documentation, les vidéos et l’aide extérieurs j’ai pu résoudre de probleme
* Ajout d’une universites en Favoris, pour résoudre cela j’ai dû utiliser AJAX que je ne maitrisais pas vraiment mais j’a
* Il fallait permettre à chaque université d’avoir plusieurs filières, et inversement. J’ai utilisé une table pivot (propose) dans Laravel avec des clés étrangères university\_id et field\_id.
* L’utilisateur devait saisir une université ou une filière et valider manuellement. J’ai intégré une barre de recherche avec suggestions instantanées (auto-complétions via AJAX), et un effet de surbrillance lors de la sélection pour améliorer l’expérience utilisateur.
* Lorsqu’un utilisateur essayait d’ajouter ou de retirer une université de ses favoris, l’action ne s’exécutait pas correctement. Parfois, rien ne se passait, ou la page se rechargeait sans effet visuel. J’ai créé une route spécifique avec une méthode AJAX pour envoyer la requête au backend sans recharger la page. J’ai ensuite ajouté un effet d’animation (pulsation) sur l’icône pour donner un retour visuel immédiat

# V. Tests (Phase Test)

## V.1 Types de tests

**Tests unitaires** :

* Test de chaque fonction individuellement (par exemple, les fonctions de validation des formulaires de connexion et d'inscription).
* Tests des modèles de la base de données (par exemple, l'ajout de favoris, l'enregistrement de candidatures).

**Tests d’intégration** :

* Test de l’interaction entre le frontend et le backend
* Tests de l’interaction avec la base de données pour les actions comme l’enregistrement d'un utilisateur.

**Tests utilisateurs** :

* Simulation d'une expérience utilisateur complète pour vérifier l'ergonomie et la fluidité (inscription, utilisation du quiz, ajout aux favoris, etc.).
* Tests avec différents types d’utilisateurs (élèves, administrateurs).

## V.2 Plan de test

* **Page d’accueil** : Vérifier que les utilisateurs peuvent accéder à la page, effectuer une recherche d’université et démarrer le quiz.
* **Formulaires de connexion/inscription** : Tester la validation des champs, l'enregistrement des utilisateurs et la gestion des erreurs (email invalide, mot de passe trop court).
* **Gestion des favoris** : Vérifier que l'ajout d'universités aux favoris fonctionne et que la base de données est bien mise à jour.
* **Page profil** : Tester l'affichage des informations utilisateur.
* **Page Universitaire** : Tester que les informations sur l’université s’affichent sur la page d’affichage.

## V.3 Résultats des tests + corrections

* **Formulaire d'inscription**Succès – Aucun problème détecté.
* **Ajout aux favoris**Échec initial – Mise à jour nécessitait un rechargement.  
  Correction : Ajout d’AJAX pour une mise à jour dynamique avec animation.
* **Recherche d’universités/filières (page d’accueil)**Succès – Fonctionne comme prévu.
* **Affichage des filières (page université)**Échec partiel – Mauvais affichage des frais.  
  Correction : Revue du Template et du formulaire admin.
* **Note/avis sur une université**Succès – Aucun souci.

## V.4 Retours utilisateurs

**Points positifs :**

* Interface claire, jolie et intuitive.
* Fonction "universités favorites" très utile.
* Informations essentielles bien regroupées.

**Suggestions :**

* Ajouter une messagerie élève-école.
* Filtrer les résultats par ville ou diplôme.
* Afficher les événements (concours, journées portes ouvertes).
* Intégrer des témoignages d’étudiants par université.

# VI. Déploiement

## VI.1 Hébergement

Actuellement, l’application **CampusGuide** est en cours de développement et de test en **environnement local**, à l’aide d’un serveur local (XAMPP). Aucun hébergement en ligne n’a encore été effectué.

## VI .2 Étapes d’installation / configuration

Pour exécuter le projet en local, les étapes suivantes sont nécessaires :

1. Cloner le projet depuis le dépôt Git (s’il est versionné).
2. Installer les dépendances via Composer : composer Install.
3. Configurer le fichier. env avec les informations de la base de données.
4. Télécharger le fichier contenant le script SQL pour la création de Table.
5. Lancer le serveur Laravel avec PHP artisan serve.
6. Accéder à l’application via <http://localhost:8000> ou en réseau a travers un adresse IP.

## VI.3 Défis ou limites rencontrées

Dépendance à un environnement local (difficile à partager avec d'autres sans mise en ligne). Limitations liées aux performances et à la sécurité en local. Pas encore de configuration pour le déploiement (SSL, domaine, optimisation). Difficulté de tester certaines fonctionnalités comme les courriels ou les notifications push sans serveur externe.

## VI.4 Suggestions pour un déploiement future

Héberger l’application sur un service comme **Render, Laravel Forge, Heroku, ou un VPS**. Configurer un domaine personnalisé et un certificat SSL. Automatiser le déploiement avec GitHub Actions ou d'autres outils CI/CD. Utiliser une solution cloud de stockage (comme AWS S3) pour les médias.

# VII. Conclusion & Travaux futurs

## VII.1 Bilan de Mon projet

Le projet *CampusGuide* a été réalisé avec succès en combinant PHP (Laravel) pour le backend et HTML/CSS/JS pour le frontend. Les principales fonctionnalités attendues ont été développées : authentification, recherche dynamique d’universités et de filières, espace personnel, et interface d’administration. Bien que certaines fonctionnalités avancées soient encore à intégrer, la plateforme répond efficacement aux besoins des élèves post-bac nigériens.

**VII.2 Objectif atteint ?**

Oui. L’objectif de proposer une solution centralisée, moderne et intuitive pour l’orientation universitaire au Niger a été atteint. *CampusGuide* permet désormais aux utilisateurs d’accéder facilement aux informations clés sur les établissements d’enseignement supérieur.

**VII.3 Compétences acquises**

Le développement de CampusGuide m’a permis de maîtriser Laravel pour créer un backend structuré, de concevoir une base de données relationnelle claire, et de développer une interface dynamique avec Blade. J’ai appris à gérer les erreurs, valider les données et intégrer AJAX pour une navigation fluide sans rechargement.

**VII.4 Évolutions prévues**

CampusGuide prévoit une version mobile, l’ajout de plus d’universités et filières, et un système de messagerie interne. Un espace actualités et opportunités, une interface entreprise, un profil public étudiant, ainsi qu’un système de recommandation intelligent et des options avancées dans l’espace personnel sont également envisagés pour enrichir l’expérience utilisateur.

# Références

(2015, May 15). *Implementation of Functionality and Debug*. OpenAI. (2023). ChatGPT. Retrieved May 1, 2025, from <https://openai.com/chatgpt>

(n.d.). *HTML TUTORIAL*. W3Schools. Retrieved April 29, 2025, from <https://www.w3schools.com/html/default.asp>

(2023, February 14). *Read Laravel Documentation*. Laravel Official Documentation. Retrieved May 5, 2025, from <https://laravel.com/docs>

(n.d.). *CampusNiger*. Plateforme d’Orientation Universitaire au Niger. Retrieved April 29, 2025, from https://www.campusniger.com

(n.d.). *KaratouPostBac*. Plateforme d’Aide à l’Orientation au Niger. Retrieved April 29, 2025, from <https://karatoupostbac.com>

(n.d.). *Parcoursup*. Portail Officiel d’Orientation post-Bac en France. Retrieved April 30, 2025, from <https://www.parcoursup.fr>

(n.d.). *CommonApp*. Plateforme d’Orientation post-Bac. Retrieved April 30, 2025, from <https://www.commonapp.org>

(2022). *GUIDE D’ORIENTATION DES BACHELIERS DU NIGER PRODUCTION : 2022* (pp. 3-52). 2022. <https://www.docdroid.net/KptzN2a/guide-dorientation-bac-abe-2022-pdf>

# Annexes

## IX.1 Annexe A

**Interface Administrateur de CampusGuide**

L’interface administrateur permet de gérer la plateforme CampusGuide de manière centralisée. Elle offre un espace réservé à l’administrateur pour :

Cette interface est accessible qu’au utilisateur ayant un droit admin depuis la base de données.

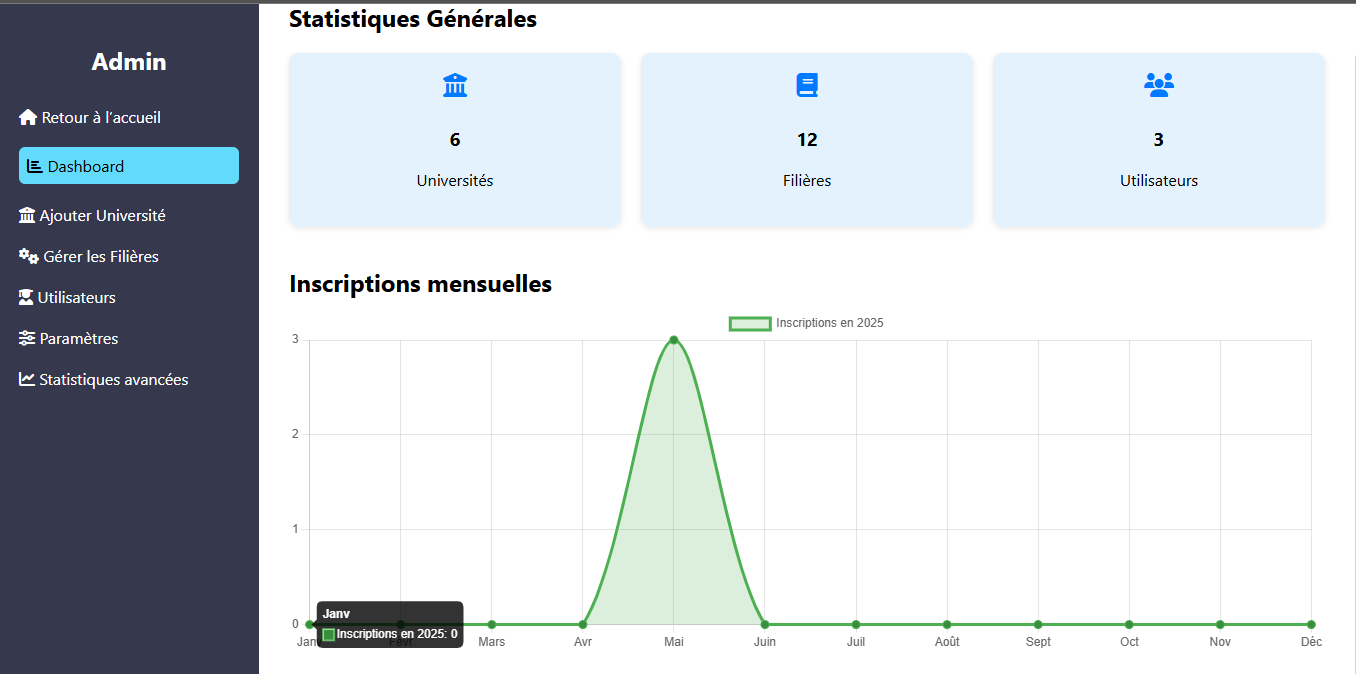


Figure 8: Page Administrateur

## IX.2 Annexe B

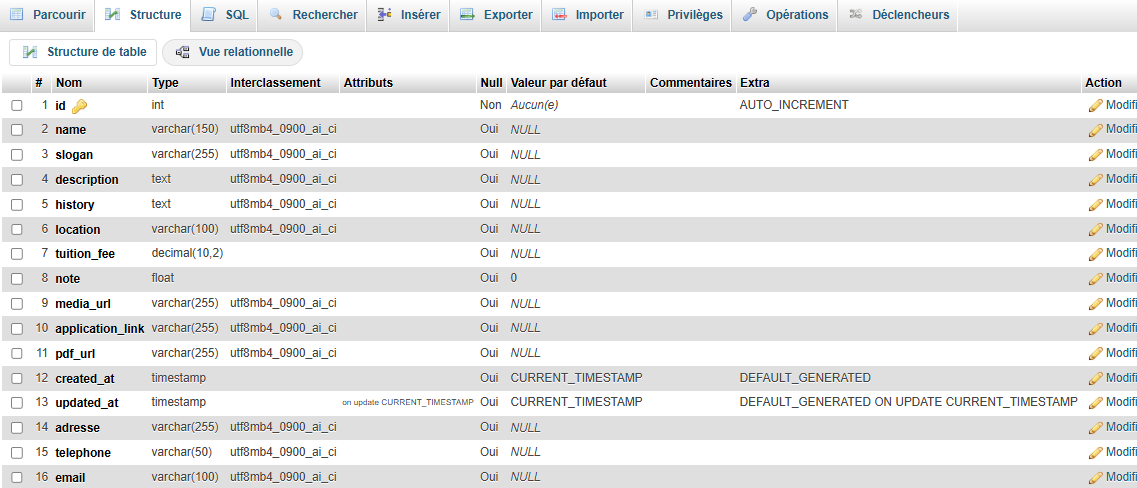
Structure de certains Tables la base de Données.

* Une image contenant texte, nombre, Police, logiciel

  Le contenu généré par l’IA peut être incorrect.**Table user**

Figure 9: Structure Table Utilisateurs

* **Table universités**



* **Table fields**

Cette table référence les différentes filières disponibles dans les universités.

Figure 10: Structure Table Universites

* **Table Filières**

Une image contenant texte, capture d’écran, Police, nombre

Le contenu généré par l’IA peut être incorrect.

Figure 11: Structure Table Filieres

* **Table Propose**

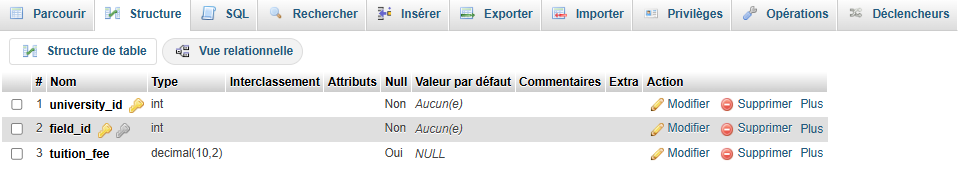
****

Figure 12: Structure Table Pivot Propose