****

**AFRICAN DEVELOPMENT UNIVERALIS (A.D.U.)**

**RAPPORT FINAL DE PROJET  
Année académique : 2024 – 2025**

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

**Réalisé par :**Nom : Aoga Penouel  
Matricule : 720230045  
Niveau : Licence 2 en Intelligence Artificielle  
Superviseur : Mr. Amadou Mahamadou Moukaila

Lieu : Niamey, Niger  
Date de soumission :

**Table de Matière :**

[Présentation du projet 4](#_Toc196843491)

[Contexte 4](#_Toc196843492)

[Problème à résoudre 4](#_Toc196843493)

[Utilisateurs visés 4](#_Toc196843494)

[Importance et utilité du système 4](#_Toc196843495)

[Portée du projet 4](#_Toc196843496)

[Objectifs 5](#_Toc196843497)

[Contraintes 5](#_Toc196843498)

[Analyse des besoins (Phase Analyse) 5](#_Toc196843499)

[Exigences fonctionnelles 5](#_Toc196843500)

[Exigences non fonctionnelles 5](#_Toc196843501)

[Témoignages utilisateurs (Attentes exprimées) 6](#_Toc196843502)

[Diagramme de cas d’utilisation 6](#_Toc196843503)

[Matrice MoSCoW 6](#_Toc196843504)

[Planning du projet (Diagramme de Gantt) 7](#_Toc196843505)

[Conception du système 7](#_Toc196843506)

[Maquettes UI/UX (Wireframes) 7](#_Toc196843507)

[Architecture système 8](#_Toc196843508)

[Schéma de base de données (ERD) 8](#_Toc196843509)

[Stack technologique 8](#_Toc196843510)

[Mise en œuvre (Phase Développement) 9](#_Toc196843511)

[Technologies et outils utilisés 9](#_Toc196843512)

[Étapes du development frontend/backend 9](#_Toc196843513)

[Fonctionnalités implémentées 10](#_Toc196843514)

[Extraits de code clé 11](#_Toc196843515)

[Problèmes rencontrés et comment tu les as résolus 11](#_Toc196843516)

[Tests (Phase Test) 11](#_Toc196843517)

[Types de tests 11](#_Toc196843518)

[Plan de test 11](#_Toc196843519)

[Résultats des tests + corrections 12](#_Toc196843520)

[Retours utilisateurs 12](#_Toc196843521)

[Déploiement 12](#_Toc196843522)

[Où est hébergée l’application 12](#_Toc196843523)

[Étapes d’installation / configuration 12](#_Toc196843524)

[Défis ou limites rencontrées 12](#_Toc196843525)

[Suggestions pour un déploiement future 12](#_Toc196843526)

[Conclusion & Travaux futurs 12](#_Toc196843527)

[Bilan de ton projet 12](#_Toc196843528)

[Objectif atteint ou non ? 13](#_Toc196843529)

[Ce que j’ai appris 13](#_Toc196843530)

[Idées d’améliorations ou d’évolutions 13](#_Toc196843531)

[Références 13](#_Toc196843532)

[Annexes 14](#_Toc196843533)

# Présentation du projet

## 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.

## 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 ?

## Utilisateurs 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

## 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.

## 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 suivre ses recherches, sauvegarder ses établissements favoris.

## 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

## Contraintes

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

# Analyse des besoins (Phase Analyse)

## 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

## 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

## Témoignages utilisateurs (Attentes exprimées)

## Diagramme de cas d’utilisation

Ci-joint mon diagramme d’utilisation :

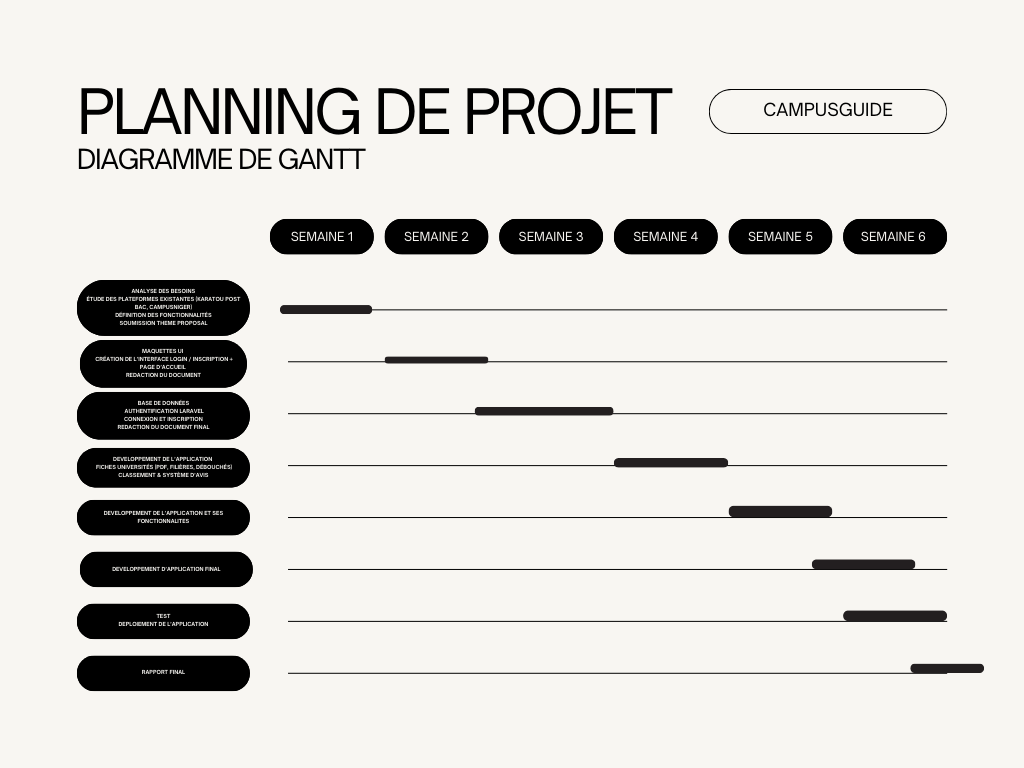
## 

## Matrice MoSCoW

| **Priorité** | **Fonctionnalité** |
| --- | --- |
| **Must** | Recherche universités, affichage des universites, 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) |

## Planning du projet (Diagramme de Gantt)

ci-joint mon diagramme de grant



# Conception du système

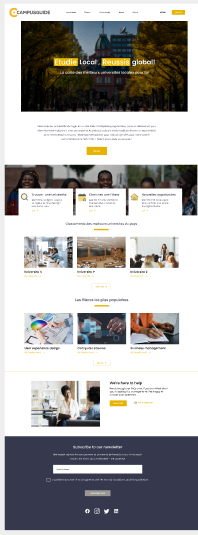
## Maquettes UI/UX (Wireframes)

Pour concevoir l’expérience utilisateur de la plateforme **CampusGuide**, des maquettes ont été réalisées à l’aide de **Figma**.  
Ces maquettes illustrent les pages principales du système, notamment :

* **Page d’accueil** avec les suggestions d’universités, des filières et le classement des universités.
* **Page de recherche d’université ou de filière**
* **Profil utilisateur**
* **Espace dédié à chaque université** (présentation, filières, brochures, etc.)
* **Espace Utilisateur**
* **Page de classement universitaires**

Les wireframes ont servi de base pour organiser l’interface de manière intuitive et responsive, avec une navigation claire et une hiérarchisation logique de l’information.

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

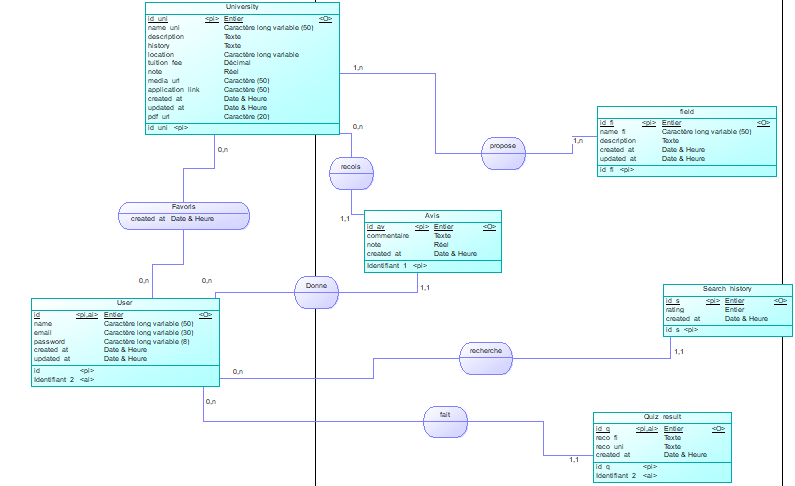
Le contenu généré par l’IA peut être incorrect.

## 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, API REST
* **Base de données** (MySQL) : Stockage des utilisateurs, universités, filières...etc.
* **Schéma simplifier de l’architecture :** Client (Browser) ⇆ Backend (Laravel ) ⇆ Base de données (MySQL)

## Schéma de base de données (ERD)



## Stack technologique

* **Frontend** : HTML5, CSS3, JavaScript (avec éventuellement du Tailwind CSS ou Bootstrap)
* **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)

## Technologies et outils utilisés

Pour le développement de l'application *CampusGuide*, les outils suivants ont été utilisés :

* **Visual Studio Code** : Environnement de développement principal.
* **Laravel 11** : Framework Laravel pour le backend et la gestion des vues avec Blade.
* **PHP 8.3** et **Composer** : Pour gérer les dépendances Laravel.
* **Git & GitHub** : Pour le versionnage du code et la collaboration éventuelle.
* **MySQL** : Système de gestion de base de données.
* **Opera GX** : Pour le test et le débogage frontend.

## Étapes du development frontend/backend

**Frontend** :

* Conception initiale des interfaces.
* Intégration dans Laravel avec Blade
* Organisation des fichiers statiques
* Responsive Design

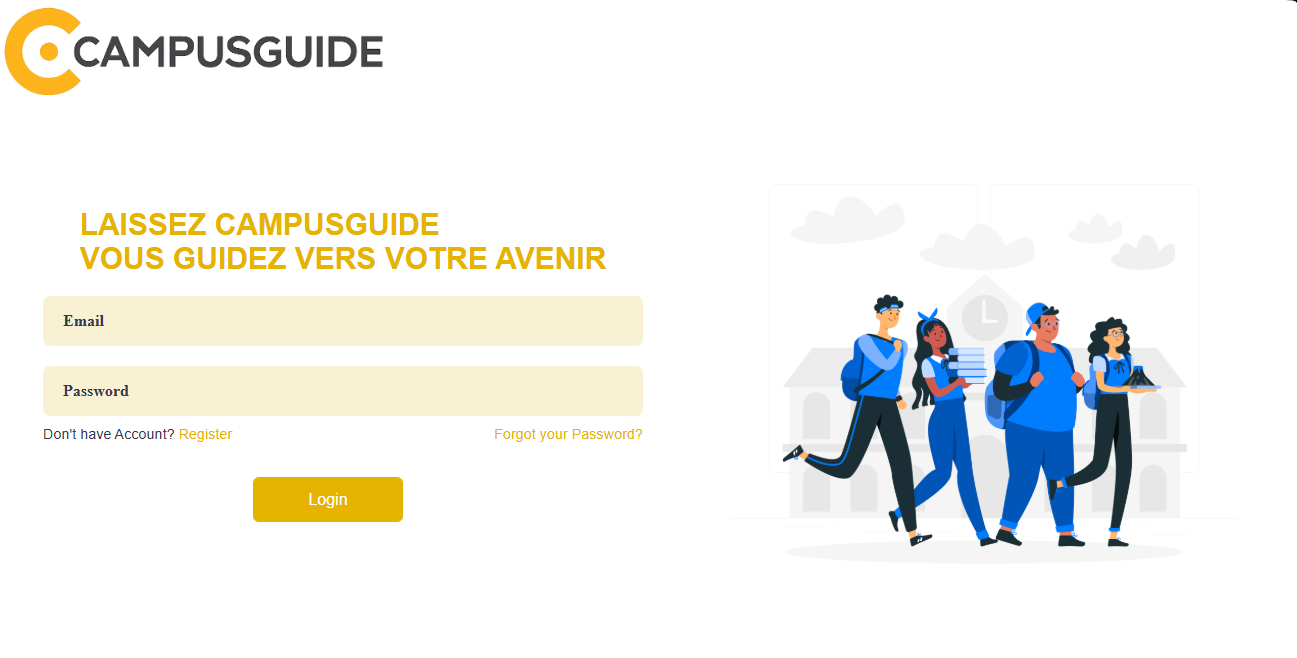
**Backend** :

* Initialisation du projet Laravel
* Modélisation de la base de données.
* Mise en place de l’authentification
* Développement des routes, contrôleurs et models.
* Connexion frontend ↔ backend
* Tests et débogage

## Fonctionnalités implémentées

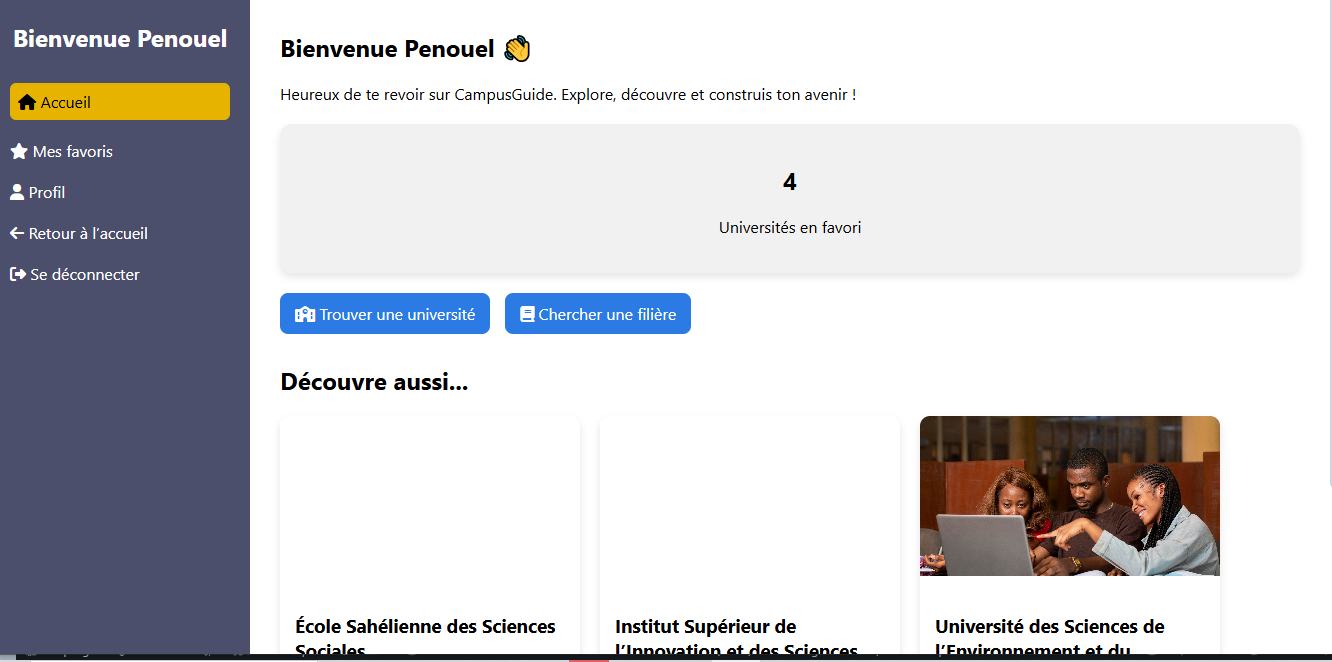
**1. Connexion / Inscription**

Création d’un compte utilisateur, connexion sécurisée, validation.

****

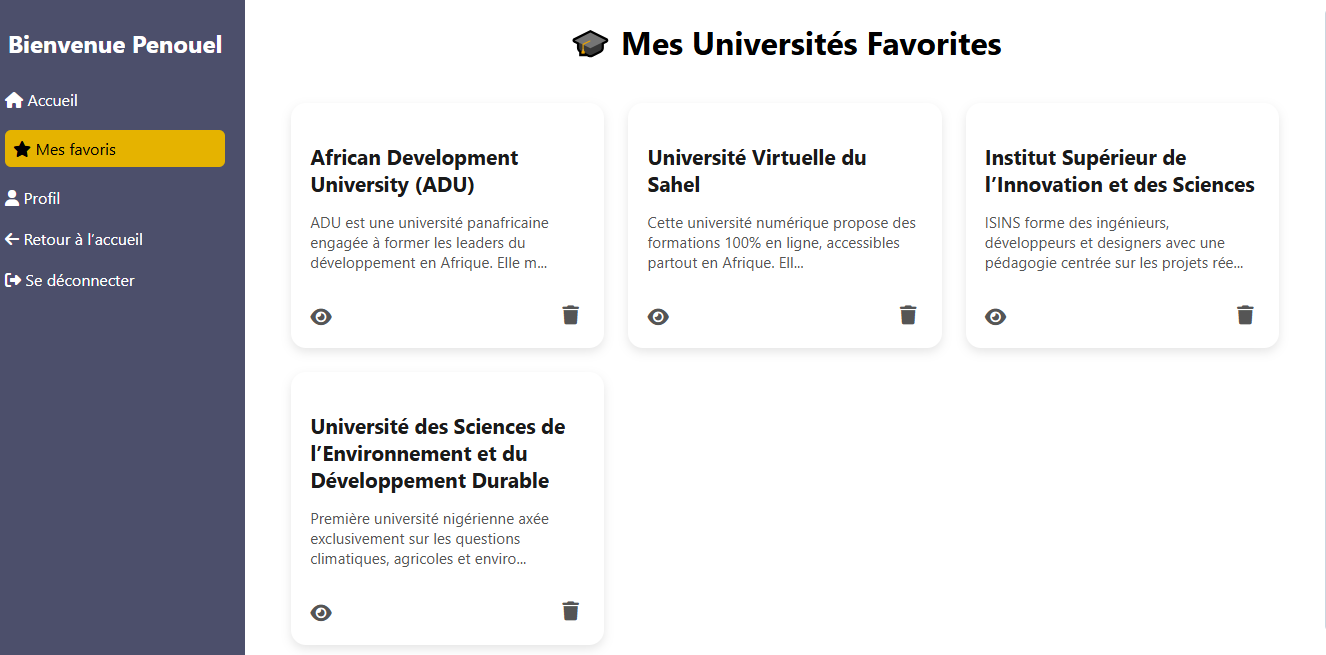
**2. Espace personnel**

Chaque utilisateur connecté dispose d’un espace personnel.



**3. Ajout / Suppression de favoris**

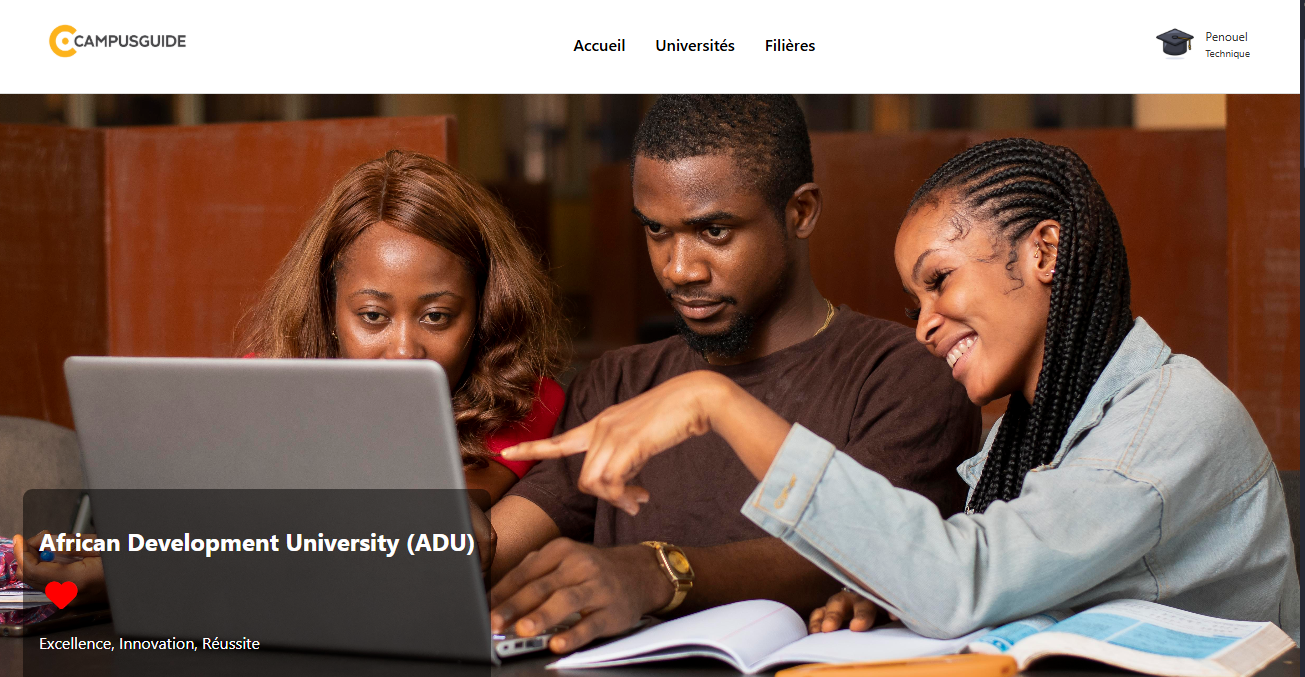
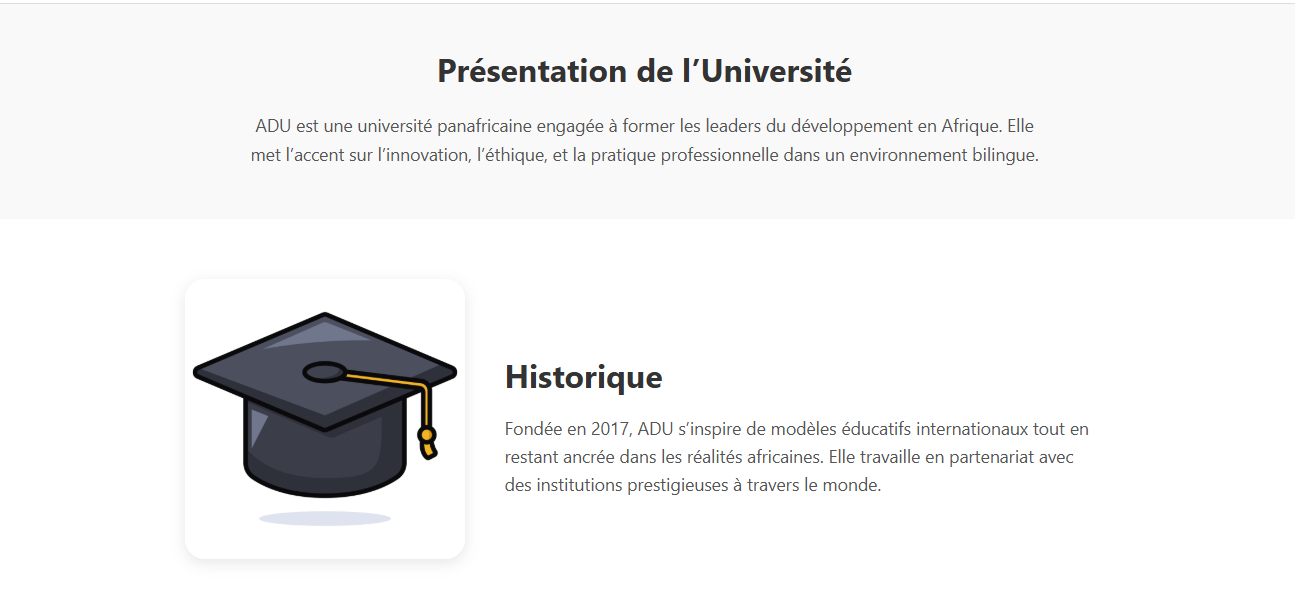
Un utilisateur peut ajouter ou retirer des universités à sa liste de favoris.



**4. Page université**

Chaque université dispose d’une fiche complète avec Nom, description, filières proposées, historique, média, débouchés. Un bouton **“S'inscrire”** redirigeant vers leur site officiel ou un lien externe.

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

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

**Une image contenant texte, commencement, carte de visite, capture d’écran

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

**5. Classements des Universités**

## 

**6. Tableau de bord Administrateur**

**Une image contenant texte, diagramme, capture d’écran, Tracé

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

## 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);

        }

    }

}

## Problèmes rencontrés et comment tu les as résolus

* Première utilisation de Laravel donc c’était un peu compliquer mais avec la lecture de la documentation, les videos 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. Ensuite, j’ai mis en place un formulaire dynamique pour ajouter les filières à une université lors de sa création.
* L’utilisateur devait saisir une université ou une filière et valider manuellement.J’ai intégré une barre de recherche avec suggestions instantanées (autocomplétion 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. J’ai aussi vérifié la logique d’ajout/suppression dans le contrôleur Laravel et corrigé les erreurs de logique sur la table de favoris (relation many-to-many).

# Tests (Phase Test)

## 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).

## 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.

## Résultats des tests + corrections

## Retours utilisateurs

# Déploiement

## Où est hébergée l’application

## Étapes d’installation / configuration

## Défis ou limites rencontrées

## Suggestions pour un déploiement future

# Conclusion & Travaux futurs

## Bilan de ton projet

Le projet **CampusGuide** a été développé avec succès en utilisant PHP pour le backend et HTML/CSS pour le frontend. Les principales fonctionnalités, telles que la recherche d'universités, l’inscription, la connexion et le quiz de recommandation, ont été intégrées et testées. Cependant, certaines améliorations sont encore nécessaires pour affiner l'expérience utilisateur et ajouter plus de contenu.

## Objectif atteint ou non ?

Oui, l'objectif principal d’offrir une plateforme de guide pour les élèves post-Bac regroupant le maximum d’information sur les universités locales au Niger a été atteint . Le projet a permis de comprendre les défis du développement d’application web mais aussi comment solutionner un réel problème.

## Ce que j’ai appris

* L’utilisation de PHP en backend avec Laravel
* L’importance de bien structurer une base de données et de gérer les relations entre les entités.

## Idées d’améliorations ou d’évolutions

* **Mobile** : Créer une version mobile de l’application pour une meilleure accessibilité.
* **Plus d’écoles** : Ajouter un nombre plus important d’universités et de filières pour améliorer la diversité des suggestions.

# Références

OpenAI. (2023). *ChatGPT technical documentation*. Retrieved from <https://openai.com/chatgpt>

W3Schools. (n.d.). *HTML, CSS, JavaScript tutorials*. Retrieved April 29, 2025, from <https://www.w3schools.com>

Laravel. (n.d.). *Laravel Documentation (v10)*. Retrieved April 29, 2025, from <https://laravel.com/docs>

CampusNiger. (n.d.). *Plateforme d’orientation universitaire au Niger*. Retrieved from <https://campusniger.ne>

Karatou Post Bac. (n.d.). *Plateforme d’aide à l’orientation au Niger*. Retrieved from <https://karatoupostbac.ne>

Parcoursup. (2024). *Portail officiel d’orientation post-bac en France*. Retrieved from <https://www.parcoursup.fr/>

Scikit-learn. (2025). *User Guide — scikit-learn 1.6.1 documentation*.  
<https://www.scikit-learn.org/stable/user_guide.html>

CommonApp <https://www.commonapp.org>

Articles : <https://www.studiokalangou.org> <https://www.uddm.edu.ne>

guide-dorientation-bac-abe-2022-pdf dorientation-bac-abe-2022-pdf Date Milestone and Gannt Chart <https://www.docdroid.net/KptzN2a/guide>

# Annexes