**Projet à rendre : Dernier délai le 08 janvier 2025**

Étude de cas en framework laravel qui porte sur la gestion automatisée de l'élection du bureau de l'amicale des étudiants: étudiants, filières, départements, candidats.

Ci-dessous les points principaux de cette étude de cas :

**1. Modèles (Models) :**

etudiant, Departement, Programme, Candidat, et Vote pour gérer les entités et leurs relations.

Chaque étudiant est associé à un département et une filière.

Les votes sont enregistrés avec une contrainte d'unicité par étudiant.

**2. Contrôleurs (Controllers) :**

CandidatController pour afficher la liste des candidats avec leurs informations.

VoteController pour enregistrer un vote.

StatisticsController pour générer des statistiques sur les votes, candidats, départements et filières.

**3. Routes :**

Routes pour les opérations principales : récupération des candidats, enregistrement des votes et affichage des statistiques.

**4. Fonctionnalités clés :**

Affichage des informations détaillées sur chaque candidat.

Restriction pour éviter qu'un étudiant vote plusieurs fois.

Statistiques claires sur les départements, les filières et les votes par candidat.

**Interface utilisateur (UI/UX)**

Pour rendre l'application plus accessible :

Utilisation de Laravel Breeze ou Jetstream :

Cela fournit des fonctionnalités d'authentification et une interface frontale préconfigurée.

Ajout de vues avec Blade :

**Créez des pages comme :**

* Une liste des candidats avec des boutons pour voter.
* Une page de statistiques avec des graphiques interactifs.

**Intégration de frameworks frontaux :**

TailwindCSS ou Bootstrap pour un design réactif.

Chart.js pour des graphiques interactifs (votes par candidat, répartition par filière, etc.).

**Sécurité avancée**

Pour garantir un bon fonctionnement et la protection des données :

* Authentification et autorisation :

Protéger les routes avec des middlewares (auth ou can pour les autorisations).

Implémenter différents rôles utilisateur (par exemple : étudiant, administrateur).

**Validation des données :**

Vérifiez les inputs au niveau du contrôleur pour éviter les soumissions malveillantes.

Ajoutez des validations strictes dans les formulaires (par exemple, un étudiant ne peut voter qu'une fois).

**Protection CSRF et SQL Injection :**

Laravel gère déjà cela en grande partie, mais assurez-vous d'utiliser les fonctionnalités de validation et d'Eloquent ORM.

**Fonctionnalités supplémentaires**

Voici quelques ajouts pour enrichir l'expérience utilisateur :

* **Résultats en temps réel :**

Affichez en temps réel les résultats de l’élection grâce à une mise à jour automatique avec Laravel Echo et Websockets.

**Notification des utilisateurs :**

Informez les étudiants des mises à jour importantes par email avec Laravel Notifications ou Mail.

**Export des statistiques :**

Permettez l'export des données (en CSV, PDF) pour analyse. Par exemple, le nombre de votes par département ou filière.