BENDAHGANE SAMI **212131085571**

Rapport sur le projet de tableau de bord universitaire (PRAVAN University)

Introduction

Le projet présente un tableau de bord interactif conçu pour gérer et visualiser les données universitaires d'une manière efficace. À travers ce tableau de bord, les administrateurs peuvent naviguer entre différentes sections, comme les spécialités, les étudiants et leurs résultats académiques, et consulter des graphiques intuitifs qui représentent les données sous différents angles.

L'application est basée sur Flask pour le backend et utilise la base de données **db_university** pour stocker toutes les informations. Le frontend inclut l'utilisation de HTML, CSS et JavaScript, avec l'intégration de la bibliothèque Chart.js pour les visualisations.

Structure du projet

1. **Base de données : db_university**
   - La base de données **db_university**
2. **Interface utilisateur**
   - L'interface principale inclut une barre latérale qui permet de naviguer entre les différentes sections du tableau de bord : « Accueil », « Spécialités », « Étudiants » et « Résultats ».
   - Une section centrale affiche les données et graphiques en fonction des actions de l'utilisateur.
3. **Graphiques dynamiques avec Chart.js**
   - **Diagramme à barres :** Représente les performances académiques des étudiants dans différentes matières. Cela permet de comparer les résultats entre différents groupes ou spécialités.
     Pour ce projet, il est utilisé pour représenter le nombre d'étudiants par année, spécialité et genre.
   - **Graphique linéaire :** Montre l'évolution des performances d'un étudiant ou d'un groupe au cours des semestres.
     Pour ce projet il est utilisé pour représenter le tot de succes et d'échec par spécialité.
   - **Diagramme circulaire (Pie Chart) :** Illustré les proportions des étudiants selon leurs résultats (par exemple, distinction, mention bien, mention passable).
     Pour ce projet il est utilisé pour représenter la moyenne des moyennes par spécialité.
   - **Diagramme en anneau (Doughnut Chart) :** Semblable au graphique circulaire, mais avec un espace central pour des annotations ou des informations supplémentaires.
     Pour ce projet il est utilisé pour représenter la répartition par genre et par spécialité des étudiants

- **Diagramme Bubble Chart :** Le **Bubble Chart** est un graphique utilisé pour représenter des données multidimensionnelles. Contrairement aux graphiques classiques à deux dimensions, le Bubble Chart utilise trois dimensions : les axes X et Y pour représenter deux variables, et la taille des bulles pour représenter une troisième variable.
  Pour ce projet il est utilisé pour représenter les étudiants ayant des moyennes d'excellence.

4. **Fonctionnalités clés**
   - **Navigation :** Une barre latérale simple pour naviguer entre les sections.
   - **Actualisation :** Bouton « Actualiser » permettant de mettre à jour les données visibles sur la page.
   - **Résultats dynamiques :** Les graphiques changent en fonction des données filtrées.

Visualisations et leur utilité

- Les graphiques apportent une dimension visuelle aux données académiques et permettent aux administrateurs d'analyser rapidement des tendances ou des problèmes.
- Par exemple, le graphique à barres est idéal pour détecter les matières où les étudiants rencontrent des difficultés.
- Le diagramme en anneau fournit une vue d'ensemble des performances générales des étudiants dans une cohorte donnée.

Fonctionnement d'AJAX

AJAX repose sur l'utilisation de l'objet `XMLHttpRequest` (ou la méthode moderne `fetch`), qui permet d'envoyer des requêtes HTTP au serveur et de récupérer des réponses en arrière-plan, sans perturber l'interface utilisateur.

1. **Création d'une requête AJAX** :
   - Une requête HTTP est initiée par le client (navigateur) en utilisant `XMLHttpRequest` ou `fetch`.
   - L'URL cible est spécifiée, ainsi que la méthode (GET, POST, etc.), et dans le cas de `XMLHttpRequest`, l'état de la requête est suivi à l'aide de `onreadystatechange`.
2. **Communication avec le serveur** :
   - Lorsque l'utilisateur effectue une action (clic, entrée de données, etc.), la requête AJAX est envoyée au serveur sans nécessiter un rechargement de la page.
   - Le serveur reçoit la requête, effectue les traitements nécessaires (par exemple, récupération ou manipulation de données), puis renvoie une réponse (généralement sous forme de JSON ou XML).
3. **Traitement de la réponse côté client** :

- Une fois la réponse reçue, le client peut l'utiliser pour mettre à jour des parties spécifiques de la page (graphiques, tableaux, etc.) sans avoir à rafraîchir l'ensemble du contenu.
- Par exemple, les données renvoyées par le serveur peuvent être utilisées pour mettre à jour un graphique avec les nouvelles valeurs, comme dans ton code.

Conclusion

Ce projet offre une solution robuste et intuitive pour gérer les données universitaires. L’utilisation de Flask et Chart.js garantit une expérience utilisateur fluide, tandis que la base de données **db_university** assure une organisation efficace des informations. Avec ces outils, les administrateurs peuvent non seulement consulter les données de manière pratique, mais aussi prendre des décisions basées sur des analyses visuelles approfondies.