Elections américaines (Version Web) - v1.2

Dans cette version, vous allez créer une **route dynamique** permettant d'éditer les statistiques d'un état spécifique. Cette fonctionnalité repose sur un formulaire HTML qui affiche les données actuelles de l'état et permet de les modifier. Une fois les modifications soumises, elles seront sauvegardées dans le fichier JSON d'origine.

Projet Flask v1.2 : Modifier les Statistiques d'un État avec un Formulaire

---

Tu trouveras plus d'informations sur la création de formulaires en HTML sur cette page.

Fonctionnalités à implémenter

Nouvelle Route : `/etats/<code_etat>/edit`

1. **Affichage du formulaire**
   Créez une route dynamique `/etats/<code_etat>/edit` qui :
   - Charge les données de l'état correspondant au `code` fourni dans l'URL.
   - Affiche un formulaire HTML pré-rempli avec les données actuelles de cet état.
2. **Soumission du formulaire**
   Configurez la route pour accepter les méthodes **GET** (pour afficher le formulaire) et **POST** (pour soumettre les modifications). Le formulaire doit être soumis vers la **même route** (`/etats/<code_etat>/edit`) pour des raisons de simplicité et de clarté dans la gestion des données.

   Pourquoi poster vers la même route ?
   Cela permet d'avoir une gestion centralisée des actions liées à l'édition d'un état. La même route peut afficher le formulaire ou enregistrer les modifications, selon la méthode HTTP utilisée (GET ou POST).

3. **Traitement et validation des données**
   - Avant d'enregistrer, vérifiez que **tous les champs du formulaire sont remplis**. Si un champ est vide, affichez un message d'erreur clair.
   - Si tous les champs sont valides, modifiez les données de l'état dans la liste JSON et sauvegardez les modifications dans le fichier d'origine.

Formulaire HTML : Champs recommandés

Pour les données de l'état, voici les types de champs HTML les plus adaptés :

Donnée	Type de champ	Explication
Nom	`<input type="text">`	Un champ de texte pour le nom complet.
Code	`<input type="text">`	Champ de texte (format 2 lettres).
Nombre grands électeurs	`<input type="number" min="0">`	Nombre entier non négatif.
Vainqueur	`<select>`	Liste déroulante (`d`, `r`, `autre`).
Tendance 2020	`<input type="text">`	Champ texte pour une description succincte.
Variation grands électeurs	`<input type="number">`	Entier positif, négatif ou nul.

Sauvegarde des données JSON

Voici le code Python pour sauvegarder les modifications dans un fichier JSON :

```python
import json

# Exemple de fonction pour enregistrer les données modifiées
def sauvegarder_donnees(data, fichier='data.json'):
    with open(fichier, 'w', encoding='utf-8') as file:
        json.dump(data, file, indent=4, ensure_ascii=False)
```

- `data` : La liste complète des états modifiée.
- `fichier` : Le chemin du fichier JSON d'origine.

Vous pouvez appeler cette fonction après avoir modifié les données d'un état.

Questions de réflexion

1. **Facile ?**
   Est-ce que la manipulation et la validation des données vous semblent simples à mettre en œuvre ?

2. **Pratique ?**
   Trouvez-vous pratique de poster vers la même route pour afficher le formulaire et traiter les données ?

3. **Problème de concurrence ?**
   Que se passe-t-il si **deux utilisateurs** modifient le même état en même temps ? Comment géreriez-vous ce type de conflit ?

Cette fonctionnalité introduit des concepts clés comme la gestion des formulaires, la validation des données, et la persistance dans des fichiers. Cela vous prépare également à réfléchir à des problématiques comme la gestion des conflits dans un environnement multi-utilisateur.