**Exercice : Validation et enregistrement d'un formulaire d'inscription utilisateur avec respect des principes SOLID**

**Objectif :** Créer un script JavaScript qui valide les données d'un formulaire d'inscription utilisateur (email, mot de passe, date de naissance, numéro fétiche) en suivant les critères spécifiés, et les enregistre dans le `localStorage` si elles sont valides. En cas d'erreur de validation, afficher un message d'erreur à l'utilisateur.

**Instructions :**

1. Créez un formulaire HTML comprenant les champs suivants :
    - Email (type "email")
    - Mot de passe (type "password")
    - Date de naissance (type "date")
    - Numéro fétiche (type "number")
2. Créez un module JavaScript avec une méthode pour chaque critère de validation, en suivant le principe SOLID. Par exemple, vous pouvez avoir un module avec les fonctions suivantes :

- `emailValidator`
- `passwordValidator`
- `dateValidator`
- `numberValidator`

3. Dans un module principal, importez les modules de validation et créez une fonction qui sera appelée lors de la soumission du formulaire.
4. Dans cette fonction, récupérez les valeurs saisies par l'utilisateur dans les champs du formulaire.
5. Utilisez le module de validation pour valider les données selon les critères spécifiés :
    - L'email doit être une adresse email valide en utilisant `emailValidator`.
    - Le mot de passe doit comporter au moins 8 caractères, au moins une majuscule, une minuscule et un caractère spécial en utilisant `passwordValidator`.
    - La date de naissance doit être une date valide en utilisant `dateValidator`.
    - Le numéro fétiche doit être une chaîne non vide en utilisant `numberValidator`.
6. Si toutes les données sont valides, enregistrez-les dans le `localStorage` sous forme d'un objet JSON contenant les informations de l'utilisateur.
7. Si des données ne sont pas valides, affichez un message d'erreur approprié à l'utilisateur pour l'informer des problèmes dans son formulaire.
8. Assurez-vous que le formulaire ne soit pas soumis si des erreurs sont présentes, afin de permettre à l'utilisateur de les corriger.

**Conseils :**

- Utilisez les principes de modularité pour organiser votre code en modules réutilisables.
- Utilisez des expressions régulières pour valider l'email et le mot de passe.
- Assurez-vous de gérer les imports et les exports de modules en conséquence.