Vue JS

La Validation

Il est important de vérifier que l'utilisateur ait correctement remplis les champs ainsi que vérifier leurs contenus. Vee-Validator est une librairie qui se charge de cela.

Un bon composant d'entrée utilisateur est composé de deux éléments :

- Le champ d'entrée utilisateur (texte, nombre, ...)
- Un espace pour afficher les erreurs relatives à ce champ

Pour améliorer, l'UX d'une application, nous devons dire à l'utilisateur quand il y a une erreur :

- "Ce champ est obligatoire"
- "Veuillez entrer votre numéro de téléphone au format xxx-xxxx"
- "Veuillez entrer une adresse e-mail valide"
- "Votre mot de passe doit comporter entre 8 et 30 caractères"

Attention :

- Une vérification du côté utilisateur n'est aucunement fiable, il faut toujours avoir une vérification côté serveur. **Ceinture et Bretelles**
- C'est une pré-étape du processus de vérification.
- Pas de JavaScript = pas de validation.

A quoi ça sert de valider côté utilisateur ?

- Obtenir de bonnes données dans un bon format
- Protéger nos utilisateurs
- Nous protéger nous-mêmes

Vee-Validator

Pour commencer avec Vee-Validator, il faut l'installer

```
npm install vee-validate
```

Il faut ensuite l'ajouter au projet.

```
// src/main.js
import * as VeeValidate from 'vee-validate';
(...)
Vue.use(VeeValidate);
```

Nous voilà prêt à utiliser Vee-Validator. Créons un nouveau composant que l'on va nommer `Contact.vue`.

Ajoutons un input ainsi qu'un espace dédié aux erreurs.

```
// src/components/Contact.vue
<ValidationProvider name="email" rules="required|email" v-slot="{ errors }">
  <input v-model="email" type="text">
  <span>{{ errors[0] }}</span>
</ValidationProvider>
(...)
import { ValidationProvider } from 'vee-validate';
(...)
components:{
  ValidationProvider
},
```

Afin de ne pas alourdir votre application Vee Validate ne charge pas l'ensemble des règles.

Il faut les charger une à une:

```
// src/components/Contact.vue
import { extend } from 'vee-validate';
import { required, email } from 'vee-validate/dist/rules';

extend('required', required);
extend('email', email);
```

Pour connaître l'ensemble des règles par défaut c'est ici

Et voilà.

Exercice: 30min

- Créer une page "Contact"
- Ajouter la page au router
- Implémenter un formulaire de contact
- Valider les différents champs utilisateur

Pour finir

Rappelez-vous de toujours aider l'utilisateur à corriger les données qu'il saisit. Pour ce faire, assurez-vous de toujours :

- Afficher des messages d'erreur explicites.
- Être tolérant sur le format des données à envoyer.
- Indiquer exactement où est l'erreur.