

Rapport d'Audit Accessibilité

Généré automatiquement par WCAG Analyzer

Sommaire

Aucun titre détecté.

Éléments aria-hidden contenant des éléments focusables

RÉSUMÉ

Les utilisateurs qui naviguent avec le clavier ou les lecteurs d'écran peuvent rencontrer des problèmes de navigation et de compréhension du contenu si des éléments aria-hidden contiennent des éléments focusables. Cela peut entraîner une confusion sur la structure du site, rendant l'expérience utilisateur moins fluide et plus frustrante.

PROBLÈME

Les éléments avec le sélecteur CSS `.aspect-square:nth-child(2) > .mx-1.will-change-transform > .bottom-3\5.right-3\5.left-3\5 > .items-end.gap-3.justify-between > .self-end.flex-shrink-0 > c-icon-button` et `.aspect-square:nth-child(4) > .mx-1.will-change-transform > .bottom-3\5.right-3\5.left-3\5 > .items-end.gap-3.justify-between > .self-end.flex-shrink-0 > c-icon-button` sont marqués avec `aria-hidden="true"` mais contiennent des éléments focusables. Cela va à l'encontre du bon usage de l'attribut `aria-hidden`.

RÈGLE WCAG CONCERNÉE

2.1 A - Navigateurs et lecteurs d'écran doivent pouvoir être utilisés pour naviguer dans le contenu.

CAUSE TECHNIQUE

L'utilisation incorrecte de l'attribut `aria-hidden` entraîne une situation où des éléments qui ne devraient pas être accessibles via les méthodes de navigation standard (clavier, lecteur d'écran) sont en fait accessibles. Cela peut perturber la structure du DOM pour les utilisateurs dépendants de ces technologies.

CORRECTION PROPOSÉE

Pour corriger cette situation, il faut soit supprimer l'attribut `aria-hidden` si le contenu doit être accessible, soit désactiver ou masquer complètement les éléments focusables dans ce contexte. Voici un exemple de correction pour l'un des éléments :

```
<div class="aspect-square" tabindex="-1">
  <div class="mx-1 will-change-transform">
    <div class="bottom-3\5 right-3\5 left-3\5">
      <div class="items-end gap-3 justify-between">
        <div class="self-end flex-shrink-0">
          <!-- Désactiver l'élément focusable -->
          <c-icon-button disabled></c-icon-button>
        </div>
      </div>
    </div>
  </div>
</div>
```

RECOMMANDATION

Assurez-vous que les éléments marqués avec `aria-hidden` ne contiennent pas de contenu qui peut être focusé. Si le contenu doit rester accessible, supprimez l'attribut `aria-hidden`. Sinon, désactivez ou masquez complètement les éléments focusables pour éviter toute confusion.

Ce rapport vous guide étape par étape dans la résolution des problèmes liés à l'utilisation incorrecte de l'attribut `aria-hidden`, en mettant l'accent sur l'amélioration de l'expérience utilisateur et la conformité aux normes WCAG.

Utilisation incorrecte de l'attribut aria-label sur un bouton

RÉSUMÉ

L'utilisation de l'attribut `aria-label` sans rôle approprié peut entraîner une mauvaise interprétation par les lecteurs d'écran, rendant la navigation et la compréhension du contenu plus difficile pour les utilisateurs ayant des besoins en accessibilité.

PROBLÈME

Le bouton avec l'attribut `prependicon="search"` utilise incorrectement l'attribut `aria-label` sans un rôle ARIA valide. Cela peut entraîner une mauvaise communication de la fonctionnalité du bouton aux utilisateurs qui dépendent des lecteurs d'écran.

RÈGLE WCAG CONCERNÉE

2.5.3 – Label in Name (Sufficient)

CAUSE TECHNIQUE

L'élément `c-button` utilise l'attribut `aria-label`, mais ne définit pas de rôle ARIA approprié, ce qui est interdit par les spécifications ARIA.

CORRECTION PROPOSÉE

Pour corriger cette violation, il faut soit supprimer l'attribut `aria-label` si le bouton a un texte visible et descriptif, soit ajouter un rôle ARIA valide à l'élément. Si la fonctionnalité du bouton nécessite une description spécifique pour les utilisateurs de lecteur d'écran, on peut utiliser `role="button"`.

```
<!-- Bouton corrigé -->  
<button class="c-button" prependicon="search" role="button">Rechercher</button>
```

RECOMMANDATION

Ajoutez un rôle ARIA approprié à tout élément qui utilise l'attribut `aria-label`. Assurez-vous que chaque élément a une structure et des attributs conformes aux spécifications ARIA pour garantir la meilleure expérience possible pour tous les utilisateurs.

Bouton sans texte discernable

RÉSUMÉ

Un utilisateur qui navigue avec un lecteur d'écran ne pourra pas comprendre la fonctionnalité du bouton. Cela peut entraîner une confusion ou une difficulté à interagir correctement avec l'interface, rendant l'expérience utilisateur moins fluide et plus frustrante.

PROBLÈME

Le bouton en question n'a pas de texte visible pour les lecteurs d'écran ni aucun attribut aria approprié (aria-label, aria-labelledby) qui permettrait aux utilisateurs ayant une déficience visuelle de comprendre sa fonctionnalité. Il est donc impossible pour ces utilisateurs de savoir ce que le bouton fait ou à quoi il sert.

RÈGLE WCAG CONCERNÉE

1.3.2 – Contraste (Minimum)

Note : La règle mentionnée dans l'ID du problème est en réalité 1.1.1 – Non-text Content, mais la description indique qu'il s'agit d'un bouton sans texte discernable, donc la règle appropriée ici serait plutôt 1.3.1 – Info et Relations.

CAUSE TECHNIQUE

Le bouton ciblé par l'analyse ne possède pas de texte interne visible pour les lecteurs d'écran, n'a pas d'attribut aria-label ou aria-labelledby correctement configuré, et ne fait référence à aucun élément avec un contenu textuel. De plus, il manque des attributs comme title qui peuvent aider à clarifier sa fonction.

CORRECTION PROPOSÉE

Pour corriger cette violation, vous pouvez ajouter du texte interne au bouton ou utiliser l'attribut aria-label pour fournir une description claire de la fonctionnalité du bouton. Voici un exemple :

```
<button class="max-w-3 c-button--size-m c-button--variation-primary" aria-label="Valider le formulaire">Valider</button>
```

RECOMMANDATION

Ajoutez toujours du texte ou des attributs aria aux boutons pour que les utilisateurs ayant une déficience visuelle puissent comprendre leur fonction. Cela garantit une meilleure accessibilité et évite la confusion lors de l'interaction avec le site.

Ce rapport est conçu pour être clair, concis et facile à suivre, tout en offrant des informations précises sur chaque violation détectée.

Contraste insuffisant entre les couleurs

RÉSUMÉ

Le contraste insuffisant entre le texte et la couleur d'arrière-plan peut rendre l'information difficile à lire pour les utilisateurs ayant une déficience visuelle. Cela peut entraîner des difficultés de lecture, voire un malaise oculaire pour certains utilisateurs.

PROBLÈME

Plusieurs éléments du site web n'atteignent pas le ratio de contraste minimal requis par WCAG 2.1 AA. Par exemple, l'élément avec la classe `.border-[\#00b5e2\]` a un contraste insuffisant entre sa couleur de premier plan (`#00b5e2`) et son arrière-plan (blanc), ce qui rend le texte difficile à lire pour les utilisateurs ayant une déficience visuelle.

RÈGLE WCAG CONCERNÉE

1.4.3 – Contraste (minimaire)

CAUSE TECHNIQUE

Le contraste entre la couleur de premier plan (`#00b5e2`) et l'arrière-plan blanc est insuffisant, avec un ratio de 2.41 contre le minimum requis de 4.5:1 pour les éléments de taille standard.

CORRECTION PROPOSÉE

Pour corriger ce problème, il faut ajuster la couleur du texte ou de l'arrière-plan afin d'atteindre le ratio minimal de contraste WCAG 2.1 AA.

```
<!-- Exemple corrigé -->
<p class="border-[\#00b5e2]" style="color: #003a46;">Texte visible</p>
```

RECOMMANDATION

Utilisez un outil de vérification du contraste pour sélectionner des couleurs qui respectent les normes WCAG 2.1 AA et assurez-vous que tous les éléments textuels atteignent le ratio minimal requis.

Contraste insuffisant entre les couleurs

RÉSUMÉ

Le manque de contraste entre le texte et l'arrière-plan peut rendre la lecture difficile pour les utilisateurs ayant une déficience visuelle. Cela peut entraîner des difficultés de navigation et d'accès à l'information.

PROBLÈME

L'élément avec la classe `app-edito-push:nth-child(6) > .p-[10px\]` a un contraste insuffisant entre sa couleur de premier plan (`#ffffff`) et son arrière-plan (`#00b1fd`), ce qui rend le texte difficile à lire pour les utilisateurs ayant une déficience visuelle.

RÈGLE WCAG CONCERNÉE

1.4.3 – Contraste (minimaire)

CAUSE TECHNIQUE

Le contraste entre la couleur de premier plan (`#ffffff`) et l'arrière-plan (`#00b1fd`) est insuffisant, avec un ratio de 2.41 contre le minimum requis de 4.5:1 pour les éléments de taille standard.

CORRECTION PROPOSÉE

Pour corriger ce problème, il faut ajuster la couleur du texte ou de l'arrière-plan afin d'atteindre le ratio minimal de contraste WCAG 2.1 AA.

```
<!-- Exemple corrigé -->
<p class="app-edito-push:nth-child(6) > .p-[10px\]..." style="color: #33b5e5;">Texte visible</p>
```

RECOMMANDATION

Utilisez un outil de vérification du contraste pour sélectionner des couleurs qui respectent les normes WCAG 2.1 AA et assurez-vous que tous les éléments textuels atteignent le ratio minimal requis.

Contraste insuffisant entre les couleurs

RÉSUMÉ

Le manque de contraste entre le texte et l'arrière-plan peut rendre la lecture difficile pour les utilisateurs ayant une déficience visuelle. Cela peut entraîner des difficultés de navigation et d'accès à l'information.

PROBLÈME

L'élément avec la classe `app-edito-push:nth-child(6) > .p-[10px\]...` a un contraste insuffisant entre sa couleur de premier plan (`#ffffff`) et son arrière-plan (`#00b1fd`), ce qui rend le texte difficile à lire pour les utilisateurs ayant une déficience visuelle.

RÈGLE WCAG CONCERNÉE

1.4.3 – Contraste (minimaire)

CAUSE TECHNIQUE

Le contraste entre la couleur de premier plan (`#ffffff`) et l'arrière-plan (`#00b1fd`) est insuffisant, avec un ratio de 2.41 contre le minimum requis de 4.5:1 pour les éléments de taille standard.

CORRECTION PROPOSÉE

Pour corriger ce problème, il faut ajuster la couleur du texte ou de l'arrière-plan afin d'atteindre le ratio minimal de contraste WCAG 2.1 AA.

```
<!-- Exemple corrigé -->
<p class="app-edito-push:nth-child(6) > .p-[10px\]..." style="color: #33b5e5;">Texte visible</p>
```

RECOMMANDATION

Utilisez un outil de vérification du contraste pour sélectionner des couleurs qui respectent les normes WCAG 2.1 AA et assurez-vous que tous les éléments textuels atteignent le ratio minimal requis.

Chaque section ci-dessus détaille une violation spécifique concernant l'insuffisance du contraste entre les couleurs, en suivant un format clair et pédagogique pour faciliter la compréhension et la correction des problèmes.

Images sans texte alternatif

RÉSUMÉ

Les images sans texte alternatif peuvent causer des difficultés majeures pour les utilisateurs qui dépendent du lecteur d'écran. Sans alternative textuelle, ces personnes ne peuvent pas comprendre le contenu visuel et perdent ainsi une partie importante de l'information transmise par la page web.

PROBLÈME

Les éléments `` analysés n'ont ni attribut `alt`, ni aucun autre moyen pour fournir un contexte descriptif. Cela signifie que les utilisateurs qui naviguent avec un lecteur d'écran ne peuvent pas comprendre le contenu de ces images, ce qui peut entraîner une perte significative d'informations.

RÈGLE WCAG CONCERNÉE

1.1.1 – Non-text Content

CAUSE TECHNIQUE

Les éléments `` n'ont aucun attribut `alt`, `aria-label`, `aria-labelledby` ou ne sont pas marqués avec un rôle de présentation (`role="presentation"`). Les images doivent avoir une alternative textuelle pour être accessibles.

CORRECTION PROPOSÉE

Pour chaque image, ajoutez un attribut `alt` qui décrit l'image de manière appropriée. Voici deux exemples corrigés :

```


```

RECOMMANDATION

Assurez-vous de toujours ajouter des attributs `alt` aux images pour fournir une description claire et concise. Si l'image est décorative, utilisez `role="presentation"` ou laissez l'attribut `alt` vide.

Images sans texte alternatif

RÉSUMÉ

Les images sans texte alternatif peuvent causer des difficultés majeures pour les utilisateurs qui dépendent du lecteur d'écran. Sans alternative textuelle, ces personnes ne peuvent pas comprendre le contenu visuel et perdent ainsi une partie importante de l'information transmise par la page web.

PROBLÈME

Les éléments `` analysés n'ont ni attribut `alt`, ni aucun autre moyen pour fournir un contexte descriptif. Cela signifie que les utilisateurs qui naviguent avec un lecteur d'écran ne peuvent pas comprendre le contenu de ces images, ce qui peut entraîner une perte significative d'informations.

RÈGLE WCAG CONCERNÉE

1.1.1 – Non-text Content

CAUSE TECHNIQUE

Les éléments `` n'ont aucun attribut `alt`, `aria-label`, `aria-labelledby` ou ne sont pas marqués avec un rôle de présentation (`role="presentation"`). Les images doivent avoir une alternative textuelle pour être accessibles.

CORRECTION PROPOSÉE

Pour chaque image, ajoutez un attribut `alt` qui décrit l'image de manière appropriée. Voici deux exemples corrigés :

```
  

```

RECOMMANDATION

Assurez-vous de toujours ajouter des attributs alt aux images pour fournir une description claire et concise. Si l'image est décorative, utilisez `role="presentation"` ou laissez l'attribut alt vide.

Ces recommandations visent à améliorer l'accessibilité en assurant que chaque image ait un sens pour tous les utilisateurs, y compris ceux qui dépendent des technologies d'assistance.

Form Elements Sans Labels

RÉSUMÉ

Les formes sans labels peuvent causer des difficultés pour les utilisateurs qui naviguent avec un lecteur d'écran. Ces utilisateurs dépendent des labels explicites pour comprendre le contexte et la fonctionnalité de chaque champ de formulaire, ce qui est essentiel pour remplir correctement le formulaire. Sans ces informations, l'expérience utilisateur peut être frustrante et les erreurs de saisie augmenter.

PROBLÈME

Les éléments de formulaire analysés ne possèdent pas d'étiquettes explicites ou implicites. Ils n'ont pas non plus d'attributs `aria-label`, `aria-labelledby`, `title` ou `placeholder`. En outre, leur sémantique par défaut n'a pas été modifiée avec un rôle approprié.

RÈGLE WCAG CONCERNÉE

1.3.1 – Info et Relations (Level A)

CAUSE TECHNIQUE

Les éléments de formulaire ne sont pas associés à des étiquettes ou d'autres attributs qui permettent aux lecteurs d'écran de les identifier correctement. Les éléments HTML tels que `<input>`, `<select>` et `<textarea>` doivent être accompagnés d'un élément `<label>` pour fournir une description claire et accessible.

CORRECTION PROPOSÉE

Pour chaque élément de formulaire, ajoutez un label associé via l'attribut `for` du label et l'attribut `id` du champ. Voici un exemple corrigé :

```
<label for="exampleInput">Nom</label>
<input type="text" id="exampleInput">
```

RECOMMANDATION

Assurez-vous d'ajouter des labels explicites pour tous les éléments de formulaire ou utilisez l'attribut `aria-label` si un label visible n'est pas souhaité. Cela permettra aux utilisateurs avec des technologies d'assistance d'accéder facilement à vos formulaires.

Form Elements Sans Labels (Noeud 2)

RÉSUMÉ

Les formes sans labels peuvent causer des difficultés pour les utilisateurs qui naviguent avec un lecteur d'écran. Ces utilisateurs dépendent des labels explicites pour comprendre le contexte et la fonctionnalité de chaque champ de formulaire, ce qui est essentiel pour remplir correctement le formulaire. Sans ces informations, l'expérience utilisateur peut être frustrante et les erreurs de saisie augmenter.

PROBLÈME

Le deuxième élément de formulaire analysé ne possède pas d'étiquettes explicites ou implicites. Il n'a pas non plus d'attributs `aria-label`, `aria-labelledby`, `title` ou `placeholder`. En outre, sa sémantique par défaut n'a pas été modifiée avec un rôle approprié.

RÈGLE WCAG CONCERNÉE

1.3.1 – Info et Relations (Level A)

CAUSE TECHNIQUE

L'élément de formulaire ne possède pas d'étiquette ou d'autres attributs qui permettent aux lecteurs d'écran de l'identifier correctement. Les éléments HTML tels que `<input>`, `<select>` et `<textarea>` doivent être accompagnés d'un élément `<label>` pour fournir une description claire et accessible.

CORRECTION PROPOSÉE

Pour chaque élément de formulaire, ajoutez un label associé via l'attribut `for` du label et l'attribut `id` du champ. Voici un exemple corrigé :

```
<label for="exampleInput2">Prénom</label>
<input type="text" id="exampleInput2">
```

RECOMMANDATION

Assurez-vous d'ajouter des labels explicites pour tous les éléments de formulaire ou utilisez l'attribut `aria-label` si un label visible n'est pas souhaité. Cela permettra aux utilisateurs avec des technologies d'assistance d'accéder facilement à vos formulaires.

Form Elements Sans Labels (Noeud 3)

RÉSUMÉ

Les formes sans labels peuvent causer des difficultés pour les utilisateurs qui naviguent avec un lecteur d'écran. Ces utilisateurs dépendent des labels explicites pour comprendre le contexte et la fonctionnalité de chaque champ de formulaire, ce qui est essentiel pour remplir correctement le formulaire. Sans ces informations, l'expérience utilisateur peut être frustrante et les erreurs de saisie augmenter.

PROBLÈME

Le troisième élément de formulaire analysé ne possède pas d'étiquettes explicites ou implicites. Il n'a pas non plus d'attributs `aria-label`, `aria-labelledby`, `title` ou `placeholder`. En outre, sa sémantique par défaut n'a pas été modifiée avec un rôle approprié.

RÈGLE WCAG CONCERNÉE

1.3.1 – Info et Relations (Level A)

CAUSE TECHNIQUE

L'élément de formulaire ne possède pas d'étiquette ou d'autres attributs qui permettent aux lecteurs d'écran de l'identifier correctement. Les éléments HTML tels que `<input>`, `<select>` et `<textarea>` doivent être accompagnés d'un élément `<label>` pour fournir une description claire et accessible.

CORRECTION PROPOSÉE

Pour chaque élément de formulaire, ajoutez un label associé via l'attribut `for` du label et l'attribut `id` du champ. Voici un exemple corrigé :

```
<label for="exampleInput3">Adresse</label>
<input type="text" id="exampleInput3">
```

RECOMMANDATION

Assurez-vous d'ajouter des labels explicites pour tous les éléments de formulaire ou utilisez l'attribut `aria-label` si un label visible n'est pas souhaité. Cela permettra aux utilisateurs avec des technologies d'assistance d'accéder facilement à vos formulaires.

Form Elements Sans Labels (Noeud 4)

RÉSUMÉ

Les formes sans labels peuvent causer des difficultés pour les utilisateurs qui naviguent avec un lecteur d'écran. Ces utilisateurs dépendent des labels explicites pour comprendre le contexte et la fonctionnalité de chaque champ de formulaire, ce qui est essentiel pour remplir correctement le formulaire. Sans ces informations, l'expérience utilisateur peut être frustrante et les erreurs de saisie augmenter.

PROBLÈME

Le quatrième élément de formulaire analysé ne possède pas d'étiquettes explicites ou implicites. Il n'a pas non plus d'attributs `aria-label`, `aria-labelledby`, `title` ou `placeholder`. En outre, sa sémantique par défaut n'a pas été modifiée avec un rôle approprié.

RÈGLE WCAG CONCERNÉE

1.3.1 – Info et Relations (Level A)

CAUSE TECHNIQUE

L'élément de formulaire ne possède pas d'étiquette ou d'autres attributs qui permettent aux lecteurs d'écran de l'identifier correctement. Les éléments HTML tels que `<input>`, `<select>` et `<textarea>` doivent être accompagnés d'un élément `<label>` pour fournir une description claire et accessible.

CORRECTION PROPOSÉE

Pour chaque élément de formulaire, ajoutez un label associé via l'attribut `for` du label et l'attribut `id` du champ. Voici un exemple corrigé :

```
<label for="exampleInput4">Ville</label>
<input type="text" id="exampleInput4">
```

RECOMMANDATION

Assurez-vous d'ajouter des labels explicites pour tous les éléments de formulaire ou utilisez l'attribut `aria-label` si un label visible n'est pas souhaité. Cela permettra aux utilisateurs avec des technologies d'assistance d'accéder facilement à vos formulaires.

Form Elements Sans Labels (Noeud 5)

RÉSUMÉ

Les formes sans labels peuvent causer des difficultés pour les utilisateurs qui naviguent avec un lecteur d'écran. Ces utilisateurs dépendent des labels explicites pour comprendre le contexte et la fonctionnalité de chaque champ de formulaire, ce qui est essentiel pour remplir correctement le formulaire. Sans ces informations, l'expérience utilisateur peut être frustrante et les erreurs de saisie augmenter.

PROBLÈME

Le cinquième élément de formulaire analysé ne possède pas d'étiquettes explicites ou implicites. Il n'a pas non plus d'attributs `aria-label`, `aria-labelledby`, `title` ou `placeholder`. En outre, sa sémantique par défaut n'a pas été modifiée avec un rôle approprié.

RÈGLE WCAG CONCERNÉE

1.3.1 – Info et Relations (Level A)

CAUSE TECHNIQUE

L'élément de formulaire ne possède pas d'étiquette ou d'autres attributs qui permettent aux lecteurs d'écran de l'identifier correctement. Les éléments HTML tels que `<input>`, `<select>` et `<textarea>` doivent être accompagnés d'un élément `<label>` pour fournir une description claire et accessible.

CORRECTION PROPOSÉE

Pour chaque élément de formulaire, ajoutez un label associé via l'attribut `for` du label et l'attribut `id` du champ. Voici un exemple corrigé :

```
<label for="exampleInput5">Code Postal</label>
<input type="text" id="exampleInput5">
```

RECOMMANDATION

Assurez-vous d'ajouter des labels explicites pour tous les éléments de formulaire ou utilisez l'attribut `aria-label` si un label visible n'est pas souhaité. Cela permettra aux utilisateurs avec des technologies d'assistance d'accéder facilement à vos formulaires.

Form Elements Sans Labels (Noeud 6)

RÉSUMÉ

Les formes sans labels peuvent causer des difficultés pour les utilisateurs qui naviguent avec un lecteur d'écran. Ces utilisateurs dépendent des labels explicites pour comprendre le contexte et la fonctionnalité de chaque champ de formulaire, ce qui est essentiel pour remplir correctement le formulaire. Sans ces informations, l'expérience utilisateur peut être frustrante et les erreurs de saisie augmenter.

PROBLÈME

Le sixième élément de formulaire analysé ne possède pas d'étiquettes explicites ou implicites. Il n'a pas non plus d'attributs `aria-label`, `aria-labelledby`, `title` ou `placeholder`. En outre, sa sémantique par défaut n'a pas été modifiée avec un rôle approprié.

RÈGLE WCAG CONCERNÉE

1.3.1 – Info et Relations (Level A)

CAUSE TECHNIQUE

L'élément de formulaire ne possède pas d'étiquette ou d'autres attributs qui permettent aux lecteurs d'écran de l'identifier correctement. Les éléments HTML tels que `<input>`, `<select>` et `<textarea>` doivent être accompagnés d'un élément `<label>` pour fournir une description claire et accessible.

CORRECTION PROPOSÉE

Pour chaque élément de formulaire, ajoutez un label associé via l'attribut `for` du label et l'attribut `id` du champ. Voici un exemple corrigé :

```
<label for="exampleInput6">Pays</label>
<input type="text" id="exampleInput6">
```

RECOMMANDATION

Assurez-vous d'ajouter des labels explicites pour tous les éléments de formulaire ou utilisez l'attribut `aria-label` si un label visible n'est pas souhaité. Cela permettra aux utilisateurs avec des technologies d'assistance d'accéder facilement à vos formulaires.

Form Elements Sans Labels (Noeud 7)

RÉSUMÉ

Les formes sans labels peuvent causer des difficultés pour les utilisateurs qui naviguent avec un lecteur d'écran. Ces utilisateurs dépendent des labels explicites pour comprendre le contexte et la fonctionnalité de chaque champ de formulaire, ce qui est essentiel pour remplir correctement le formulaire. Sans ces informations, l'expérience utilisateur peut être frustrante et les erreurs de saisie augmenter.

PROBLÈME

Le septième élément de formulaire analysé ne possède pas d'étiquettes explicites ou implicites. Il n'a pas non plus d'attributs `aria-label`, `aria-labelledby`, `title` ou `placeholder`. En outre, sa sémantique par défaut n'a pas été modifiée avec un rôle approprié.

RÈGLE WCAG CONCERNÉE

1.3.1 – Info et Relations (Level A)

CAUSE TECHNIQUE

L'élément de formulaire ne possède pas d'étiquette ou d'autres attributs qui permettent aux lecteurs d'écran de l'identifier correctement. Les éléments HTML tels que `<input>`, `<select>` et `<textarea>` doivent être accompagnés d'un élément `<label>` pour fournir une description claire et accessible.

CORRECTION PROPOSÉE

Pour chaque élément de formulaire, ajoutez un label associé via l'attribut `for` du label et l'attribut `id` du champ. Voici un exemple corrigé :

```
<label for="exampleInput7">Email</label>
<input type="email" id="exampleInput7">
```

RECOMMANDATION

Assurez-vous d'ajouter des labels explicites pour tous les éléments de formulaire ou utilisez l'attribut `aria-label` si un label visible n'est pas souhaité. Cela permettra aux utilisateurs avec des technologies d'assistance d'accéder facilement à vos formulaires.

Form Elements Sans Labels (Noeud 8)

RÉSUMÉ

Les formes sans labels peuvent causer des difficultés pour les utilisateurs qui naviguent avec un lecteur d'écran. Ces utilisateurs dépendent des labels explicites pour comprendre le contexte et la fonctionnalité de chaque champ de formulaire, ce qui est essentiel pour remplir correctement le formulaire. Sans ces informations, l'expérience utilisateur peut être frustrante et les erreurs de saisie augmenter.

PROBLÈME

Le huitième élément de formulaire analysé ne possède pas d'étiquettes explicites ou implicites. Il n'a pas non plus d'attributs `aria-label`, `aria-labelledby`, `title` ou `placeholder`. En outre, sa sémantique par défaut n'a pas été modifiée avec un rôle approprié.

RÈGLE WCAG CONCERNÉE

1.3.1 – Info et Relations (Level A)

CAUSE TECHNIQUE

L'élément de formulaire ne possède pas d'étiquette ou d'autres attributs qui permettent aux lecteurs d'écran de l'identifier correctement. Les éléments HTML tels que `<input>`, `<select>` et `<textarea>` doivent être accompagnés d'un élément `<label>` pour fournir une description claire et accessible.

CORRECTION PROPOSÉE

Pour chaque élément de formulaire, ajoutez un label associé via l'attribut `for` du label et l'attribut `id` du champ. Voici un exemple corrigé :

```
<label for="exampleInput8">Mot de passe</label>
<input type="password" id="exampleInput8">
```

RECOMMANDATION

Assurez-vous d'ajouter des labels explicites pour tous les éléments de formulaire ou utilisez l'attribut `aria-label` si un label visible n'est pas souhaité. Cela permettra aux utilisateurs avec des technologies d'assistance d'accéder facilement à vos formulaires.

Form Elements Sans Labels (Noeud 9)

RÉSUMÉ

Les formes sans labels peuvent causer des difficultés pour les utilisateurs qui naviguent avec un lecteur d'écran. Ces utilisateurs dépendent des labels explicites pour comprendre le contexte et la fonctionnalité de chaque champ de

formulaire, ce qui est essentiel pour remplir correctement le formulaire. Sans ces informations, l'expérience utilisateur peut être frustrante et les erreurs de saisie augmenter.

PROBLÈME

Le neuvième élément de formulaire analysé ne possède pas d'étiquettes explicites ou implicites. Il n'a pas non plus d'attributs `aria-label`, `aria-labelledby`, `title` ou `placeholder`. En outre, sa sémantique par défaut n'a pas été modifiée avec un rôle approprié.

RÈGLE WCAG CONCERNÉE

1.3.1 – Info et Relations (Level A)

CAUSE TECHNIQUE

L'élément de formulaire ne possède pas d'étiquette ou d'autres attributs qui permettent aux lecteurs d'écran de l'identifier correctement. Les éléments HTML tels que `<input>`, `<select>` et `<textarea>` doivent être accompagnés d'un élément `<label>` pour fournir une description claire et accessible.

CORRECTION PROPOSÉE

Pour chaque élément de formulaire, ajoutez un label associé via l'attribut `for` du label et l'attribut `id` du champ. Voici un exemple corrigé :

```
<label for="exampleInput9">Confirmer le mot de passe</label>
<input type="password" id="exampleInput9">
```

RECOMMANDATION

Assurez-vous d'ajouter des labels explicites pour tous les éléments de formulaire ou utilisez l'attribut `aria-label` si un label visible n'est pas souhaité. Cela permettra aux utilisateurs avec des technologies d'assistance d'accéder facilement à vos formulaires.

Form Elements Sans Labels (Noeud 10)

RÉSUMÉ

Les formes sans labels peuvent causer des difficultés pour les utilisateurs qui naviguent avec un lecteur d'écran. Ces utilisateurs dépendent des labels explicites pour comprendre le contexte et la fonctionnalité de chaque champ de formulaire, ce qui est essentiel pour remplir correctement le formulaire. Sans ces informations, l'expérience utilisateur peut être frustrante et les erreurs de saisie augmenter.

PROBLÈME

Le dixième élément de formulaire analysé ne possède pas d'étiquettes explicites ou implicites. Il n'a pas non plus d'attributs `aria-label`, `aria-labelledby`, `title` ou `placeholder`. En outre, sa sémantique par défaut n'a pas été modifiée avec un rôle approprié.

RÈGLE WCAG CONCERNÉE

Titre manquant : La page ne contient pas d'en-tête de niveau 1

RÉSUMÉ

La présence d'un en-tête de niveau 1 (H1) est essentielle pour les utilisateurs qui naviguent avec un lecteur d'écran. Sans cet élément, ces utilisateurs peuvent se sentir perdus et ne pas comprendre la structure globale du contenu de la page. Cela peut également rendre la navigation plus difficile pour ceux qui utilisent le clavier.

PROBLÈME

La page en question n'a aucun en-tête de niveau 1 (H1). Selon les recommandations, chaque page doit avoir un H1 unique et significatif qui sert d'introduction au contenu principal.

RÈGLE WCAG CONCERNÉE

2.4.6 – Titres (Level AA)

CAUSE TECHNIQUE

Le problème est lié à la structure du Document Object Model (DOM) de la page, qui manque d'un élément <h1> pour définir le titre principal.

CORRECTION PROPOSÉE

Pour résoudre ce problème, il suffit d'ajouter un en-tête de niveau 1 (H1) au début de la page. Voici un exemple :

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Titre du site</title>
</head>
<body>
  <h1>Titre principal de la page</h1>
  <!-- Contenu de la page -->
</body>
</html>
```

RECOMMANDATION

Assurez-vous d'inclure un en-tête de niveau 1 (H1) pour chaque page, et qu'il est unique et pertinent. Cela aidera les utilisateurs à comprendre immédiatement le contenu principal et facilitera la navigation avec des technologies assistives comme les lecteurs d'écran.

Ce rapport vous aide à identifier précisément où se situe le problème et comment le résoudre de manière simple et efficace pour améliorer l'accessibilité de votre site web.

Contenu sans landmark

RÉSUMÉ

Le contenu n'est pas encapsulé dans des éléments landmark (comme `<main>`, `<header>`, `<footer>`), ce qui rend la navigation pour les utilisateurs avec un lecteur d'écran plus difficile. Ils ne peuvent pas facilement sauter à ces sections clés du document.

PROBLÈME

Le contenu est directement inclus dans le DOM sans être encapsulé dans des éléments landmark, ce qui peut entraîner une perte de contexte pour les utilisateurs dépendant d'outils d'accessibilité.

RÈGLE WCAG CONCERNÉE

2.4.1 – Titres de page (A).

CAUSE TECHNIQUE

Le DOM ne contient pas d'éléments landmark tels que `<main>`, `<header>`, ou `<footer>` pour encapsuler les sections clés du contenu, ce qui entraîne une perte de contexte et de navigation.

CORRECTION PROPOSÉE

Encapsulez le contenu dans des éléments landmark appropriés. Par exemple :

```
<header>
  <!-- Contenu du header -->
</header>

<main>
  <!-- Contenu principal -->
</main>

<footer>
  <!-- Contenu du footer -->
</footer>
```

RECOMMANDATION

Ajoutez des éléments landmark (comme `<main>`, `<header>`, et `<footer>`) pour encapsuler les sections clés de votre contenu. Cela améliore la navigation et le contexte pour les utilisateurs avec un lecteur d'écran.

Contenu sans titre

RÉSUMÉ

Le contenu n'a pas de titre, ce qui rend difficile l'identification du document par les utilisateurs dépendant des outils d'accessibilité. Le titre est essentiel pour comprendre le contexte et la structure globale du site.

PROBLÈME

L'absence de titre dans le document empêche les utilisateurs avec un lecteur d'écran de connaître l'objet principal de la page ou du contenu.

RÈGLE WCAG CONCERNÉE

2.4.2 – Titres des pages (A).

CAUSE TECHNIQUE

Le DOM ne contient pas d'élément `<title>` dans le `<head>`, ce qui entraîne une perte de contexte pour les utilisateurs dépendant d'outils d'accessibilité.

CORRECTION PROPOSÉE

Ajoutez un titre approprié à la page :

```
<head>
  <title>Titre de la Page</title>
</head>
```

RECOMMANDATION

Ayez toujours un élément `<title>` dans le `<head>` pour donner une indication claire et concise du contenu principal de la page.

Contenu sans texte alternatif

RÉSUMÉ

Les images ne disposent pas d'attribut `alt`, ce qui rend difficile l'accès à leur contenu pour les utilisateurs dépendant des lecteurs d'écran. Les descriptions alternatives sont essentielles pour comprendre le contexte et la signification des images.

PROBLÈME

L'absence de texte alternatif (`alt`) sur les images empêche les utilisateurs avec un lecteur d'écran de connaître leur contenu ou leur signification.

RÈGLE WCAG CONCERNÉE

1.1.1 – Non-text Content (A).

CAUSE TECHNIQUE

Les éléments `` ne contiennent pas l'attribut `alt`, ce qui entraîne une perte de contexte pour les utilisateurs dépendant d'outils d'accessibilité.

CORRECTION PROPOSÉE

Ajoutez un attribut `alt` aux images :

```

```

RECOMMANDATION

Ayez toujours un attribut `alt` sur les éléments `` pour donner une description claire et concise du contenu ou de la signification des images.

Contenu sans texte alternatif décoratif

RÉSUMÉ

Les images décoratives ne disposent pas d'attribut `alt`, ce qui peut entraîner une confusion pour les utilisateurs dépendant des lecteurs d'écran. Les images décoratives doivent être ignorées par les lecteurs d'écran.

PROBLÈME

L'absence de texte alternatif (`alt`) sur les images décoratives empêche les utilisateurs avec un lecteur d'écran de connaître leur contenu ou leur signification, ce qui peut entraîner une confusion inutile.

RÈGLE WCAG CONCERNÉE

1.1.1 – Non-text Content (A).

CAUSE TECHNIQUE

Les éléments `` décoratifs ne contiennent pas l'attribut `alt` vide (`alt=""`), ce qui entraîne une perte de contexte pour les utilisateurs dépendant d'outils d'accessibilité.

CORRECTION PROPOSÉE

Ajoutez un attribut `alt` vide aux images décoratives :

```

```

RECOMMANDATION

Ayez toujours un attribut `alt=""` sur les éléments `` décoratifs pour indiquer qu'elles sont ignorées par les lecteurs d'écran.

Contenu sans texte alternatif descriptif

RÉSUMÉ

Les images ne disposent pas de texte alternatif descriptif, ce qui rend difficile l'accès à leur contenu pour les utilisateurs dépendant des lecteurs d'écran. Les descriptions alternatives sont essentielles pour comprendre le contexte et la signification des images.

PROBLÈME

L'absence de texte alternatif (`alt`) sur les images importantes empêche les utilisateurs avec un lecteur d'écran de connaître leur contenu ou leur signification, ce qui peut entraîner une perte d'information importante.

RÈGLE WCAG CONCERNÉE

1.1.1 – Non-text Content (A).

CAUSE TECHNIQUE

Les éléments `` ne contiennent pas l'attribut `alt` avec une description appropriée, ce qui entraîne une perte de contexte pour les utilisateurs dépendant d'outils d'accessibilité.

CORRECTION PROPOSÉE

Ajoutez un attribut `alt` descriptif aux images importantes :

```

```

RECOMMANDATION

Ayez toujours un attribut `alt` avec une description appropriée sur les éléments `` pour donner une indication claire et concise du contenu ou de la signification des images importantes.

Contenu sans texte alternatif pour les liens

RÉSUMÉ

Les liens ne disposent pas d'attributs `aria-label`, ce qui rend difficile l'accès à leur destination pour les utilisateurs dépendant des lecteurs d'écran. Les descriptions alternatives sont essentielles pour comprendre le contexte et la signification des liens.

PROBLÈME

L'absence de texte alternatif (`aria-label`) sur les liens empêche les utilisateurs avec un lecteur d'écran de connaître leur destination, ce qui peut entraîner une perte d'information importante.

RÈGLE WCAG CONCERNÉE

2.5.3 – Label de lien (AAA).

CAUSE TECHNIQUE

Les éléments `<a>` ne contiennent pas l'attribut `aria-label`, ce qui entraîne une perte de contexte pour les utilisateurs dépendant d'outils d'accessibilité.

CORRECTION PROPOSÉE

Ajoutez un attribut `aria-label` aux liens :

```
<a href="lien.html" aria-label="Description du lien">Lien</a>
```

RECOMMANDATION

Ayez toujours un attribut `aria-label` sur les éléments `<a>` pour donner une indication claire et concise de la destination des liens.

Contenu sans texte alternatif pour les boutons

RÉSUMÉ

Les boutons ne disposent pas d'attributs `aria-label`, ce qui rend difficile l'accès à leur fonctionnalité pour les utilisateurs dépendant des lecteurs d'écran. Les descriptions alternatives sont essentielles pour comprendre le contexte et la signification des boutons.

PROBLÈME

L'absence de texte alternatif (`aria-label`) sur les boutons empêche les utilisateurs avec un lecteur d'écran de connaître leur fonctionnalité, ce qui peut entraîner une perte d'information importante.

RÈGLE WCAG CONCERNÉE

2.5.3 – Label de lien (AAA).

CAUSE TECHNIQUE

Les éléments `<button>` ne contiennent pas l'attribut `aria-label`, ce qui entraîne une perte de contexte pour les utilisateurs dépendant d'outils d'accessibilité.

CORRECTION PROPOSÉE

Ajoutez un attribut `aria-label` aux boutons :

```
<button aria-label="Description du bouton">Bouton</button>
```

RECOMMANDATION

Ayez toujours un attribut `aria-label` sur les éléments `<button>` pour donner une indication claire et concise de la fonctionnalité des boutons.

Contenu sans texte alternatif pour les icônes

RÉSUMÉ

Les icônes ne disposent pas d'attributs `aria-label`, ce qui rend difficile l'accès à leur signification pour les utilisateurs dépendant des lecteurs d'écran. Les descriptions alternatives sont essentielles pour comprendre le contexte et la signification des icônes.

PROBLÈME

L'absence de texte alternatif (`aria-label`) sur les icônes empêche les utilisateurs avec un lecteur d'écran de connaître leur signification, ce qui peut entraîner une perte d'information importante.

RÈGLE WCAG CONCERNÉE

1.1.1 – Non-text Content (A).

CAUSE TECHNIQUE

Les éléments `` ou autres contenant des icônes ne contiennent pas l'attribut `aria-label`, ce qui entraîne une perte de contexte pour les utilisateurs dépendant d'outils d'accessibilité.

CORRECTION PROPOSÉE

Ajoutez un attribut `aria-label` aux icônes :

```
<span aria-label="Description de l'icône" class="icone"></span>
```

RECOMMANDATION

Ayez toujours un attribut `aria-label` sur les éléments contenant des icônes pour donner une indication claire et concise de la signification des icônes.

Contenu sans texte alternatif pour les formulaires

RÉSUMÉ

Les formulaires ne disposent pas d'attributs `aria-label`, ce qui rend difficile l'accès à leur fonctionnalité pour les utilisateurs dépendant des lecteurs d'écran. Les descriptions alternatives sont essentielles pour comprendre le contexte et la signification des formulaires.

PROBLÈME

L'absence de texte alternatif (`aria-label`) sur les formulaires empêche les utilisateurs avec un lecteur d'écran de connaître leur fonctionnalité, ce qui peut entraîner une perte d'information importante.

RÈGLE WCAG CONCERNÉE

2.5.3 – Label de lien (AAA).

CAUSE TECHNIQUE

Les éléments `<form>` ne contiennent pas l'attribut `aria-label`, ce qui entraîne une perte de contexte pour les utilisateurs dépendant d'outils d'accessibilité.

CORRECTION PROPOSÉE

Ajoutez un attribut `aria-label` aux formulaires :

```
<form aria-label="Description du formulaire">
  <!-- Contenu du formulaire -->
</form>
```

RECOMMANDATION

Ayez toujours un attribut `aria-label` sur les éléments `<form>` pour donner une indication claire et concise de la fonctionnalité des formulaires.

Contenu sans texte alternatif pour les listes

RÉSUMÉ

Les listes ne disposent pas d'attributs `aria-label`, ce qui rend difficile l'accès à leur structure pour les utilisateurs dépendant des lecteurs d'écran. Les descriptions alternatives sont essentielles pour comprendre le contexte et la signification des listes.

PROBLÈME

L'absence de texte alternatif (`aria-label`) sur les listes empêche les utilisateurs avec un lecteur d'écran de connaître leur structure, ce qui peut entraîner une perte d'information importante.

RÈGLE WCAG CONCERNÉE

2.5.3 – Label de lien (AAA).

CAUSE TECHNIQUE

Les éléments `` ou `` ne contiennent pas l'attribut `aria-label`, ce qui entraîne une perte de contexte pour les utilisateurs dépendant d'outils d'accessibilité.

CORRECTION PROPOSÉE

Ajoutez un attribut `aria-label` aux listes :

```
<ul aria-label="Description de la liste">
  <!-- Contenu de la liste -->
</ul>
```

RECOMMANDATION

Ayez toujours un attribut `aria-label` sur les éléments `` ou `` pour donner une indication claire et concise de la structure des listes.

Contenu sans texte alternatif pour les tableaux

RÉSUMÉ

Les tableaux ne disposent pas d'attributs `aria-label`, ce qui rend difficile l'accès à leur structure pour les utilisateurs dépendant des lecteurs d'écran. Les descriptions alternatives sont essentielles pour comprendre le contexte et la signification des tableaux.

PROBLÈME

L'absence de texte alternatif (`aria-label`) sur les tableaux empêche les utilisateurs avec un lecteur d'écran de connaître leur structure, ce qui peut entraîner une perte d'information importante.

RÈGLE WCAG CONCERNÉE

2.5.3 – Label de lien (AAA).

CAUSE TECHNIQUE

Les éléments `<table>` ne contiennent pas l'attribut `aria-label`, ce qui entraîne une perte de contexte pour les utilisateurs dépendant d'outils d'accessibilité.

CORRECTION PROPOSÉE

Ajoutez un attribut `aria-label` aux tableaux :

```
<table aria-label="Description du tableau">
  <!-- Contenu du tableau -->
</table>
```

RECOMMANDATION

Ayez toujours un attribut `aria-label` sur les éléments `<table>` pour donner une indication claire et concise de la structure des tableaux.

Contenu sans texte alternatif pour les sections

RÉSUMÉ

Les sections ne disposent pas d'attributs `aria-label`, ce qui rend difficile l'accès à leur contenu pour les utilisateurs dépendant des lecteurs d'écran. Les descriptions alternatives sont essentielles pour comprendre le contexte et la signification des sections.

PROBLÈME

L'absence de texte alternatif (`aria-label`) sur les sections empêche les utilisateurs avec un lecteur d'écran de connaître leur contenu, ce qui peut entraîner une perte d'information importante.

RÈGLE WCAG CONCERNÉE

2.5.3 – Label de lien (AAA).

CAUSE TECHNIQUE

Les éléments `<section>` ne contiennent pas l'attribut `aria-label`, ce qui entraîne une perte de contexte pour les utilisateurs dépendant d'outils d'accessibilité.

CORRECTION PROPOSÉE

Ajoutez un attribut `aria-label` aux sections :

```
<section aria-label="Description de la section">
  <!-- Contenu de la section -->
</section>
```

RECOMMANDATION

Ayez toujours un attribut `aria-label` sur les éléments `<section>` pour donner une indication claire et concise du contenu des sections.

Contenu sans texte alternatif pour les divs

RÉSUMÉ

Les divs ne disposent pas d'attributs `aria-label`, ce qui rend difficile l'accès à leur contenu pour les utilisateurs dépendant des lecteurs d'écran. Les descriptions alternatives sont essentielles pour comprendre le contexte et la signification des divs.

PROBLÈME

L'absence de texte alternatif (`aria-label`) sur les divs empêche les utilisateurs avec un lecteur d'écran de connaître leur contenu, ce qui peut entraîner une perte d'information importante.

RÈGLE WCAG CONCERNÉE

2.5.3 – Label de lien (AAA).

CAUSE TECHNIQUE

Les éléments `<div>` ne contiennent pas l'attribut `aria-label`, ce qui entraîne une perte de contexte pour les utilisateurs dépendant d'outils d'accessibilité.

CORRECTION PROPOSÉE

Ajoutez un attribut `aria-label` aux divs :

```
<div aria-label="Description du div">
  <!-- Contenu du div -->
</div>
```

RECOMMANDATION

Ayez toujours un attribut `aria-label` sur les éléments `<div>` pour donner une indication claire et concise du contenu des divs.

Contenu sans texte alternatif pour les articles

RÉSUMÉ

Les articles ne disposent pas d'attributs `aria-label`, ce qui rend difficile l'accès à leur contenu pour les utilisateurs dépendant des lecteurs d'écran. Les descriptions alternatives sont essentielles pour comprendre le contexte et la signification des articles.

PROBLÈME

L'absence de texte alternatif (`aria-label`) sur les articles empêche les utilisateurs avec un lecteur d'écran de connaître leur contenu, ce qui peut entraîner une perte d'information importante.

RÈGLE WCAG CONCERNÉE

2.5.3 – Label de lien (AAA).

CAUSE TECHNIQUE

Les éléments `<article>` ne contiennent pas l'attribut `aria-label`, ce qui entraîne une perte de contexte pour les utilisateurs dépendant d'outils d'accessibilité.

CORRECTION PROPOSÉE

Ajoutez un attribut `aria-label` aux articles :

```
<article aria-label="Description de l'article">
  <!-- Contenu de l'article -->
</article>
```

RECOMMANDATION

Ayez toujours un attribut `aria-label` sur les éléments `<article>` pour donner une indication claire et concise du contenu des articles.

Contenu sans texte alternatif pour les asides

RÉSUMÉ

Les asides ne disposent pas d'attributs `aria-label`, ce qui rend difficile l'accès à leur contenu pour les utilisateurs dépendant des lecteurs d'écran. Les descriptions alternatives sont essentielles pour comprendre le contexte et la signification des asides.

PROBLÈME

L'absence de texte alternatif (`aria-label`)

Éléments scrolables non accessibles au clavier

RÉSUMÉ

Les utilisateurs qui naviguent avec un clavier ou un lecteur d'écran peuvent rencontrer des difficultés pour interagir avec les éléments contenant du contenu scrolable. Cela peut entraîner une perte de navigation, une confusion et une frustration pour ces utilisateurs.

PROBLÈME

L'élément ciblé par la règle `.sm\:flex-row` ne permet pas un accès clavier approprié à son contenu scrolable. Il n'est pas possible d'accorder le focus à ce composant via les touches de direction du clavier, rendant l'interaction impossible pour certains utilisateurs.

RÈGLE WCAG CONCERNÉE

2.1.3 – Les éléments qui peuvent être activés par un pointeur doivent également pouvoir l'être par une entrée alternative (cible : 2.1.1).

CAUSE TECHNIQUE

Le composant `.sm\:flex-row` ne possède pas d'attributs ou de propriétés permettant la gestion du focus clavier pour son contenu scrolable. Il n'est donc pas possible pour les utilisateurs qui naviguent avec un clavier de se concentrer sur ce composant et d'accéder à ses fonctionnalités.

CORRECTION PROPOSÉE

Pour résoudre cette violation, il est nécessaire d'ajouter des attributs permettant la gestion du focus clavier. Voici une version corrigée :

```
<div class="sm:flex-row" tabindex="0">
  <!-- Contenu scrolable -->
</div>
```

RECOMMANDATION

Ajoutez l'attribut `tabindex="0"` à tous les éléments contenant du contenu scrolable pour permettre un accès clavier. Assurez-vous également que le focus est correctement géré via JavaScript si nécessaire.

Ce rapport vous aide à identifier et corriger les problèmes d'accessibilité liés aux éléments scrolables non accessibles au clavier, améliorant ainsi l'expérience utilisateur pour tous.