ChatGPT

Audit CSS PériMap – Analyse et corrections

Étape 1 – Cartographie du CSS

Le projet comprend un grand nombre de feuilles CSS organisées en modules :

- **modules/base** : variables, reset, typographie, animations… Le fichier `reset.css` définit la structure globale (`html`, `body`, `#app-view-root`) avec un scroll porté par `html` et un `body` en flex-column. Des corrections successives ont déjà été appliquées pour réduire les effets de rebond sur mobile et empêcher les barres grises (`overscroll-behavior: none;`, `overflow-y: auto !important`).
- **modules/layout** : gère la grille, le header, la navigation et leurs variantes (`header.css`, `header.dropdown.css`, `navigation.css`, `grid.css`). Le header est collant (sticky) et passe en `fixed` en-dessous de 900 px. Une barre de navigation inférieure (`.bottom-nav`) est fixée en bas de l'écran sur mobile avec un z-index très élevé et des propriétés telles que `touch-action: none;` et `overscroll-behavior: none;`.
- **modules/pages** : styles spécifiques aux pages (itinéraire, carte, horaires…). Dans `itinerary.css` et `map.css`, plusieurs sections utilisent des conteneurs `fixed` et `overflow: hidden` pour bloquer le défilement lorsque des overlays sont actifs (#detail-map, #itinerary-detail-container). Le conteneur des résultats comporte un panneau latéral scrollable (`overflow-y: auto`) et une carte en flex.
- **modules/components** : composants réutilisables (boutons, cartes, popups, formulaires, modals). Les listes d'autocomplétion sont positionnées absolument sous leurs champs (`top: 100%`) avec un z-index très élevé pour passer par-dessus les autres éléments. Le fichier `timepicker.css` définit une règle générique pour les listes déroulantes (".timepicker-dropdown, .combobox-dropdown, .autocomplete-dropdown") avec `inset: auto 0 0 0`, ce qui ancre la liste au bas du bloc au lieu de l'aligner sous l'input.
- **modules/utilities** : classes utilitaires (display, spacing, stacking, mobile). Le fichier `mobile.css` applique des adaptations spécifiques en dessous de 768 px, notamment pour les suggestions d'autocomplétion où le wrapper est forcé en `position: relative` pour ancrer les listes sous les champs.

Étape 2 – Recherche de patterns dangereux

Voici les points sensibles relevés lors de l'analyse :

1. **Scroll global :** Le fichier `reset.css` force `html { overflow-y: auto !important; }` et place `body` en flex sans scroll. Cette configuration est correcte mais l'utilisation systématique de `!important` rend difficile l'override par d'autres règles. Plusieurs overlays (`#itinerary-detail-container`, `#detail-map`) appliquent `overflow: hidden !important` pour bloquer le scroll quand ils sont visibles, ce qui est cohérent.

2. **Navigation inférieure fixe :** Dans `navigation.css`, la barre `.bottom-nav` est `position: fixed` avec `touch-action: none;` et `overscroll-behavior: none;`. Ces valeurs désactivent toute action tactile lorsqu'on glisse sur la barre; l'utilisateur qui commence

son geste sur la navigation ne peut plus faire défiler la page, ce qui donne l'impression que le scroll est bloqué.

3. **Listes d'autocomplétion :** Le fichier `components/timepicker.css` déclare pour les listes déroulantes :

```
.timepicker-dropdown,
.combobox-dropdown,
.autocomplete-dropdown {
  position: absolute;
  z-index: var(--z-dropdown);
  inset: auto 0 0 0;
}
```

Cette règle place la liste à `bottom: 0` de son conteneur sans définir de `top`. Si le parent n'a pas de `position: relative`, la dropdown peut se retrouver collée en bas de la fenêtre ou mal alignée par rapport au champ. Cela explique des popups d'autosuggestion flottantes ou mal positionnées.

1. **Popups et overlays :** Des conteneurs comme `#itinerary-detail-container` définissent `overflow: hidden !important` et `touch-action: none;` : ils bloquent tout scroll tant qu'ils restent actifs. Il est important que la fermeture de ces overlays remette bien l'état initial, sinon l'utilisateur se retrouve bloqué.

2. **Palette et cohérence visuelle :** Le projet utilise un grand nombre de couleurs (plus d'une centaine de codes hexadécimaux). Les couleurs principales sont bien définies via des variables CSS (`--primary-color`, `--secondary-color`, etc.), mais certaines valeurs codées en dur (ex : `#4285f4` dans `data-exporter.css`) pourraient être remplacées par des variables pour plus d'homogénéité.

Étape 3 – Plan d'action structuré

Priorité	Problème détecté	Proposition de correction
Haute	**Scroll bloqué sur mobile lorsque l'utilisateur touche la navigation inférieure.** La règle `touch-action: none; overscroll-behavior: none;` empêche toute propagation du geste de défilement.	Modifier la barre `.bottom-nav` pour permettre le défilement vertical : utiliser `touch-action: pan-y;` et `overscroll-behavior: contain;`. Cela autorise le scroll vertical tout en bloquant le rebond et en conservant l'intention du design.
Haute	**Popups d'autocomplétion mal alignées.** Les listes déroulantes (`.timepicker-dropdown`, `.combobox-dropdown`, `.autocomplete-dropdown`) sont ancrées avec `bottom: 0` via `inset: auto 0 0 0;`, ce qui les détache de leur champ.	Positionner explicitement les dropdowns juste sous leur champ : remplacer `inset: auto 0 0 0;` par `left: 0; right: 0; top: calc(100% + 0.25rem); bottom: auto;` afin qu'elles s'ouvrent sous l'input avec un léger espacement.

Priorité	Problème détecté	Proposition de correction
Moyenne	**Multiples couleurs codées en dur et bordures hétérogènes.** Certains fichiers utilisent des valeurs fixes de couleurs ou de `border-radius` différentes.	Rationaliser la palette en remplaçant les couleurs fixes par les variables du fichier `_config.css` (ex : `var(--primary-color)` pour les éléments principaux). Uniformiser les rayons (`--radius-md` ou `--radius-lg`) et les ombres (`--shadow-md`) pour les cartes et les boutons.
Moyenne	**Prolifération de règles `!important` sur les `overflow` et `height`.** Cela rend la maintenance difficile et peut masquer des bugs.	Supprimer l'usage de `!important` lorsque ce n'est pas indispensable. Par exemple, `html { overflow-y: auto !important; }` pourrait devenir `overflow-y: auto;` si aucune autre règle ne vient contredire ce comportement.
Basse	**Couverture des overlays** : certains conteneurs en `fixed` et `overflow: hidden` bloquent le scroll du site tant qu'ils sont actifs.	Vérifier dans le code JS que la classe `.is-active` ou `.hidden` est correctement appliquée/retirée. Ajouter éventuellement une classe `.scroll-lock` sur `body` plutôt que d'appliquer des `overflow: hidden !important` multiples.

Étape 4 – Corrections concrètes apportées

Deux correctifs ciblés ont été appliqués pour traiter les problèmes les plus urgents.

1. **Alignement des dropdowns (autocomplétion et timepicker)**

Dans `components/timepicker.css`, la règle pour les listes déroulantes a été réécrite. Au lieu d'utiliser la propriété `inset: auto 0 0 0;` qui ancre le bloc au bas du conteneur, les dropdowns sont désormais positionnés juste sous leur déclencheur :

```
- .timepicker-dropdown,
- .combobox-dropdown,
- .autocomplete-dropdown {
-   position: absolute;
-   z-index: var(--z-dropdown);
-   inset: auto 0 0 0;
- }
+ /* Position des dropdowns : sous l'input avec un petit espace */
+ .timepicker-dropdown,
+ .combobox-dropdown,
+ .autocomplete-dropdown {
+   position: absolute;
```

```
+   z-index: var(--z-dropdown);
+   left: 0;
+   right: 0;
+   /* Place la liste juste en dessous du champ */
+   top: calc(100% + 0.25rem);
+   bottom: auto;
+ }
```

Intention : s'assurer que les listes d'autosuggestion apparaissent toujours immédiatement sous le champ concerné, même si leur conteneur ne dispose pas d'un positionnement relatif. Cela évite que la liste flotte au bas de l'écran ou se trouve masquée par d'autres éléments.

1. **Déblocage du scroll sur la barre de navigation mobile**

Dans `modules/layout/navigation.css`, la règle de la barre `.bottom-nav` a été ajustée. Les propriétés `touch-action: none;` et `overscroll-behavior: none;` empêchaient le défilement lorsque l'utilisateur passait son doigt sur la navigation. La modification remplace ces valeurs par des paramètres plus permissifs :

```
  /* V491: Empêche le scroll quand on touche la barre de navigation */
-     touch-action: none;
-     overscroll-behavior: none;
+     /* Autoriser le scroll vertical tout en conservant le design fixe */
+     touch-action: pan-y;
+     overscroll-behavior: contain;
```

Intention : permettre au geste vertical de se propager au reste de la page lorsque l'utilisateur glisse sur la barre inférieure, tout en conservant un contrôle sur l'effet de rebond. Cette modification améliore la fluidité du scroll et empêche la sensation de blocage.

Étape 5 – Vérifications mentales

- **Sur mobile (largeur < 768 px)** : la barre de navigation inférieure reste fixe, mais un swipe vertical au niveau de la barre permet désormais de faire défiler la page, ce qui améliore l'ergonomie. Le `padding-bottom` de `#app-view-root` garantit que le contenu ne passe pas sous la barre.
- **Sur la page d'itinéraire** : les conteneurs `.route-input-wrapper` et les champs de recherche possèdent déjà un `position: relative` en media query mobile. Avec la nouvelle règle des dropdowns, les listes d'autocomplétion (`.combobox-dropdown`, `#results-from-suggestions`, etc.) s'ouvriront systématiquement sous les champs, alignées sur leur largeur, sans recouvrir le bas de l'écran.
- **Activation des overlays** : les panneaux en `fixed` qui bloquent le scroll demeurent inchangés. Il faudra s'assurer côté JavaScript qu'ils sont bien cachés (`.hidden` ou absence de classe `.is-active`) pour ne pas verrouiller l'interface.
- **Cohérence visuelle** : aucune refonte majeure de la palette n'a été faite dans cette itération, mais l'usage de variables CSS facilite de futures harmonisations.

Conclusion

Ces premières corrections ciblent deux sources majeures de gêne utilisateur : les popups mal placées et le blocage du scroll sur mobile. Elles sont volontairement minimalistes afin de respecter la structure existante. Pour aller plus loin, il conviendrait de :

- Continuer à remplacer les couleurs codées en dur par les variables définies dans `_config.css`.
- Supprimer graduellement les `!important` inutiles sur les `overflow` / `height` pour améliorer la maintenabilité.
- Harmoniser les valeurs de `border-radius`, d'ombres et d'espacement pour renforcer la cohérence visuelle (par exemple en adoptant systématiquement `--radius-md` et `--shadow-md` pour les cartes et boutons).

Un second audit pourra être réalisé après intégration et tests réels afin de détecter d'autres anomalies éventuelles.
