

Contexte GSB Application mobile de gestion des visites et des médecins

Partie 3 - Gestion des rapports

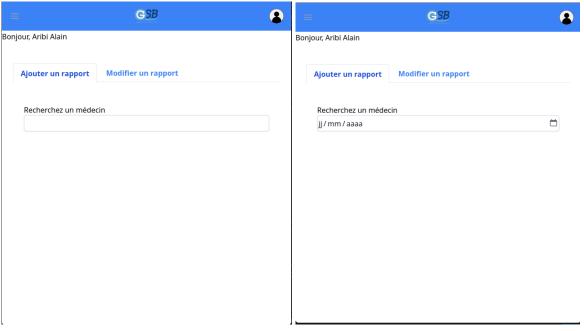
Mise en place de la page des rapports

Nous allons mettre en place la gestion des rapports. Pour cela vous allez mettre en place une page qui se divisera en deux parties :

- Partie "Ajouter un rapport":
 - Pouvoir rechercher un médecin, et le sélectionner
 - Une fois le médecin sélectionné, afficher un formulaire de saisie de rapport,
 - Valider la saisie, qui devra être enregistrée dans la base de données
- Partie "Modifier un rapport" :
 - Rechercher les rapports saisies par l'utilisateur connecté en sélectionnant une date,
 - Afficher la liste des médecins visités à la date choisie,
 - o Pouvoir sélectionner un rapport, et afficher un formulaire pré-rempli avec :
 - Le motif de la visite,
 - Le bilan de la visite
 - A l'issue de cette consultation, on doit pouvoir aussi modifier ce rapport.

Page rapports - Composition:

Pour cette page, nous allons nous inspirer de ce qui a été fait pour la fiche du médecin, à savoir créer un sous-menu, qui va gérer l'affichage selon le choix de l'utilisateur:



Ajouter un rapport

Modifier un rapport

Donc en plus de notre composant Rapports, nous aurons :

- function AjouterRapport(): pour la partie ajout,
- function ModifierRapport(): pour la partie modifier

Travail à faire

Modifier votre composant Rapports avec les modifications demandées, et vérifiez si vous arrivez à bien afficher les différentes parties selon le choix de l'utilisateur.

Ajouter un rapport

Comme indiqué précédemment, ce composant devra permettre au visiteur d'ajouter un nouveau rapport concernant le médecin sélectionné. Attention à bien vérifier les données dont vous aurez besoin afin de réaliser une insertion en bonne et due forme.

Lors que le visiteur le validera :

- Déclenchement d'un appel à l'API via l'URL /ajouterRapport et la métode PUT, avec comme paramètre les données du formulaire (consultez l'API de Axios),
- Si succès de la requête : afficher une bannière de succès,
- Si échec : afficher une bannière d'erreur

Voici une ébauche possible pour ce composant AjouterRapport:

```
export function AjouterRapport({ visiteur }) {
    const [medecin, setMedecin] = useState({}); // État qui contient les données du medecin selectionné
    const [addRapportSuccess, setaddRapportSuccess] = useState(); // Variable qui va verifier si le rapport à bien été ajouté

/**
    * Fonction qui va traîter la soumission du formulaire :
    * Il va récupérer les champs saisies, et faire appel à l'API avec ces données
    * afin d'ajouter le rapport dans la base deonnées
    * @param {any} e
    */
    function ajouteRapport(e) {
        /**
         * A compléter
         */
    }
}

/**

* Appel à l'API pour ajouter le nouveau rapport dans la base de données,
    * via la méthode PUT
    * @param {JSON} params
    * @returns Promesse Axios
    */
    async function ajouterRapportBase(params) {
         /**
         * A compléter
         */
         * A compléter
```

Travail à faire

- Modifier votre composant AjouterRapport avec les ajouts demandées,
- Vérifiez que la mise à jour fonctionne : affichage de la bannière, et surtout au l'insertion au niveau de la base de données :

