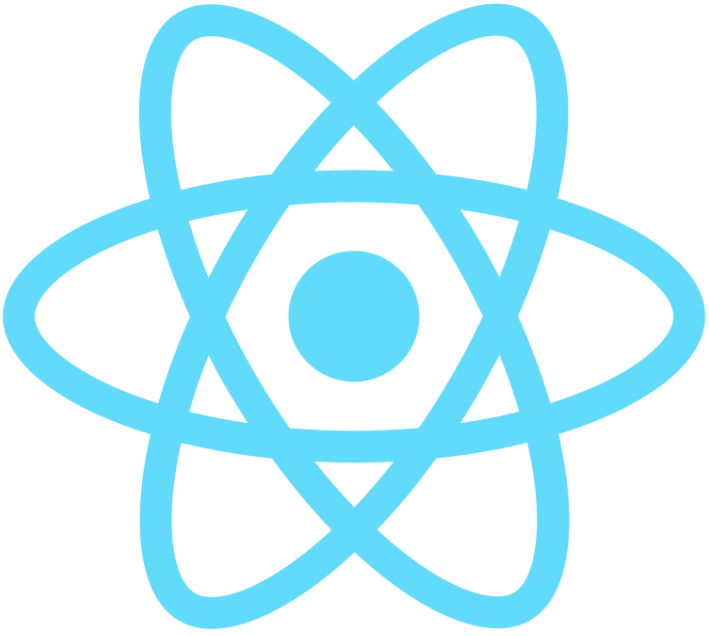
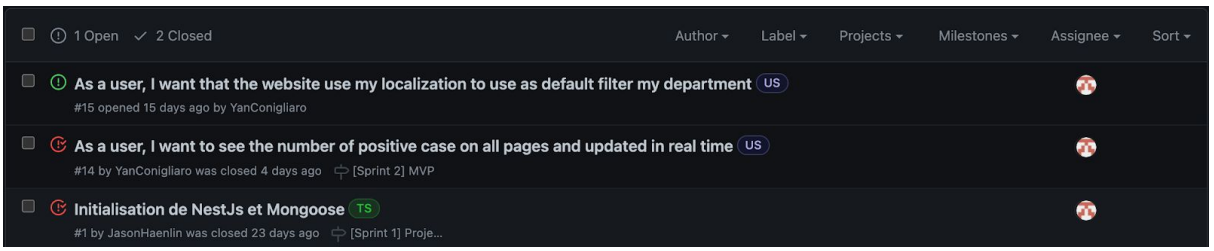
Compte rendu programmation web : Client Side

Ruben Houri



- **Github :** https://github.com/HouriRuben

- **Tâches effectuées :**



- Initialisation de NestJs avec Mongoose pour MongoDB.

- La fonctionnalité qui permet d'afficher le nombre de personnes hospitalisées en live via une API (Refresh Rate : 1 min).

- La fonctionnalité de filtre via la géolocalisation de l'utilisateur.

- Stratégie employée pour la gestion des versions avec Git :

En ce qui concerne ma stratégie git elle est plutôt simple:

1) On crée l'issue sur git
2) On crée une branche sur git pour cette issue
3) On développe sur cette branche , on push comme on veut
4) Une fois l'issue finie , on fait une pull request pour pouvoir merge la branche sur la branche develop.

- Solutions choisies :

1 | Nombre de personnes hospitalisées en live via une API (Refresh Rate : 1 min).

Préciser pourquoi :

J'ai choisi d'utiliser une API du gouvernement :
https://coronavirusapi-france.now.sh/FranceLiveGlobalData

Quelles sont les alternatives ?

Une alternative aurait été de download et update automatiquement nos données puis de créer une query mongo pour sommer le nombre de cas du jour.
Ça aurait été beaucoup plus long.

Difficultés rencontrées :

Durant le développement l'api a été mise à jour ce qui à mis casser la fonctionnalité à cause d'un changement de nom de variable.

Temps de développement :

Un peu plus de **3 heures** le temps de lire la doc NestJs , faire un état de l'art des API et de trouver le bon composant matérial UI.

2 | Filtre via la géolocalisation de l'utilisateur.

Il s'agit de ma tâche la plus complexe. J'ai utilisé **react-hook-geolocation** pour obtenir la longitude et latitude de l'utilisateur.

Ensuite je requette l'API du gouvernement pour obtenir la région associé à cette position.

Avec le nom de la région je peux obtenir le code de la région ce qui me permet de requêter notre back et de filtrer sur se code.

Préciser pourquoi :

Cela permet à l'utilisateur d'obtenir des informations uniquement sur sa région.

Quelles sont les alternatives ?

Nous aurions pu utiliser le service de géolocalisation " `navigator.geolocation`. " mais il nécessite un contexte sécurisé pour son utilisation (HTTPS) ce qui rend les tests en local plus difficiles.

Difficultés rencontrées :

L'utilisation de " `navigator.geolocation` " fût particulièrement pénible car je n'ai pas tout de suite compris cette histoire de contexte sécurisé.

Il m'a fallu un peu chercher pour passer de longitude , latitude à un code région.

Le merge final de cette issue fût un enfer de plusieurs heures sur la partie back en particulier car mes collègues travaillaient en parallèle dessus et certain changement furent particulièrement incompatibles.

Temps de développement :

Plus 15 heures facilement entre les recherches , tous les mécanismes , les dépendances dans le processus de localisation de région et le merge final.

- **Code :**

    - Commenter une fonction ou un composant (max 100 lignes) que vous avez écrit et qui vous semble élégant ou optimal

```
export const Geolocation = ({ setGeoFunction }) => {
  const geolocation = useGeolocation();
  const classes = useStyles();
  const [open, setOpen] = React.useState(true);
  const [currentRegion, setCurrentRegion] = useState(null);
  let pointer;
  let GeoLocPrefUser = localStorage.getItem('GeoLocFilter');

  if (GeoLocPrefUser === 'false') {
    GeoLocPrefUser = false;
  } else if (GeoLocPrefUser === 'true') {
    GeoLocPrefUser = true;
  }

  if (currentRegion) {
    pointer = <RoomIcon />;
  } else {
    pointer = <RoomIcon className={classes.PointerWith} />;
  }
  const getCodeFromRegionName = (nameRegion) => {
    return mapNameRegionToCode[nameRegion];
  };

  useEffect(() => {
    if (geolocation.longitude && geolocation.latitude && GeoLocPrefUser) {
      fetch(`https://api-adresse.data.gouv.fr/reverse/?lon=${geolocation.longitude.toString()}&lat=${geolocation.latitude.toString()}&type=street`)
        .then((response) => response.json()).then((data) => {
          const codeReg = getCodeFromRegionName(data.features[0].properties.context.split(',')[2].trim());
          setCurrentRegion(codeReg);
          setGeoFunction(codeReg);
        });
    }
  }, [geolocation]);
```

```
const Geolocate = () => {
  nbGeolocateClicks += 1;
  if (!currentRegion) {
    if (geolocation.longitude && geolocation.latitude) {
      fetch(`https://api-adresse.data.gouv.fr/reverse/?lon=${geolocation.longitude.toString()}&lat=${geolocation.latitude.toString()}&type=street`)
        .then((response) => response.json()).then((data) => {
          const codeReg = getCodeFromRegionName(data.features[0].properties.context.split(',')[2].trim());
          setCurrentRegion(codeReg);
          setGeoFunction(codeReg);
        });
    }
    localStorage.setItem('GeoLocFilter', true);
  } else {
    localStorage.setItem('GeoLocFilter', false);
    setCurrentRegion(null);
    setGeoFunction(null);
  }
};

const handleClose = () => {
  setOpen(false);
};
const activateGeo = () => {
  Geolocate();
  handleClose();
};

const action = (
  <Button onClick={() => activateGeo()} color="primary" size="small">      You, 18 hours ago • #15 fix pull request
    Oui
  </Button>
);
```

Commentaire :

En cas de clic sur l'icon " Map " , nous appelons la fonction géolocalisation , cette fonction vérifie plusieurs choses notamment si la localisation à déjà été déterminée. Si c'est le cas , elle efface cette localisation pour le composant Géolocalisation et pour le composant parent (HomePage) tout en inscrivant dans le localstorage que l'utilisateur ne souhaite pas être localisé.
Si la localisation n'est pas déterminée elle met dans le localstorage la volonté de l'utilisateur d'être localisé puis test si la géolocalisation a été déterminée. Si ce n'est pas le cas , la fonction finit ici. Mais pas de soucis car la requête s'exécutera dans le useEffect a l'actualisation de l'objet geolocation.

Dans ce composant il y a également:

- La gestion du statut de l'icon pour l'IHM
- La gestion du statut de la navbar avec un button custom
- L'utilisation du thème de l'app

Si le code de la région est trouvé il sera passé au parent via un set passé par une prop.

Se composant rassemble tout la logique jusqu'à l'obtention du code de région ce qui m'a permis de débugger plus facilement tout en englobant les concepts de hooks , communication parent - child via props , fetch , css , localstorage etc...

Il y a évidemment une multitude de codes implémentés en plus de composant côté front et back pour la géolocalisation mais tout commence dans ce composant.

- Commenter une fonction ou un composant (max 100 lignes) que vous avez écrit et qui mériterait une optimisation ou amélioration

```
<FormControl className={classes.formControl}>
  <InputLabel id="reg-label">Regions</InputLabel>
  <Controller
    control={control}
    id="reg"
    name="reg"
    render={(
      {
        name,
      },
    ) => (
      <Select
        name={name}
        value={currentRegion}
        onChange={(e) => handleChangeRegion(e)}
      >
        {codeRegions.map((code) => (
          <MenuItem key={code} value={code}>{mapIdToRegion[code]}</MenuItem>
        ))}
      </Select>
    )}
  />

</FormControl>
```

Je n’ai pas réussi à comprendre comment faire remonter la valeur du Select dans le data du form pour que se soit plus clean pour récupérer la valeur.

```
const onSubmit = (data) => {
  data.since.setHours(0, 0, 0);
  data.to.setHours(0, 0, 0);

  let url;
  if (currentRegion === '00') {
    url = `${process.env.REACT_APP_API_ENDPOINT}/api/v0/incidences?class_age=${data.classAge}&since=${data.since}&to=${data.to}`;
  } else {
    url = `${process.env.REACT_APP_API_ENDPOINT}/api/v0/incidences?class_age=${data.filteredIncidences}&since=${data.since}&to=${data.to}&reg=${currentRegion}`;
  }
  fetch(url)
    .then((response) => response.json())
    .then((filteredIncidences) => {
      setData(filteredIncidences);
    });
};
```

Comme on peut le voir dans le code , je n’utilise pas data.reg mais currentRegion dans le fetch. J’aurais préféré trouver la solution pour faire ça proprement.