UI – écran principal :

- ajouter des icones aux boutons

Flux :

- console : renderTxt pour afficher les messages

- option facultatif :

* ajouter un attribut à event : logical() facultatif
* ajouter une fonction set\_facultatif()
* ajouter une checkbox à treeboutton
* ajouter un obserevent pour qui appelle set\_facultatif
* Modifier next/previous event :
  + Si facultatif est true : prendre l’event n-1 / n+1 (df\_events) ; retirer les events sélectionnés

Sankey :

- colonnes des attributs : lorsque je récupère les attributs, je dois récupérer aussi le type (factor, numerique, date …)

- filtrer les colonnes à afficher

- afficher les colonnes patients

- au lieu de remover, faire un update ; remover seulement ceux qui sont retirer de values

Tabset :

- Améliorer les graphiques

Carto :

- Récupérer l’algo pour avoir la df\_spatial => calculer à partir de df\_selection

- renderLeaflet

- 2 boutons : previous – next

- dénominateur pour les couleurs des polygones ? : une colonne par exemple.

Timelines: le faire pour un patient ; centré sur l’event0

- Patient :

age, sexe, région, département, code géo

La sélection est stocké et passé en paramètre à Event

- Survie :

quand on a l’event décès

=> sélectionner 2 events

datestart du premier event

dateend du dernier = date de dernière nouvelle

Possibiliter d’extraire un tableau pour réaliser des stats.

Parcours aberrants :

- détection via des règles métiers

- possibilité de créer un indicateur sur un pattern d’event => nouvelle colonne à la df patient.

Utilisation de cette nouvelle colonne pour les localiser sur une carte => recherche un paterrn géographique

- Algorithme de clustering : à partir d’un event ou de tous les events ?

- Quelles sont les variables à sélectionner ?

- Chargement des couches et localisation Etab Leaflet : revoir le code