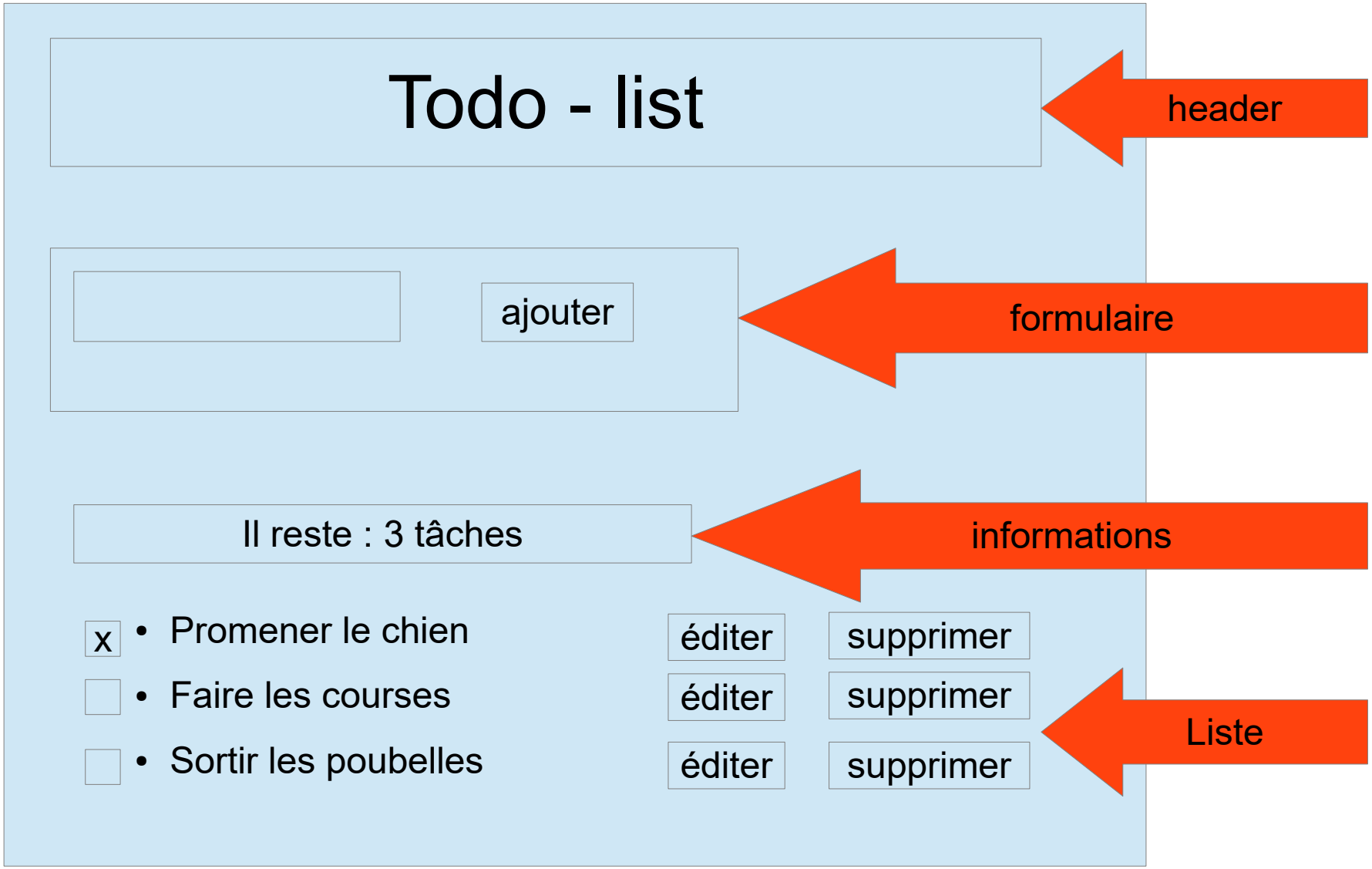
Pour mieux comprendre ce projet vous pouvez vous aider du repo :

Repo Github todo_list

Partie 1
Créer l'interface Utilisateur

1ère étape : on fait une maquette et on implémente le code html/css



2ème étape : on intègre le code HTML dans App.js

- On place le code voulu (*sans le <doctype>)* **dans** le return() de la fonction App()
- On renomme les attributs class en className (*ici c'est du **jsx** et non plus du html*)
- On vérifie que toutes les balises son bien fermées (*attention avec les balises
 ou <input>*)
- On importe les fichiers CSS nécessaires
- On vérifie que tout fonctionne dans le serveur de React : la page doit être identique à ce que nous avions précédemment

3ème étape : on créé un composant qui va remplacer les balises

- On créé le fichier ListComponent.js
- On implémente le composant en mettant dans un return() les éléments que l'on désire (*ici tout ce qui figure dans unr balise *)
- On rajoute export default à la fin pour pouvoir utiliser le comosant ailleurs
- On importe le composant dans App.js
- On place la balise avec le nom du composant (*<ListComponent />*) à l'endroit souhaité

Bilan 1ère partie

- On obtient une liste dont chaque élément est implémenté à l'intérieur d'un composant React que l'on pourra utiliser plusieurs fois : on parle de recycler un composant. Malheureusement ici, le composant affiche toujours la même tâche...
- Il va maintenant falloir injecter des données dans ce composant. Cela va se faire grâce aux **props**. Les props servent à envoyer des données depuis un Composant **parent** (*App*) vers un composant **enfant** (*ListComponent*).

Partie 2
Mise en place des props

1ère étape : mise en place des props dans le composant App

- On place un nouvel attribut dans le composant <*ListComponent*> dans le fichier *App.js*
- Cet attribut portera permettra d'injecter l'énoncé de la tâche dans le composant correspondant.
- On peut appeler cet attribut *name*
- Par exemple :

```
<ListComponent name='Faire les courses' />
```

2ème étape : mise en place des props dans le composant ListComponent

- Il faut déterminer à quel endroit à l'intérieur du composant ListComponent, on veut afficher les données injectées depuis le composant App (composant parent)
- A cet endroit, on représente les données à injecter, par la propriété de l'objet props qui les contient. On écrit donc à cet endroit :

```
|{props.name}
```

- Pour afficher les props dans la console, on peut écrire `console.log(props)` dans la fonction du composant *ListComponent*, avant le *return()*...

3ème étape : envoyer des données comme propriété d'attribut

- Dans le composant *ListComponent*, on rajoute à la balise *<input type= "checkbox"*> un 2ème attribut : `defaultChecked`
- Cet attribut prend une valeur *true* ou *false* et permet de cocher/décocher la checkbox.
- Sur le même modèle que précédemment, on transmet depuis le composant *App* la donnée *true* ou *false* avec l'attribut *checked*

```
|<ListComponent name='...' checked={true} />
```

- On met la valeur à injecter entre des accolades pour que la valeur booléenne soit respectée
- Dans *ListComponent*, il reste à ajouter sa valeur à l'attribut *defautChecked*

```
|defaultChecked={props.checked}
```


GitHub