

## Cours Inf3522 - Développement d'Applications N tiers

### Lab 13 : Styler l'interface frontend avec React MUI

Ce lab explique comment utiliser les composants MUI dans notre interface frontend. Nous utiliserons le composant Button pour afficher des boutons stylisés. Nous utiliserons également des icônes MUI et le composant IconButton. Les champs d'entrée du formulaire modal sont remplacés par des composants TextField.

#### Utilisation du composant Button

Suivez les étapes suivantes pour implémenter le composant Button :

Commençons par remplacer tous les boutons par l'utilisation du composant Button de MUI. Importez le composant Button dans le fichier **AddCar.js** (ligne 7) :

```
JS AddCar.js M X
src > components > JS AddCar.js > ...
1 import React, { useState } from "react";
2 import Dialog from "@mui/material/Dialog";
3 import DialogActions from "@mui/material/DialogActions";
4 import DialogContent from "@mui/material/DialogContent";
5 import DialogTitle from "@mui/material/DialogTitle";
6
7 import Button from '@mui/material/Button';
8
```

Modifiez les boutons pour utiliser le composant Button. Sur la page de la liste, nous utilisons le bouton contenu (**variant="contained"**) et dans le formulaire modal, nous utilisons les boutons de texte (par défaut).

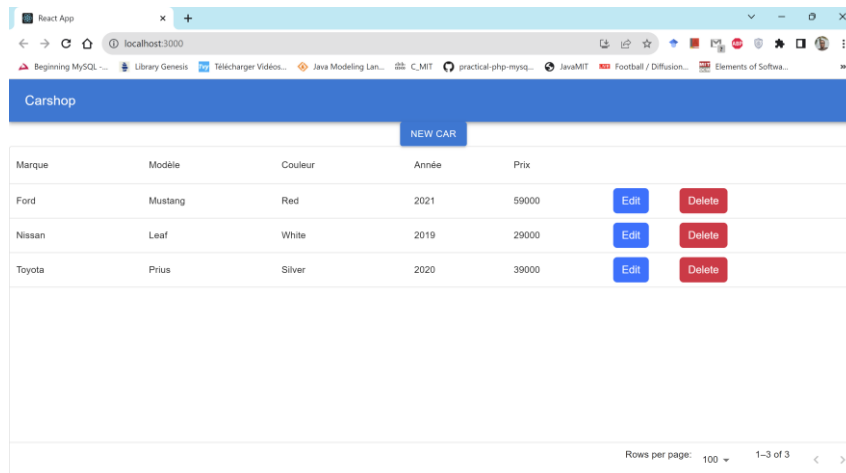
Le code suivant montre l'instruction de retour du composant **AddCar** avec les modifications (Lignes 39-41 et 82-83) :

```
JS AddCar.js M X
src > components > JS AddCar.js > AddCar
31 };
32
33 const handleChange = event => {
34   setCar({ ...car, [event.target.name]: event.target.value });
35 };
36
37 return (
38   <div>
39     <Button variant="contained" onClick={handleClickOpen}>
40       New Car
41     </Button>

```

```
JS AddCar.js M X
src > components > JS AddCar.js > AddCar
66 <div>
67   <input
68     placeholder="Année"
69     name="year"
70     value={car.year}
71     onChange={handleChange}
72   />
73   <br />
74   <input
75     placeholder="Prix"
76     name="price"
77     value={car.price}
78     onChange={handleChange}
79   />
80 </DialogContent>
81 <DialogActions>
82   <Button onClick={handleClose}>Cancel</Button>
83   <Button onClick={handleSave}>Save</Button>
84 </DialogActions>
85 </Dialog>
86 </div>
87 );
88 }
89
90 export default AddCar;
91
```

Maintenant, le bouton "NOUVELLE VOITURE" est également rendu en utilisant le composant MUI Button, mais la mise en page ne semble pas bonne car le bouton touche la barre d'outils.



Nous ajouterons des marges au bouton en utilisant le composant de mise en page MUI Stack. Importez le composant MUI Stack dans le fichier **Carlist.js** :

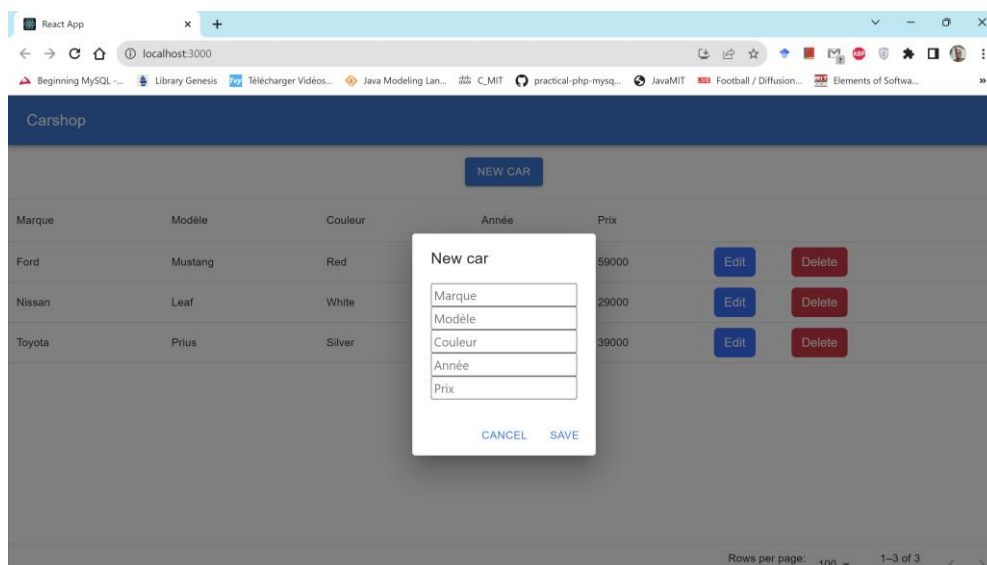
```
import Stack from '@mui/material/Stack';
```

4. Ensuite, enveloppez le composant **AddCar** à l'intérieur du composant Stack et ajoutez des marges supérieures et inférieures en utilisant les propriétés mt et mb (104-107) :

```
101
102   return (
103     <React.Fragment>
104       <Stack mt={2} mb={2}>
105         <AddCar addCar={addCar} />
106       </Stack>
107     <div style={{ height: 500, width: "100%" }}>
108
```

Maintenant, le bouton "NOUVELLE VOITURE" devrait ressembler à ce qui est montré dans la capture d'écran suivante :

Les boutons du formulaire modal devraient ressembler à ceci :



Il faut également modifier l'apparence des boutons save et cancel du composant dans EditCar.js.

### Utilisation des composants d'icônes

MUI propose des icônes SVG préconstruites que nous devons installer en utilisant la commande suivante dans le terminal :

**npm install @mui/icons-material**

Commençons par implémenter le bouton "SUPPRIMER" dans la grille. Le composant MUI IconButton peut être utilisé pour afficher des boutons d'icônes. Le package **@mui/icons-material** que nous venons d'installer contient de nombreuses icônes qui peuvent être utilisées avec MUI.

Vous pouvez trouver la liste des icônes disponibles dans la documentation MUI. Nous avons besoin d'une icône pour notre bouton "SUPPRIMER", nous utiliserons donc une icône appelée DeleteIcon :

1. Ouvrez le fichier **CarList.js** et ajoutez les imports suivants :

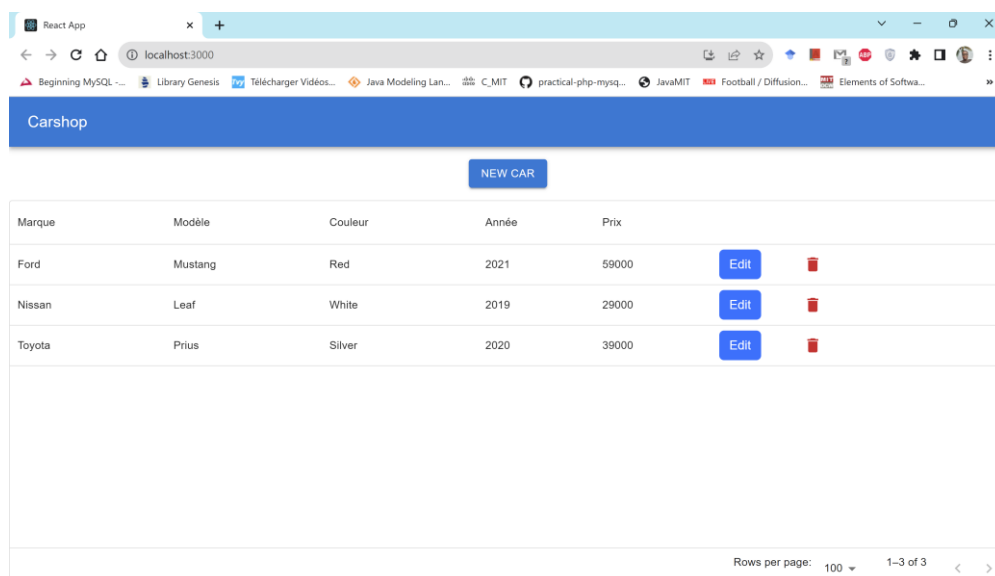
**import IconButton from '@mui/material/IconButton';**

**import DeleteIcon from '@mui/icons-material/Delete';**

2. Ensuite, nous allons rendre le composant **IconButton** dans notre grille. Nous modifierons le bouton "SUPPRIMER" dans le code où nous définissons les colonnes de la grille. Nous remplaçons l'élément du bouton par le composant **IconButton** et nous rendons **DeleteIcon** à l'intérieur du composant **IconButton**. Pour obtenir l'icône "supprimer" en rouge, nous pouvons utiliser la propriété **color** de **DeleteIcon** (Ligne 98-100) :

```
JS CarList.js M X
src > components > JS CarList.js > ...
93   field: "_links.self.href",
94   headerName: "",
95   sortable: false,
96   filterable: false,
97   renderCell: row => (
98     <IconButton onClick={() => onDeleteClick(row.id)}>
99       <DeleteIcon color="error" />
100     </IconButton>
101   ),
102 },
103 ];
104
```

Maintenant, le bouton "SUPPRIMER" dans la grille devrait ressembler à la capture d'écran suivante :



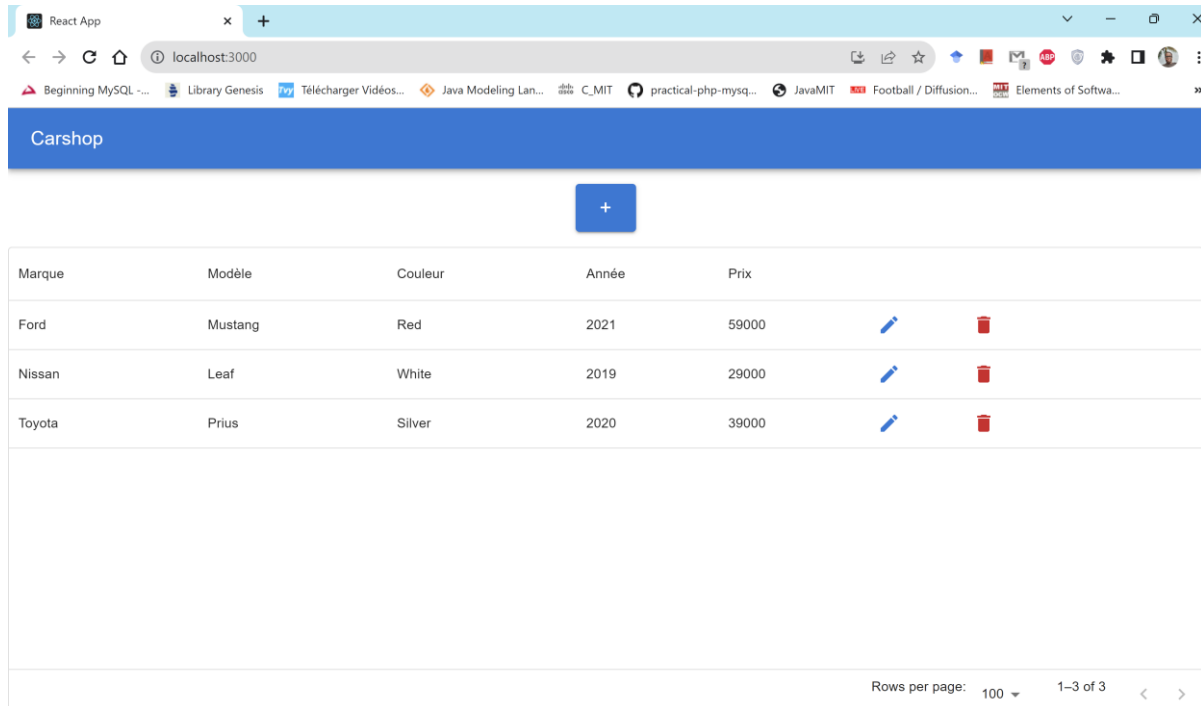
Ensuite, vous allez implémenter le bouton "Éditer" en utilisant le composant **IconButton**.

Vous allez importer pour ce faire dans **EditCar.js** les composants ci-dessous, puis terminer pour obtenir la capture suivante :

```
import IconButton from '@mui/material/IconButton';
```

```
import EditIcon from '@mui/icons-material/Edit';
```

Finalement, les deux boutons sont rendus sous forme d'icônes, comme le montre la capture d'écran suivante :



Ensuite, nous implémenterons des champs de texte en utilisant le composant **TextField** de MUI.

### Utilisation des composants **TextField**

Dans cette section, nous allons changer les champs de texte dans les formulaires modaux en utilisant les composants **TextField** et **Stack** de MUI :

1. Ajoutez la déclaration d'importation suivante dans les fichiers **AddCar.js** et **EditCar.js** :

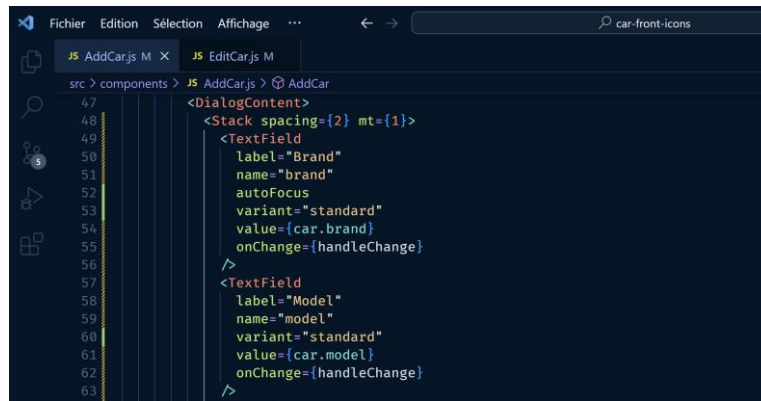
```
import TextField from '@mui/material/TextField';
```

```
import Stack from '@mui/material/Stack';
```

2. Ensuite, remplacez les éléments de saisie par les composants **TextField** dans les formulaires d'ajout et de modification. Nous utilisons la propriété **label** pour définir les libellés des composants **TextField**. Le premier composant **TextField** contient la propriété **autoFocus**, et la saisie sera focalisée sur ce champ. Il existe trois variantes différentes disponibles, et nous utilisons la variante standard.

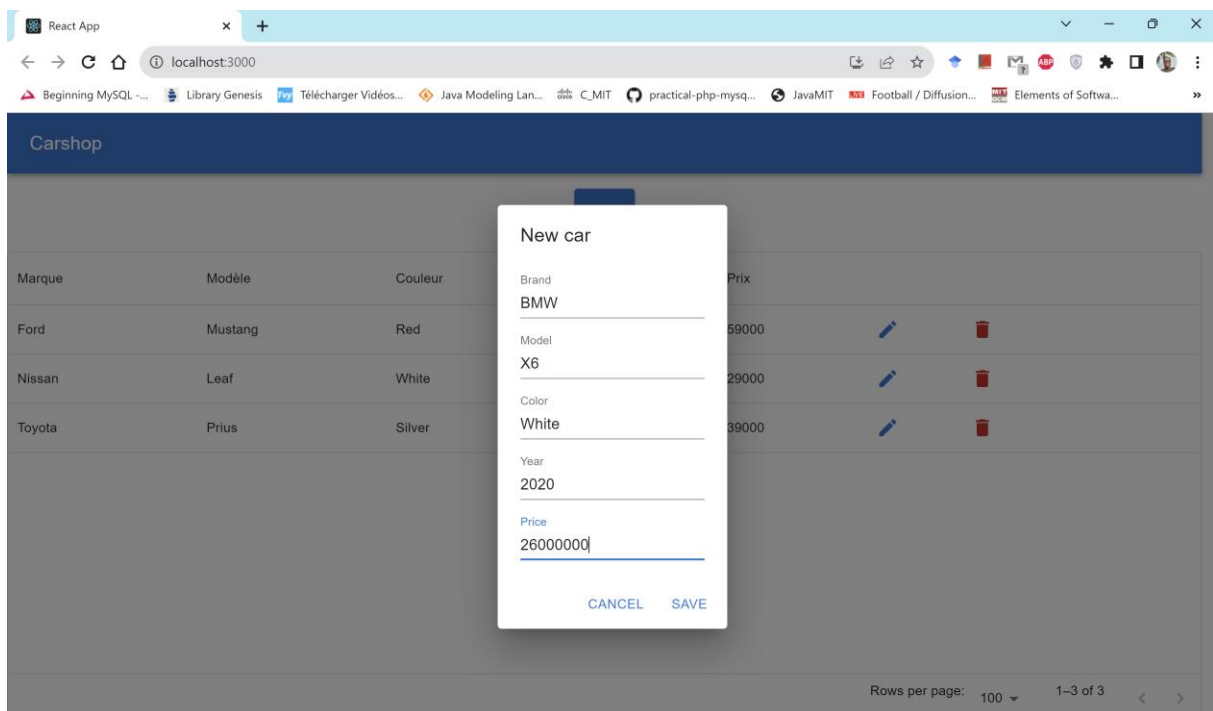
Les champs de texte sont enveloppés à l'intérieur du composant **Stack** pour obtenir un espacement entre les composants et pour définir la marge supérieure :

Les modifications sont à faire dans **AddCar.js** ainsi que dans **EditCar.js**.



```
47 <DialogContent>
48
49   <Stack spacing={2} mt={1}>
50     <TextField
51       label="Brand"
52       name="brand"
53       autoFocus
54       variant="standard"
55       value={car.brand}
56       onChange={handleChange}
57     />
58     <TextField
59       label="Model"
60       name="model"
61       variant="standard"
62       value={car.model}
63       onChange={handleChange}
64     />
59
```

Mettre les autres TextField manquants (nous avons capturé que la marque et le modèle).



Maintenant, nous avons terminé de styliser notre interface utilisateur en utilisant les composants MUI.

## Résumé

Dans ce lab, nous avons finalisé notre interface utilisateur en utilisant MUI, la bibliothèque de composants React qui implémente le design matériel de Google. Nous avons remplacé les boutons par les composants MUI Button et IconButton. Notre formulaire modal a obtenu un nouveau look en utilisant le composant MUI TextField. Après ces modifications, notre interface utilisateur a une apparence plus professionnelle et homogène.