

# NOTE DE STAGE 2021

## CHEZ COFLINE

### SOMMAIRE

Lundi 7 Juin et Mardi 8 Juin 2021.....	2
Missions:.....	2
Réalisation: .....	2
Mercredi 9 Juin 2021 .....	7
Missions:.....	7
Réalisations: .....	7
Jeudi 10 juin 2021 .....	10
Missions:.....	10
Réalisations: .....	10
Vendredi 11 juin 2021 .....	15
Missions:.....	15
Réalisations: .....	15

LUNDI 7 JUIN ET MARDI 8 JUIN 2021

## MISSIONS:

1. Continuer les activités sur l'application *ReactJS* et son apprentissage.

## RÉALISATIONS:

- 1.

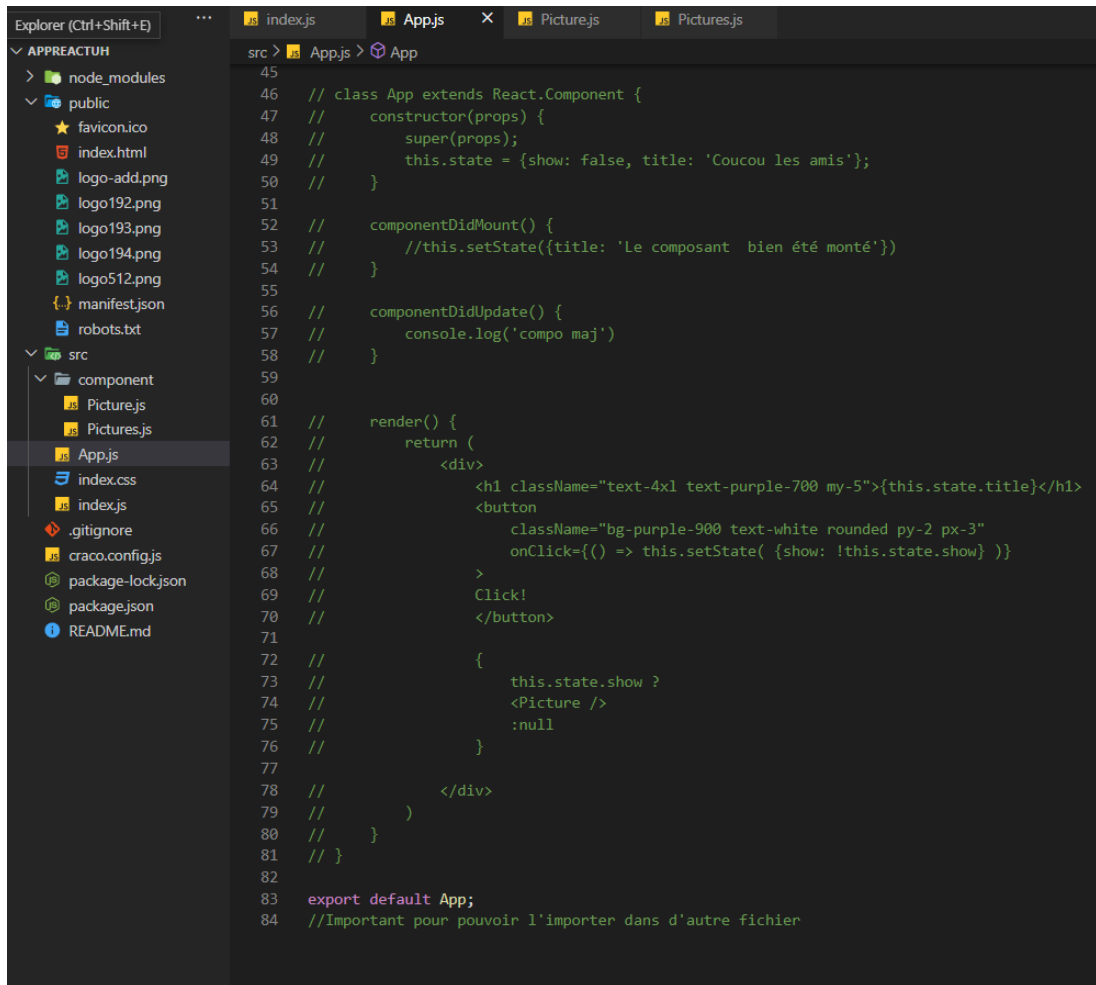
Suite à une mauvaise compréhension j'ai dû reprendre ce que j'avais fait. En effet nous allons créer une application *ReactJS* qui sera reliée à notre projet Laravel par **api** dans le futur. J'ai donc repris de zéro et suivi un tutoriel **Nord Coders**<sup>1</sup> sur la création d'une application *React*. Voici le résultat de ce tutoriel :

```

src > App.js > App
1  import React, {useState, useEffect, useRef} from 'react';
2  import Picture from './component/Pictures';
3
4  function App() {
5
6      const [title, setTitle] = useState('Coucou les amis');
7      //Hook qui est utile pour faire la même chose
8      //que componentDidMount etc...
9      const [show, setShow] = useState(false); //Hook
10     const isShowInitialize = useRef(false);
11     //Pour enregistrer les valeur du DOM
12     //Pour sauvegarder des noeuds du DOM de base
13
14     useEffect(() => console.log('composant monté'), []);
15     // S'utilise à la place des ComponentDidMount...
16     //En application fonctionnel
17     useEffect(() => {
18         if (isShowInitialize.current) {
19             console.log('Show maj')
20         } else {
21             isShowInitialize.current = true;
22         }
23     }, [show]);
24
25     function handleClick() {
26         setShow(!show);
27     }
28
29     return (
30         <div>
31             {console.log('JSX Render')}
32             <h1 className="text-4xl text-purple-700 my-5">{title}</h1>
33             <button
34                 className="bg-purple-900 text-white rounded py-2 px-3"
35                 onClick={handleClick}
36             >
37                 Click!
38             </button>
39
40             { show ? <Picture /> : null }
41
42         </div>
43     )
44 }
45

```

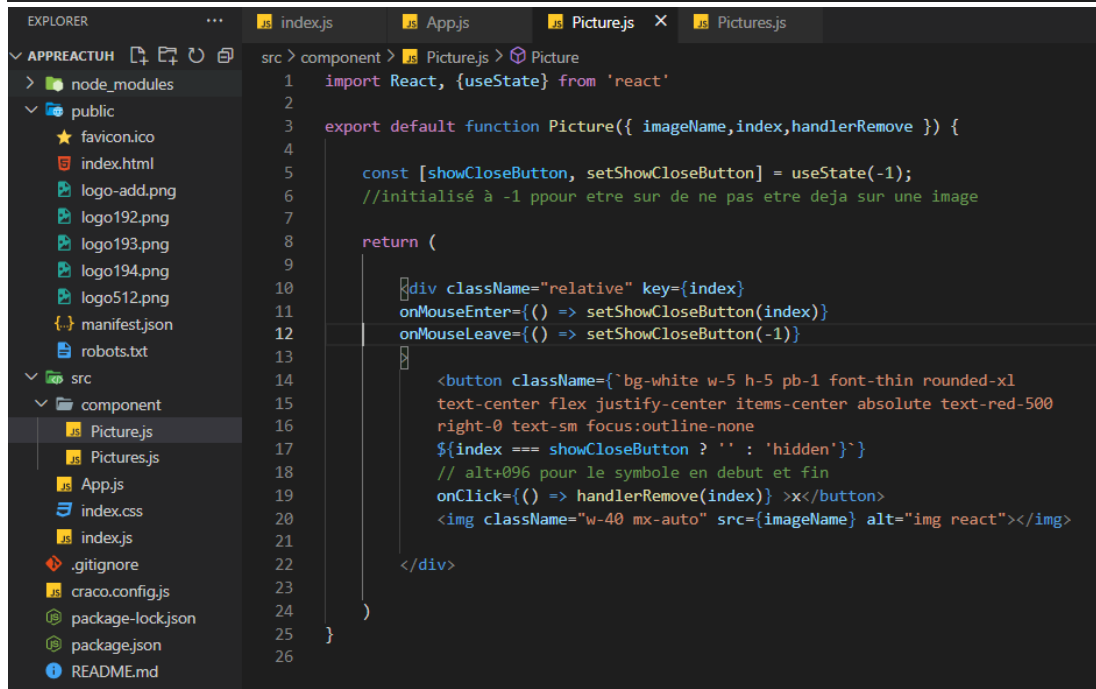
<sup>1</sup> [https://www.youtube.com/watch?v=zNEck1k3\\_zY&list=PLeeuWNW2FHVjVHC8LTbqAvGe9I23sl0Bj](https://www.youtube.com/watch?v=zNEck1k3_zY&list=PLeeuWNW2FHVjVHC8LTbqAvGe9I23sl0Bj)



```

src > Appjs > App
45
46 // class App extends React.Component {
47 //   constructor(props) {
48 //     super(props);
49 //     this.state = {show: false, title: 'Coucou les amis'};
50 //   }
51
52 //   componentDidMount() {
53 //     //this.setState({title: 'Le composant bien été monté'})
54 //   }
55
56 //   componentDidUpdate() {
57 //     console.log('compo maj')
58 //   }
59
60
61 //   render() {
62 //     return (
63 //       <div>
64 //         <h1 className="text-4xl text-purple-700 my-5">{this.state.title}</h1>
65 //         <button
66 //           className="bg-purple-900 text-white rounded py-2 px-3"
67 //           onClick={() => this.setState( {show: !this.state.show} )}
68 //         >
69 //           Click!
70 //         </button>
71 //
72 //         {
73 //           this.state.show ?
74 //           <Picture />
75 //           :null
76 //         }
77 //       </div>
78 //     )
79 //   }
80 // }
81 // }
82
83 export default App;
84 //Important pour pouvoir l'importer dans d'autre fichier

```



```

src > component > Picturejs > Picture
1  import React, {useState} from 'react'
2
3  export default function Picture({ imageName,index,handlerRemove }) {
4
5      const [showCloseButton, setShowCloseButton] = useState(-1);
6      //initialisé à -1 ppour etre sur de ne pas etre deja sur une image
7
8      return (
9
10         <div className="relative" key={index}
11           onMouseEnter={() => setShowCloseButton(index)}
12           onMouseLeave={() => setShowCloseButton(-1)}
13
14         >
15
16             <button className={`bg-white w-5 h-5 pb-1 font-thin rounded-xl
17               text-center flex justify-center items-center absolute text-red-500
18               right-0 text-sm focus:outline-none
19               ${index === showCloseButton ? '' : 'hidden'}>
20               // alt+096 pour le symbole en debut et fin
21               onClick={() => handlerRemove(index)} >x</button>
22             <img className="w-40 mx-auto" src={imageName} alt="img react"></img>
23
24         </div>
25
26     )
27 }

```

```

src > component > Pictures.js > Pictures
1  import React, { useState, useEffect, useRef } from 'react';
2  import Picture from './Picture';
3
4  export default function Pictures() {
5
6      //const [myTimer, setMyTimer] = useState(null); //Hook
7      // const images = [
8      //   'logo192.png',
9      //   'logo193.png',
10     'logo194.png',
11     'logo512.png'
12   ];
13
14   const [images, setImages] = useState([
15     'logo192.png',
16     'logo193.png',
17     'logo194.png',
18     'logo512.png'
19   ]);
20
21   const [image, setImage] = useState(null);
22   //pour éviter d'écraser le tableau
23
24   function ImagesComponent() {
25     return images.map((name, index) => <Picture imageName={name} index={index} handlerRemove={handlerRemoveImage}/> );
26   } //composant intégré à un autre composant
27
28   function handlerRemoveImage(index) {
29     setImages(images.filter((image, i) => i !== index));
30   }
31
32   function handleImageName(event) {
33     //console.log(event.target.value);
34     setImage(event.target.value);
35   }
36
37   function addImageName() {
38     let newImages = [ ... images, image];
39     setImages(newImages);
40   }
41 }
42
43 // useEffect(() => {
44 //   const myTimer = setInterval(() => {
45 //     //console.log('timer appelé');
46 //   }, 1000); //Timer en seconde
47 //   return () => clearInterval(myTimer); //Remet le timer à 0
48 // }, []);
49
50 // const inputToFocus = useRef(null);
51
52 // useEffect(() => {
53 //   inputToFocus.current.focus();
54 //   // curseur directement sur l'endroit à compléter
55 // })
56
57 // const fruitRef = useRef(null);
58 // const [fruitState, setFruitState] = useState(null);
59
60 return (
61   <div className="container mx-auto">
62     <div className="flex items-center justify-between">
63       <ImagesComponent /> {/* self-closing tag */}
64     </div>
65     <div>
66       <button onClick={() => {
67         setFruitState('orange');
68         fruitRef.current = 'pomme'
69       }}> Cliquez ici pour générer un fruit</button>
70     </div>
71     <div>
72       {fruitState}
73       {fruitRef.current}
74     </div>
75     <div className="mt-5">
76       <input type="text" className="border border-gray-600 shadow rounded p-3 mr-2 outline-none"
77         onChange={handleImageName} />
78       <div>
79         <input type="submit" className="bg-purple-400 text-white rounded p-3"
80           onClick={addImageName}> Inscrire le nom d'une image</button>
81       </div>
82     </div>
83   </div>
84 )
85 }
86

```

```

86
87
88
89 // export default class Picture extends Component {
90
91 //   constructor(props) {
92 //     super(props);
93 //     this.state = {timer: null};
94 //   }
95
96 //   componentDidMount() {
97 //     this.setState({timer:
98 //       setInterval(() => {
99 //         console.log('Timer appelé')
100 //       }, 1000)
101 //     });
102 //   }
103
104 //   componentWillUnmount() {
105 //     clearInterval(this.state.timer);
106 //   }
107
108 //   render() {
109 //     return (
110 //       </img>
111 //     )
112 //   }
113 // }
114

```

```

1 /* ./src/index.css */
2 @tailwind base;
3 @tailwind components;
4 @tailwind utilities;

```

J'en ai profité pour réinstaller TailwindCSS sur cette application.

Le code commenté sur les captures d'écran fait partie du tutoriel pas à pas que j'ai suivi. Cela m'a permis de m'exercer et de vraiment comprendre la logique du ReactJS.

Sur la page suivante se trouve le rendu visuel.



# Coucou les amis

Click!



# Coucou les amis

Click!

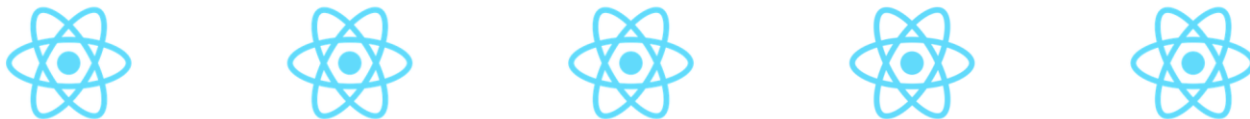


Inscrire le nom d'une image



# Coucou les amis

Click!



Inscrire le nom d'une image



# Coucou les amis

Click!



Inscrire le nom d'une image



# Coucou les amis

Click!



Inscrire le nom d'une image

MERCREDI 9 JUIN 2021

## MISSIONS:

1. Continuer les activités sur l'application *ReactJS*.

## RÉALISATIONS:

- 1.

J'ai essayé d'adapter ce que j'ai vu les jours précédents à notre application. Cependant cela s'est révélé infructueux malgré plusieurs essais. Voici l'avancé du code :

```

EXPLORER
src > App.js > App
1  import React, {useState, useEffect, useRef} from 'react';
2  import Picture from './component/Pictures';
3  import Addresses from './component/Addresses';
4
5  function App() {
6
7      const [title, setTitle] = useState('Coucou les amis');
8      //Hook qui est utile pour faire la même chose
9      //que componentDidMount etc...
10     const [show, setShow] = useState(false); //Hook
11     const isShowInitialize = useRef(false);
12     //Pour enregistrer les valeur du DOM
13     //Pour sauvegarder des noeuds du DOM de base
14
15     const [showAd, setShowAd] = useState(false); //Hook
16     const isShowInitAd = useRef(false);
17
18     //useEffect(() => console.log('composant monté'), []);
19     // S'utilise à la place des ComponentDidMount...
20     //En application fonctionnel
21     useEffect(() => {
22         if (isShowInitialize.current) {
23             console.log('Show maj')
24         } else {
25             isShowInitialize.current = true;
26         }
27     }, [show]);
28     useEffect(() => {
29         if (isShowInitAd.current) {
30             console.log('Show maj')
31         } else {
32             isShowInitAd.current = true;
33         }
34     }, [showAd]);

```

The image consists of two screenshots of a VS Code editor showing a React application. The top screenshot shows the `App.js` file, and the bottom screenshot shows the `Address.js` file.

**Top Screenshot (App.js):**

```

35
36     function handleClick() {
37         setShow(!show);
38     }
39     function handleClickAddresses() {
40         setShowAd(!showAd);
41     }
42
43     return (
44         <div>
45             <div>
46                 {/* {console.log('JSX Render')} */}
47                 <h1 className="text-4xl text-purple-700 my-5">{title}</h1>
48                 <button
49                     className="bg-purple-900 text-white rounded py-2 px-3"
50                     onClick={handleClick}>
51                     >
52                     Click!
53                 </button>
54
55                 { show ? <Picture /> :null }
56             </div>
57             <div>
58                 <button
59                     className="bg-purple-900 text-white rounded py-2 px-3"
60                     onClick={handleClickAddresses}>
61                     >
62                     Addresses
63                 </button>
64
65                 { showAd ? <Addresses /> :null }
66             </div>
67         </div>
68     )
69 }
70

```

**Bottom Screenshot (Address.js):**

```

1  import React, {useState} from 'react';
2  import Address from './Address';
3
4  export default function Addresses() {
5
6      const [address1, setAddress1] = useState('2 Rue Example ');
7      const [address2, setAddress2] = useState('Résidence 1 Tréed');
8      const [city, setCity] = useState('Toulon ');
9      const [zipCode, setZipCode] = useState('83000');
10     const [index] = useState(-1);
11
12     const [addresses, setAddresses] = useState([
13
14     ]);
15
16     function AddressComponent() {
17         return addresses.map((address1,address2,city,zipCode) => <Address a1={address1} a2={address2} c={city} zc={zipCode} />);
18     }
19
20     // function handlerRemoveAddress(index) {
21     //     setAddresses(addresses.filter((address, i) => i !== index));
22     // }
23
24     function handleAd1(event) {
25         setAddress1(event.target.value);
26     }
27     function handleAd2(event) {
28         setAddress2(event.target.value);
29     }
30     function handleCity(event) {
31         setCity(event.target.value);
32     }
33     function handleZC(event) {
34         setZipCode(event.target.value);
35     }
36
37     function handleNewAddress() {
38         const addresses = any[]
39         let newAddress = [...addresses, <Address a1={address1} a2={address2} c={city} zc={zipCode}/> ] // index={index} handlerRemove={handlerRemoveAddress}
40         setAddresses(newAddress);
41     }
42

```



```

EXPLORER
  APPREACTUW
    node_modules
    public
      favicon.ico
      index.html
      logo-add.png
      logo192.png
      logo193.png
      logo194.png
      logo512.png
      manifest.json
      robots.txt
    src
      component
        Address.js
        Address.js
        Picture.js
        Picture.js
        App.js
        index.css
        index.js
        .gitignore
        craco.config.js
        package-lock.json
        package.json
        README.md

src > component > Address.js
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78

return (
  <div className="container mx-auto">
    <div className="flex items-center justify-between">
      <AddressComponent />
    </div>
    <div className="mt-5">
      <form>
        <label for="ad">Address 1:</label>
        <input type="text" id="ad"
          className="border border-gray-600 shadow rounded p-3 mr-2 outline-none"
          onChange={handleAd1}
        />
        <label for="ad2">Address 2:</label>
        <input type="text" id="ad2"
          className="border border-gray-600 shadow rounded p-3 mr-2 outline-none"
          onChange={handleAd2}
        />
        <label for="city">City:</label>
        <input type="text" id="city"
          className="border border-gray-600 shadow rounded p-3 mr-2 outline-none"
          onChange={handleCity}
        />
        <label for="zipcode">Zip code :</label>
        <input type="text" id="zipcode"
          className="border border-gray-600 shadow rounded p-3 mr-2 outline-none"
          onChange={handleZC}
        />
        <button type="submit" className="bg-purple-400 text-white rounded p-3"
          onClick={handleNewAddress}>Add an Address</button>
      </form>
    </div>
  </div>
)

```

```

EXPLORER
  APPREACTUW
    node_modules
    public
      favicon.ico
      index.html
      logo-add.png
      logo192.png
      logo193.png
      logo194.png
      logo512.png
      manifest.json
      robots.txt
    src
      component
        Address.js
        Address.js
        Picture.js
        Picture.js
        App.js
        index.css
        index.js
        .gitignore
        craco.config.js
        package-lock.json
        package.json
        README.md

src > component > Address.js
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46

import React, {useState} from 'react'
export default function Address(a1,a2,c,zc) {
  // const [showCloseButton, setShowCloseButton] = useState(-1);

  return (
    <div className="relative">
      { /* key={index} onMouseEnter={() => setShowCloseButton(index)} onMouseLeave={() => setShowCloseButton(-1)} */ }
      <div>
        { /* <button className="bg-white w-5 h-5 pb-1 font-thin rounded-xl
          text-center flex justify-center items-center absolute text-red-500
          right-0 text-sm focus:outline-none
          ${index === showCloseButton ? '' : 'hidden'}"
          onClick={() => handlerRemove(index)} >x</button> */ }
        <h2>test</h2>
        <table>
          <tr>
            <th>Names</th>
            <th>Values</th>
          </tr>
          <tr>
            <td>Address :</td>
            <td>{a1}</td>
          </tr>
          <tr>
            <td>Address 2 :</td>
            <td>{a2}</td>
          </tr>
          <tr>
            <td>City :</td>
            <td>{c}</td>
          </tr>
          <tr>
            <td>Zip code :</td>
            <td>{zc}</td>
          </tr>
        </table>
      </div>
    </div>
  )
}

```

JEUDI 10 JUIN 2021

## MISSIONS:

1. Continuer les activités sur l'application *ReactJS*. En réfléchissant à une solution temporaire avec du **JSON** pour faire évoluer l'application.

## REALISATIONS:

1.
  - a. Méthode avec les composants React par **fonction**.

Je n'ai toujours pas réussi à faire fonctionner cette partie de code rien ne s'affiche lorsque je clique sur le bouton Adresses dû à des erreurs provoquées par ma mauvaise manipulation des tableaux. Je cherche encore comment gérer cela, et faire persister les données au moins le temps du fonctionnement de l'application. De plus j'ai trouvé d'autre tutoriel se servant de packages supplémentaires, je ne les ai pas encore assez exploités (dont un avec Axios qu'on utilisera plus tard lors de la liaison React/Laravel)<sup>2345</sup>.

```

src > App.js > ...
1 import React, {Component,useState, useEffect, useRef} from 'react';
2 import '../node_modules/bootstrap/dist/css/bootstrap.min.css';
3 import Picture from './component/Pictures';
4 import Addresses from './component/Addresses';
5
6
7 //Essaie 1ère Méthode rfc
8
9 function App() {
10
11   const [title, setTitle] = useState('Coucou les amis');
12   //Hook qui est utile pour faire la même chose
13   //que componentDidMount etc...
14   const [show, setShow] = useState(false); //Hook
15   const isShowInitialize = useRef(false);
16   //Pour enregistrer les valeur du DOM
17   //Pour sauvegarder des noeuds du DOM de base
18
19   const [showAd, setShowAd] = useState(false); //Hook
20   const isShowInitAd = useRef(false);
21
22   //useEffect(() => console.log('composant monté'), []);
23   // S'utilise à la place des componentDidMount...
24   //En application fonctionnel
25   useEffect(() => {
26     if (isShowInitialize.current) {
27       console.log('Show maj')
28     } else {
29       isShowInitialize.current = true;
30     }
31   }, [show]);
32   useEffect(() => {
33     if (isShowInitAd.current) {
34       console.log('Show maj')
35     } else {
36       isShowInitAd.current = true;
37     }
38   }, [showAd]);
39
40   function handleClick() {
41     setShow(!show);
42   }
43   function handleClickAddresses() {
44     setShowAd(!showAd);
45   }

```

<sup>2</sup> <https://www.valentinog.com/blog/fake/> (non exploité, semble trop avancé pour notre besoin)

<sup>3</sup> <https://www.pluralsight.com/guides/react-mock-api> (essayé mais une commande ne semble pas fonctionner)

<sup>4</sup> <https://www.verypossible.com/insights/> (non exploité, car compliqué à comprendre mais intéressant)

<sup>5</sup> <https://blog.harveydelaney.com/setting-up-a-mock-api-for-your-front-end-react-project/>

```

EXPLORER
  APPREACTU
    json-mock-api
    node_modules
    src
      db.json
      package-lock.json
      package.json
    node_modules
    public
      favicon.ico
      index.html
      logo-add.png
      logo192.png
      logo193.png
      logo194.png
      logo512.png
      manifest.json
      robots.txt
    src
      component
        Address.js
        Address.js
        Address.json
        Pictures.js
        Pictures.js
        App.js
        index.css
        index.js
        .gitignore
        craco.config.js
        package-lock.json
        package.json
        README.md

src > component > Address.js > ...
1  import React, {Component,useState} from 'react';
2  import Address from './Address';
3
4  //Essaie 1ère Méthode en rfc
5
6  export default function Addresses() {
7
8    const [address1, setAddress1] = useState("");
9    const [address2, setAddress2] = useState("");
10   const [city, setCity] = useState("");
11   const [zipCode, setZipCode] = useState("");
12
13   const [data,setData] = useState({});
14
15   const [addresses, setAddresses] = useState([
16     {
17       "2Rue exemple",
18       "residence",
19       "toulon",
20       "83000"
21     },
22     {
23       "2Rpjcc fcfscfe",
24       "residencdsdcse",
25       "la garde",
26       "83130"
27     }
28   ]);
29
30   function AddressComponent() {
31     return addresses.map((item,index) => <Address item={item} index={index} handlerRemove={handlerRemoveAddress}/>);
32   }
33
34   function handlerRemoveAddress(index) {
35     setAddresses(addresses.filter((address, i) => i !== index));
36   }
37
38   function handleNewAddress() {
39     setData([address1,address2,city,zipCode]);
40     let newAddress =[...addresses, data ] // index={index} handlerRemove={handlerRemoveAddress}
41     setAddresses(newAddress);
42   }
43
44 }
45

```

```

EXPLORER
  APPREACTU
    json-mock-api
    node_modules
    src
      db.json
      package-lock.json
      package.json
    node_modules
    public
      favicon.ico
      index.html
      logo-add.png
      logo192.png
      logo193.png
      logo194.png
      logo512.png
      manifest.json
      robots.txt
    src
      component
        Address.js
        Address.js
        Address.json
        Pictures.js
        Pictures.js
        App.js
        index.css
        index.js
        .gitignore
        craco.config.js
        package-lock.json
        package.json
        README.md

src > component > Address.js > ...
45
46   return (
47     <div className = "container mx-auto">
48       <div className="flex items-center justify-between">
49         <AddressComponent />
50       </div>
51       <div className="mt-5">
52         <form>
53           <label for="ad">Address 1:</label>
54           <input type="text" id="ad"
55             className="border border-gray-600 shadow rounded p-3 mr-2 outline-none"
56             onChange={(event) => {setAddress1(event.target.value)}}
57           />
58           <label for="ad2">Address 2:</label>
59           <input type="text" id="ad2"
60             className="border border-gray-600 shadow rounded p-3 mr-2 outline-none"
61             onChange={(event) => {setAddress2(event.target.value)}}
62           />
63           <label for="city">City:</label>
64           <input type="text" id="city"
65             className="border border-gray-600 shadow rounded p-3 mr-2 outline-none"
66             onChange={(event) => {setCity(event.target.value)}}
67           />
68           <label for="zipcode">Zip code :</label>
69           <input type="text" id="zipcode"
70             className="border border-gray-600 shadow rounded p-3 mr-2 outline-none"
71             onChange={(event) => {setZipCode(event.target.value)}}
72           />
73           <button type="submit" className="bg-purple-400 text-white rounded p-3"
74             onClick={handleNewAddress}>Add an Address</button>
75         </form>
76       </div>
77     </div>
78   )
79
80 }
81

```

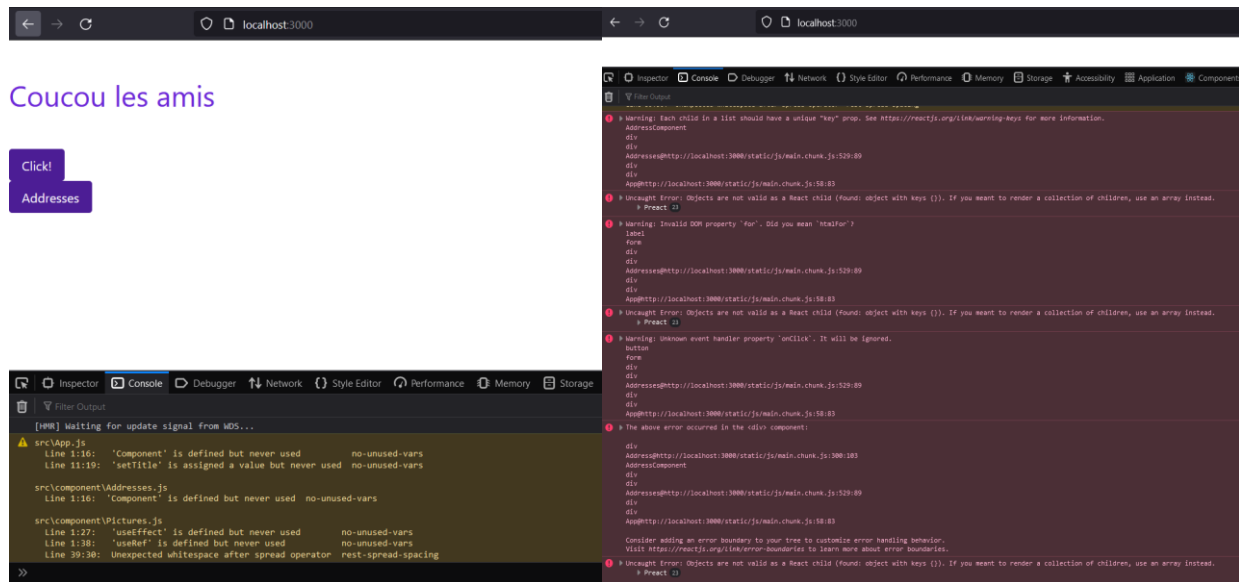
```
EXPLORER    ...    Addresses.js    Address.js X    App.js

> APPRECIATUH    src > component > Address.js > Address
  json-mock-api    1 import React, {useState} from 'react';
  node_modules    2
  > db.json        3
  {}              4 //Essaie 1ère Méthode rfc
  package-lock.json 5
  package.json    6 export default function Address(item,index,handlerRemove) {
  > node_modules  7
  public          8   const [showCloseButton, setShowCloseButton] = useState(-1);
  favicon.ico     9
  index.html      10   return (
  logo-add.png    11     <div className="relative" >
  logo192.png     12       key={index} onMouseEnter={() => setShowCloseButton(index)} onMouseLeave={() => setShowCloseButton(-1)}
  logo193.png     13
  logo194.png     14
  logo512.png     15
  manifest.json   16
  robots.txt      17
  > src           18
  > component     19
  Address.js      20
  Addresses.js    21
  > Addresses.json 22
  Picture.js      23
  Pictures.js     24
  App.js          25
  index.css       26
  index.js        27
  .gitignore      28
  craco.config.js 29
  package-lock.json 30
  package.json    31
  README.md       32

  <h2>Addresses :</h2>
  <button className="bg-white w-5 h-5 pb-1 font-thin rounded-xl
  text-center flex justify-center items-center absolute text-red-500
  right-0 text-sm focus:outline-none
  ${index} === showCloseButton ? '' : 'hidden'">
  onClick={() => handlerRemove(index)}>x</button>

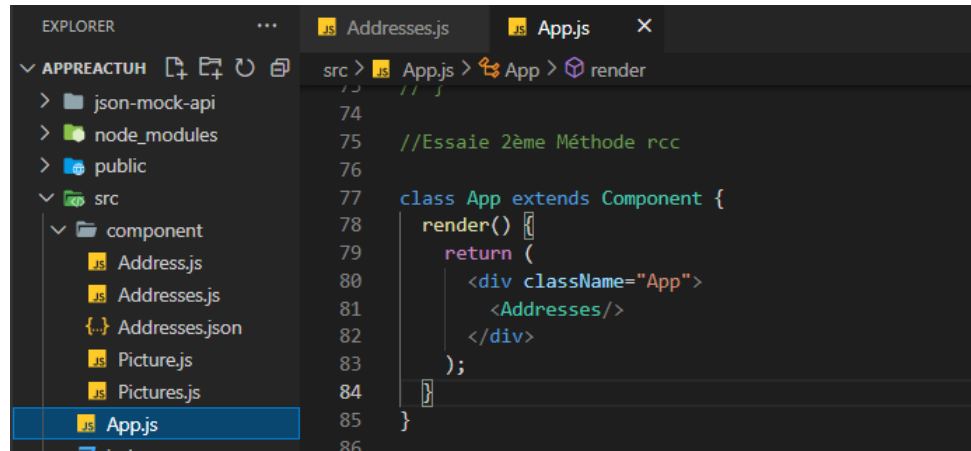
  <table className="table-fixed border border-black">
    <thead className="bg-purple-400">
      <tr>
        <th className="mr-5 p-3 text-center">Address 1</th>
        <th className="mr-5 p-3 text-center">Address 2</th>
        <th className="mr-5 p-3 text-center">City</th>
        <th className="mr-5 p-3 text-center">Zip Code</th>
      </tr>
    </thead>
    <tbody>
      <tr key={index}>
        <td className="mr-5 p-3 text-center">${item[0]}</td>
        <td className="mr-5 p-3 text-center">${item[1]}</td>
        <td className="mr-5 p-3 text-center">${item[2]}</td>
        <td className="mr-5 p-3 text-center">${item[3]}</td>
      </tr>
    </tbody>
  </table>
  <img className="w-40 mx-auto" src="" alt=[item[0] + item[1] + item[2] + item[3]]></img> */>
</div>
</div>
```

### Résultats dans le navigateur :



b. Méthode avec les composants React par **classe**.

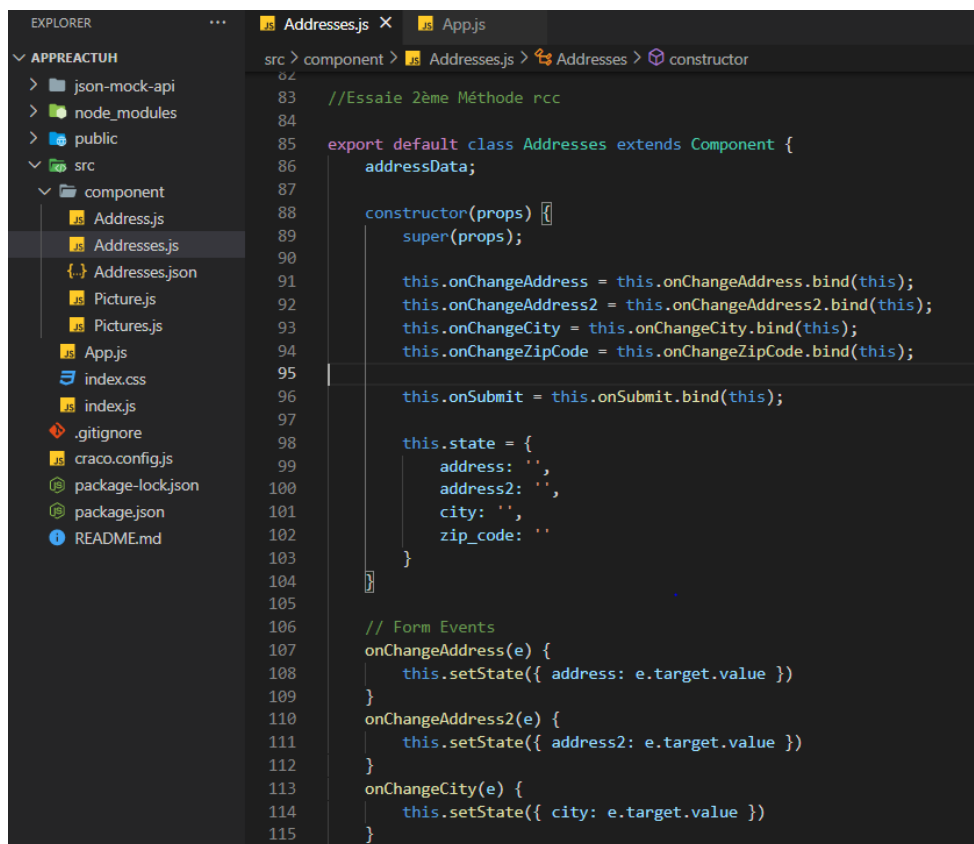
Sur cette partie j'ai tenté une autre approche, cette méthode est en bonne voie cependant elle n'est pas encore finie, le problème reste encore de faire persister la donnée en la stockant de manière à pouvoir l'afficher par la suite, j'ai trouvé quelques sources qui pourraient m'aider à faire cela. Avec ce code quand l'on appuie sur le bouton submit les données sont effacées cependant si l'on n'appuie pas sur ce bouton et que l'on rafraichit la page les données persistent<sup>678</sup>.



```

73 // ...
74
75 //Essaie 2ème Méthode rcc
76
77 class App extends Component {
78   render() {
79     return (
80       <div className="App">
81         <Addresses/>
82       </div>
83     );
84   }
85 }
86

```



```

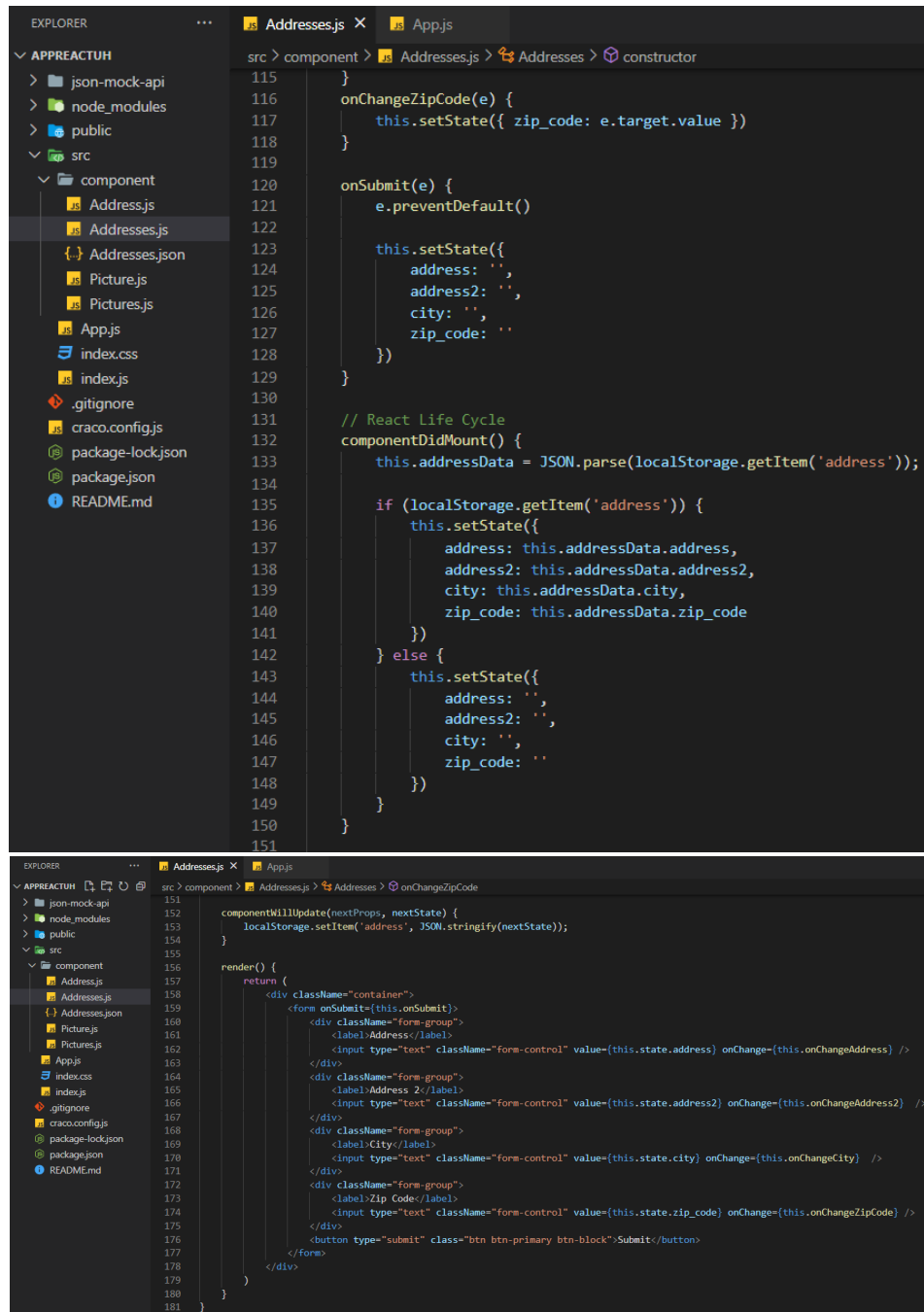
83 //Essaie 2ème Méthode rcc
84
85 export default class Addresses extends Component {
86   addressData;
87
88   constructor(props) {
89     super(props);
90
91     this.onChangeAddress = this.onChangeAddress.bind(this);
92     this.onChangeAddress2 = this.onChangeAddress2.bind(this);
93     this.onChangeCity = this.onChangeCity.bind(this);
94     this.onChangeZipCode = this.onChangeZipCode.bind(this);
95
96     this.onSubmit = this.onSubmit.bind(this);
97
98     this.state = {
99       address: '',
100       address2: '',
101       city: '',
102       zip_code: ''
103     };
104   }
105
106   // Form Events
107   onChangeAddress(e) {
108     this.setState({ address: e.target.value })
109   }
110   onChangeAddress2(e) {
111     this.setState({ address2: e.target.value })
112   }
113   onChangeCity(e) {
114     this.setState({ city: e.target.value })
115   }
116

```

<sup>6</sup> <https://www.positronx.io/store-react-form-data-or-state-in-local-storage/> (principal tutoriel suivi)

<sup>7</sup> <https://medium.com/@subalerts/create-dynamic-table-from-json-in-react-js-1a4a7b1146ef> (intéressant pour les méthodes mais peu clair)

<sup>8</sup> <https://www.digitalocean.com/community/tutorials/how-to-build-forms-in-react> (semble répondre à une partie de mon problème)



The image displays two screenshots of a code editor (VS Code) showing the development of a React application. The top screenshot shows the 'Addresses.js' file, and the bottom screenshot shows the 'render' method of the same file.

**Top Screenshot: Addresses.js**

```

src > component > Addresses.js > Addresses > constructor
115 }
116
117 onChangeZipCode(e) {
118   this.setState({ zip_code: e.target.value })
119 }
120
121 onSubmit(e) {
122   e.preventDefault()
123
124   this.setState({
125     address: '',
126     address2: '',
127     city: '',
128     zip_code: ''
129   })
130 }
131
132 // React Life Cycle
133 componentDidMount() {
134   this.addressData = JSON.parse(localStorage.getItem('address'));
135
136   if (localStorage.getItem('address')) {
137     this.setState({
138       address: this.addressData.address,
139       address2: this.addressData.address2,
140       city: this.addressData.city,
141       zip_code: this.addressData.zip_code
142     })
143   } else {
144     this.setState({
145       address: '',
146       address2: '',
147       city: '',
148       zip_code: ''
149     })
150   }
151 }

```

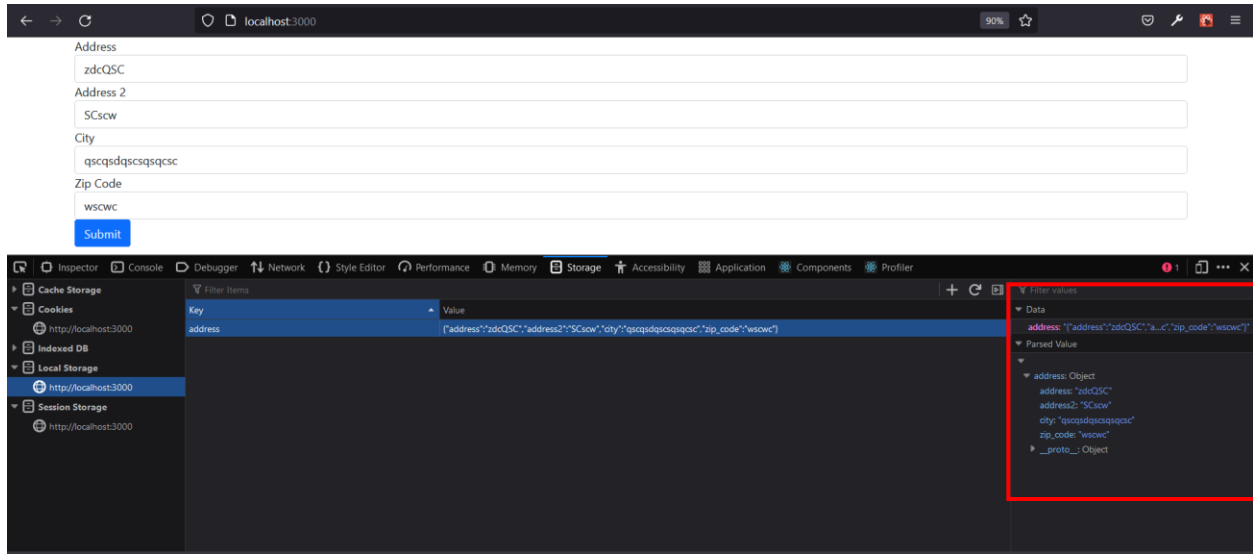
**Bottom Screenshot: Addresses.js**

```

src > component > Addresses.js > Addresses > onDidChangeZipCode
151
152 componentWillUpdate(nextProps, nextState) {
153   localStorage.setItem('address', JSON.stringify(nextState));
154 }
155
156 render() {
157   return (
158     <div className="container">
159       <form onSubmit={this.onSubmit}>
160         <div className="form-group">
161           <label>Address</label>
162           <input type="text" className="form-control" value={this.state.address} onChange={this.onChangeAddress} />
163         </div>
164         <div className="form-group">
165           <label>Address 2</label>
166           <input type="text" className="form-control" value={this.state.address2} onChange={this.onChangeAddress2} />
167         </div>
168         <div className="form-group">
169           <label>City</label>
170           <input type="text" className="form-control" value={this.state.city} onChange={this.onChangeCity} />
171         </div>
172         <div className="form-group">
173           <label>Zip Code</label>
174           <input type="text" className="form-control" value={this.state.zip_code} onChange={this.onChangeZipCode} />
175         </div>
176         <button type="submit" class="btn btn-primary btn-block">Submit</button>
177       </form>
178     </div>
179   )
180 }
181

```

## Résultats dans le navigateur :



VENDREDI 11 JUIN 2021

### MISSIONS:

1. Continuer les activités sur l'application *ReactJS*.

### REALISATIONS:

1.
  - a. Méthode avec les composants React par **classe**.

Je continue à exploiter cette façon et cherche à retrouver la même présentation que l'autre méthode<sup>9</sup>.

## En Cours ...

<sup>9</sup> <https://www.c-sharpcorner.com/article/how-to-show-and-hide-component-in-react-application/>