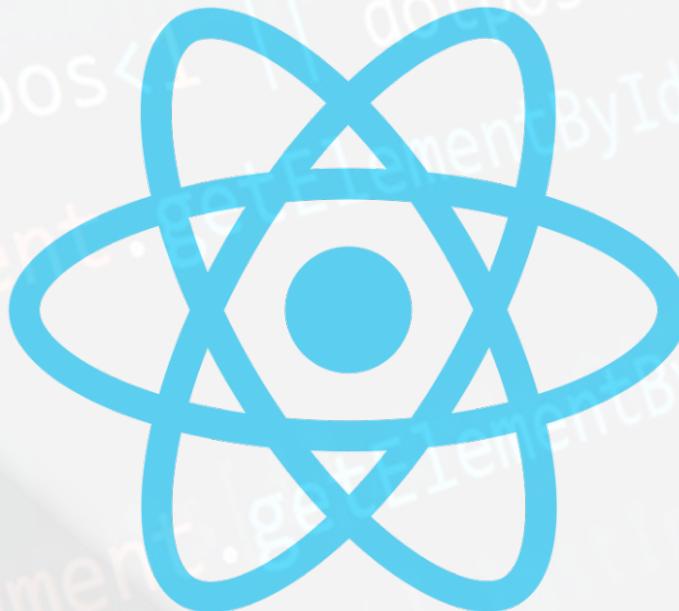
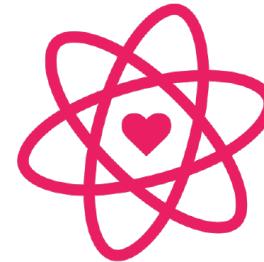


React js

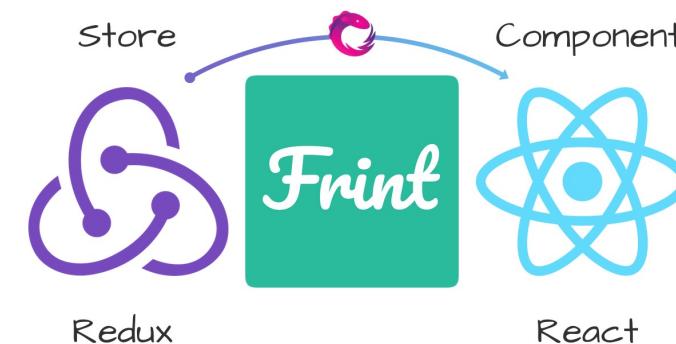


Giới thiệu



React JS

React && Sự tương tác giữa các component



Props

Props là gì ?

- ❑ Props là thuộc tính của thẻ (Ta có thể hiểu prop là property của thẻ). Ví dụ thẻ input bên dưới ta có các props **ClassName**, **type**, **placeholder**.

```
<input className="form-control" type="text" placeholder="Nhập họ tên"/>
```

Props đối với component

- ❑ Props là thuộc tính mặc định của component để nhận dữ liệu từ các giá trị component cha truyền vào => Để binding dữ liệu ra component con tại html tương ứng.
- ❑ Props của component chỉ nhận các thuộc tính được truyền vào từ component cha của nó và không thể bị chỉnh sửa bên trong component.
- ❑ Đối với stateful và stateless component có các cách sử dụng props khác nhau.

Prop trong react class component

- ❖ Đối với react class component, thì **props** là **thuộc tính mặc định của class** dùng để **nhận giá trị từ component cha**. Chỉ cần gọi **this.props**

❖ **Footer component (con)**

```
import React from 'react';

import './footer.css'  Truyền props vào dưới dạng
                      tham số
const footer = (props) => {
  |
  return (
    |
    <footer className="footer">
      <p>{props.name}</p>
    </footer>
  );
};

export default footer;  Export để có thể gọi ở các component
                      khác
```

❖ **Homelayout component (cha)**

```
import React, { Component, Fragment } from 'react'

import Footer from '../components/Home/footer';

export default class HomeLayout extends Component {
  name = "dang trung hieu";
  render() {
    return (
      //Code Jsx được trả về
      <Footer name={'hieu'}></Footer>
    );
  }
}
```

import footer component để có thể gọi nó ra trong homelayout component.

Vì bên footer component ta export default nên bên này có thể đặt lại tên cho nó tùy ý và không cần dấu

sử dụng props, trong đó name là thuộc tính được truyền vào từ component cha, ở đây sử home-layout component

Gọi footer component ra như một thẻ HTML thông thường

Trong đó name là thuộc tính ta truyền vào footerComponent với giá trị là chuỗi 'hieu'. Thay vì truyền chuỗi, ta có thể truyền vào biến name

Đó là ví dụ vì sao props của footer component lại có thuộc tính name

Prop trong react class component

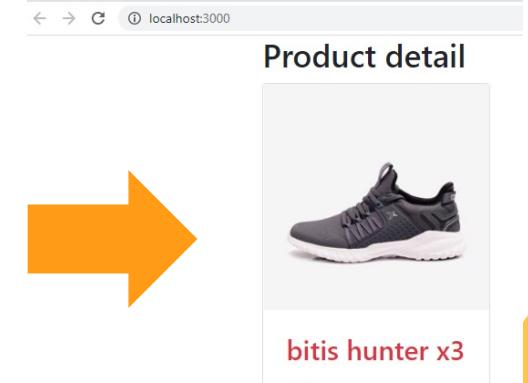
- ❖ Giá trị của **props** có thể là 1 giá trị **string**, **number**, **Boolean**, v...v... Ngoài ra giá trị props có thể là **object**, **array** hoặc là **function**.

```
1 import React, { Component } from 'react'  8.3K (gzipped: 3.3K)
2 import ChildComponent from './ChildComponent'
3
4 export default class DemoProps extends Component {
5
6     render() {
7
8         let product = [
9             id:1,
10            name:'bitis hunter x3',
11            price:'49$',
12            img: 'http://svcy2.myclass.vn/bitis/bitis.jpg'
13        ]
14
15         return (
16             <div className="container">
17                 <h3>Product detail</h3>
18                 <ChildComponent productProp ={product}/>
19             </div>
20         )
21     }
22 }
23 }
```

DemoPropsComponent

```
1 import React, { Component } from 'react'  8.3K (gzipped: 3.3K)
2
3 export default class ChildComponent extends Component {
4     render() {
5
6         //Lấy giá trị product từ component cha truyền vào thông qua this.props.productProp
7         let {productProp} = this.props; ← props nhận từ component DemoProps, dùng
8                                         destructuring để lấy giá trị từ props
9
10        return (
11            <div className="card text-left" style={{width:200}} >
12                <img className="card-img-top" src={productProp.img} alt={productProp.img}>
13                <div className="card-body">
14                    <h4 className="card-title text-danger">{productProp.name}</h4>
15                    <p className="card-text">{productProp.price}</p>
16                </div>
17            </div>
18        )
19    }
20 }
21 }
```

ChildComponent



Browser

Bài tập

❖ Cho mảng dataJson. Link:

https://drive.google.com/open?id=1M36QC9s4VreAV6UIRbgfq_VSk5m844i

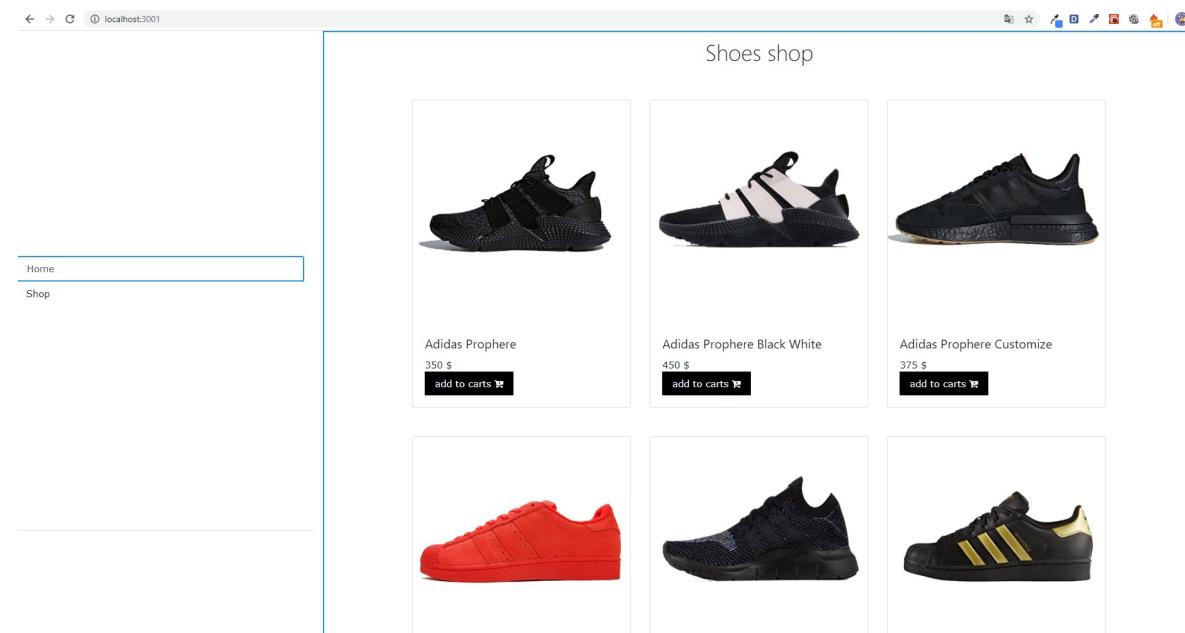
- Tạo cấu trúc component như sau:

<App>

```
<ProductList arrProduct={dataJson}>  
    <ProductItem item={item} />  
    <ProductItem item={item} />  
    <ProductItem item={item} />  
</ProductList>
```

</App>

- Yêu cầu dùng props để tạo giao diện như sau.Có thể dùng bootstrap, w3css, material ... để hỗ trợ.



Truyền sự kiện – nhận dữ liệu các cấp component



Vấn đề đặt ra:

Ta có 2 component là `DanhSachSanPham`, `SanPham`

+ `DanhSachSanPham` chứa: 2 thuộc tính là

- ◆ `mangSanPham`
- ◆ `state.sanPhamChiTiet`

Yêu cầu:

- Khi click vào nút xem chi tiết thì thông tin của sản phẩm được click sẽ được cập nhật vào `this.state.sanPhamChiTiet` làm cho nội dung bên dưới thay đổi
- Điều này thật đơn giản khi ta binding tất cả html chỉ trên 1 component duy nhất `DanhSachSanPham`.
- Tuy nhiên ta đã tách các `sanPham` ra thành riêng 1 component nên bên trong class của component `sanPham` không chứa `state.sanPhamChiTiet` nữa nên ta không thể cập nhật được.
- Do vậy tại component cha bắt buộc ta phải viết 1 hàm để truyền vào sự kiện click của component con để sau khi click thì nó sẽ lấy được dữ liệu tại hàm ta xây dựng ở component cha từ đó cập nhật lại `state.sanPhamChiTiet`
=> làm cho giao diện thay đổi

Danh sách sản phẩm



VinSmart Live

Xem chi tiết onclick



Meizu 16Xs

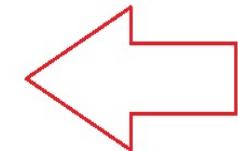
Xem chi tiết onclick



Iphone XS Max

Xem chi tiết onclick

`this.mangSanPham`



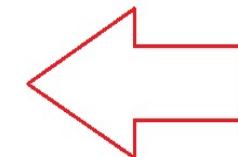
VinSmart Live



Thông số kỹ thuật

Màn hình	AMOLED, 6.2", Full HD+
Hệ điều hành	Android 9.0 (Pie)
Camera trước	20 MP
Camera sau	Chính 48 MP & Phụ 8 MP, 5 MP
RAM	4 GB
ROM	64 GB

`this.state.sanPhamChiTiet`



Truyền sự kiện – nhận dữ liệu các cấp component

Link hình ảnh: <https://drive.google.com/open?id=1g15gVNroRB6e2vOJBkuMwgCg9byVcbzt>

```
[  
  { "maSP": 1, "tenSP": "VinSmart Live", "manHinh": "AMOLED, 6.2, Full HD+", "heDieuHanh": "Android 9.0 (Pie)", "cameraTruoc": "20 MP", "cameraSau": "Chính 48 MP & Phụ 8 MP, 5 MP", "ram": "4 GB", "rom": "64 GB", "giaBan": 5700000, "hinhAnh": "./img/vsphone.jpg" },  
  { "maSP": 2, "tenSP": "Meizu 16Xs", "manHinh": "AMOLED, FHD+ 2232 x 1080 pixels", "heDieuHanh": "Android 9.0 (Pie); Flyme", "cameraTruoc": "20 MP", "cameraSau": "Chính 48 MP & Phụ 8 MP, 5 MP", "ram": "4 GB", "rom": "64 GB", "giaBan": 7600000, "hinhAnh": "./img/meizuphone.jpg" },  
  { "maSP": 3, "tenSP": "Iphone XS Max", "manHinh": "OLED, 6.5, 1242 x 2688 Pixels", "heDieuHanh": "iOS 12", "cameraSau": "Chính 12 MP & Phụ 12 MP", "cameraTruoc": "7 MP", "ram": "4 GB", "rom": "64 GB", "giaBan": 27000000, "hinhAnh": "./img/applephone.jpg" }  
]
```

✓ Code demo

```
4  export default class DanhSachSanPham extends Component {  
5  
6    constructor(props) {  
7      super(props);  
8      this.state = {  
9        sanPhamChiTiet: { maSP: 1, tenSP: 'VinSmart Live', manHinh: 'AMOLED, 6.2", Full HD+', heDieuHanh: 'Android 9.0 (Pie)',  
10          cameraTruoc: '20 MP', cameraSau: 'Chính 48 MP & Phụ 8 MP, 5 MP', ram: '4 GB', rom: '64 GB', giaBan: 5700000, hinhAnh: './img/  
11          vsphone.jpg' },  
12      }  
13    }  
14    mangDienThoai = [  
15      { maSP: 1, tenSP: 'VinSmart Live', manHinh: 'AMOLED, 6.2", Full HD+', heDieuHanh: 'Android 9.0 (Pie)', cameraTruoc: '20 MP',  
16          cameraSau: 'Chính 48 MP & Phụ 8 MP, 5 MP', ram: '4 GB', rom: '64 GB', giaBan: 5700000, hinhAnh: './img/vsphone.jpg' },  
17      { maSP: 2, tenSP: 'Meizu 16Xs', manHinh: 'AMOLED, FHD+ 2232 x 1080 pixels', heDieuHanh: 'Android 9.0 (Pie); Flyme', cameraTruoc:  
18          '20 MP', cameraSau: 'Chính 48 MP & Phụ 8 MP, 5 MP', ram: '4 GB', rom: '64 GB', giaBan: 7600000, hinhAnh: './img/meizuphone.jpg' },  
19      { maSP: 3, tenSP: 'Iphone XS Max', manHinh: 'OLED, 6.5", 1242 x 2688 Pixels', heDieuHanh: 'iOS 12', cameraSau: 'Chính 12 MP & Phụ  
20          12 MP', cameraTruoc: '7 MP', ram: '4 GB', rom: '64 GB', giaBan: 27000000, hinhAnh: './img/applephone.jpg' }  
21    ]
```

Truyền sự kiện – nhận dữ liệu các cấp component

✓ Code demo

Binding dữ liệu từ state => table chi tiết

```
<div className="row">
  <div className="col-md-4">
    <br />
    <h3 className="text-center">{sanPhamChiTiet.tenSP}</h3>
    <img className="card-img-top" src={sanPhamChiTiet.hinhAnh} width={170} height={300} alt={sanPhamChiTiet.tenSP} />
  </div>
  <div className="col-md-8">
    <table className="table">
      <thead>
        <tr>
          <td colSpan="2"><h3>Thông số kỹ thuật</h3></td>
        </tr>
        <tr>
          <td>Màn hình</td>
          <td>{sanPhamChiTiet.manHinh}</td>
        </tr>
        <tr>
          <td>Hệ điều hành</td>
          <td>{sanPhamChiTiet.heDieuHanh}</td>
        </tr>
        <tr>
          <td>Camera trước</td>
          <td>{sanPhamChiTiet.cameraTruoc}</td>
        </tr>
        <tr>
          <td>Camera sau</td>
          <td>{sanPhamChiTiet.cameraSau}</td>
        </tr>
        <tr>
          <td>RAM</td>
          <td>{sanPhamChiTiet.ram}</td>
        </tr>
        <tr>
          <td>ROM</td>
          <td>{sanPhamChiTiet.rom}</td>
        </tr>
      </thead>
      </table>
    </div>
  </div>
```

Truyền sự kiện – nhận dữ liệu các cấp component

✓ Code demo

DemoCallback.jsx • SanPham.jsx X

demo > src > components > Demo > SanPham.jsx > ...

```
1 import React, { Component } from 'react'
2
3 export default class SanPham extends Component {
4
5   render() {
6     let {sanPham} = this.props;
7     let {xemChiTiet} = this.props;
8     //Hoặc có thể khai báo
9     // let {sanPham,xemChitiet} = this.props;
10    return (
11      <div className="card text-left">
12        <img className="card-img-top" src={sanPham.hinhAnh} width={170} height={300} alt={'true'} />
13        <div className="card-body">
14          <h4 className="card-title">{sanPham.tenSP}</h4>
15          <button className="btn btn-success" onClick={() => xemChiTiet(sanPham)}>Xem chi tiết</button>
16        </div>
17      </div>
18    )
19  }
20 }
```

Prop và sự kiện nhận từ component cha

Sử dụng hàm từ component cha để truyền vào sự kiện click

SanPhamComponent.js

Truyền dữ liệu thông qua innerHTML của component với thuộc tính this.props.children

- Yêu cầu: Tạo cấu trúc component như bên dưới, xây dựng xử lý khi người dùng click vào nút xem chi tiết tại component ProductItem thì state của modal sẽ thay đổi.
Như hình ảnh bên dưới

```
<ExerciseCarStore>
```

```
    <Modal content={this.state.productDetail} />
```

```
    <ProductList productsData={products} setStateModal = {this.setStateModal}>
```

```
        <ProductItem item={product} setStateModal={this.props.setStateModal}> </ProductItem>
```

```
        <ProductItem item={product} setStateModal={this.props.setStateModal}> </ProductItem>
```

```
        <ProductItem item={product} setStateModal={this.props.setStateModal}> </ProductItem>
```

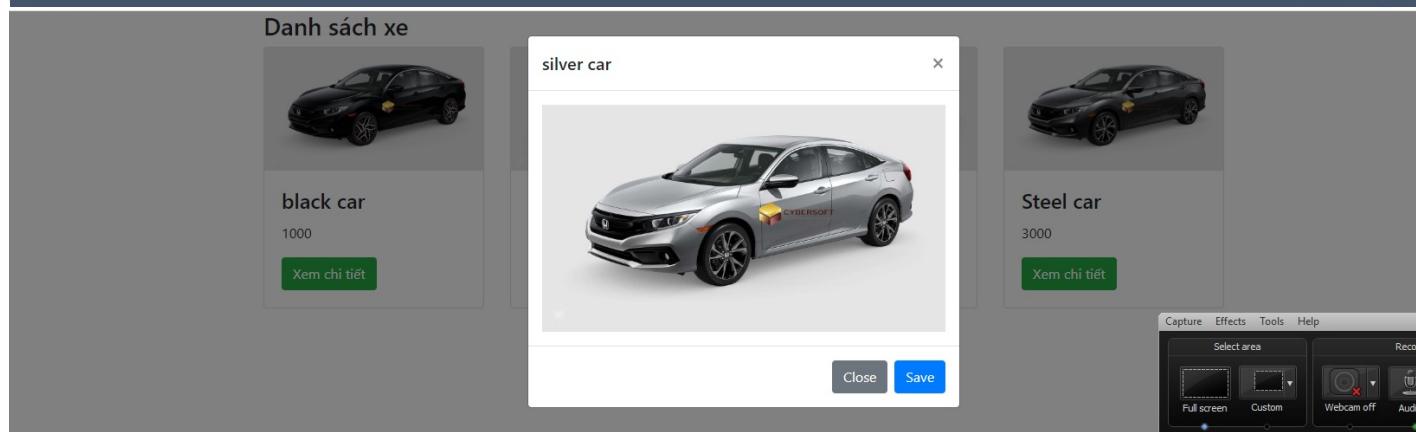
```
    </ ProductList >
```

```
</ExerciseCarStore >
```

Sử dụng link để lấy hình ảnh: <https://drive.google.com/open?id=1kaGEHc7IQJYCuXO-GHRO-KUFY1Xj9LHD>

Sử dụng data sau (Nhớ config lại so với thư mục hình ảnh của source_hiện tại):

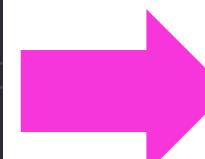
```
products = [
    { id: 1, name: 'black car', img: './images/products/black-car.jpg', price: 1000 },
    { id: 2, name: 'red car', img: './images/products/red-car.jpg', price: 2000 },
    { id: 3, name: 'silver car', img: './images/products/silver-car.jpg', price: 3000 },
    { id: 3, name: 'Steel car', img: './images/products/steel-car.jpg', price: 4000 }
];
```



Truyền dữ liệu thông qua innerHTML của component với thuộc tính this.props.children

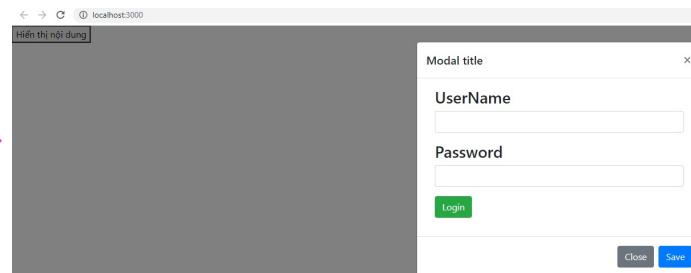
- Tương tự truyền dữ liệu sang thuộc tính props của component, reactjs còn cho phép ta truyền nội dung từ component cha sang component con thông qua phần **innerHTML** của component. Nội dung đó sẽ được hiển thị thông qua thuộc tính **this.props.children** từ component con. Thường ứng dụng khi giá trị cần truyền là 1 tag jsx hoặc 1 component nào khác.

```
JS App.js
video_cyberlearn > src > JS App.js > App > render
1 import React, { Component } from 'react' 8.3K (gzipped: 3.3K)
2 import DemoPropsChildren from './Props/DemoPropsChildren/DemoPropsChildren'
3 import FormLogin from './Props/DemoPropsChildren/FormLogin'
4 import FormRegister from './Props/DemoPropsChildren/FormRegister'
5
6 export default class App extends Component {
7
8   render() {
9     return [
10       <div>
11         <DemoPropsChildren>
12           <FormRegister /> ← Nội dung được gửi từ component app =>
13         </DemoPropsChildren> ← component DemoPropsChildren nằm ở phần
14
15         <button data-toggle="modal" data-target="#modelId">Hiển thị nội dung</button>
16       </div>
17     ]
18   }
19 }
20 }
```



```
video_cyberlearn > src > Props > DemoPropsChildren > JS DemoPropsChildren.js > render
1 import React, { Component } from 'react' 8.3K (gzipped: 3.3K)
2
3 export default class DemoPropsChildren extends Component {
4   render() {
5     return (
6       <div className="modal fade" id="modelId" tabIndex={-1} role="dialog" aria-labelledby="modelTitleId" aria-
7
8         <div className="modal-dialog" role="document">
9           <div className="modal-content">
10             <div className="modal-header">
11               <h5 className="modal-title">Modal title</h5>
12               <button type="button" className="close" data-dismiss="modal" aria-label="Close">
13                 <span aria-hidden="true">&times;</span>
14               </button>
15             </div>
16             <div className="modal-body">
17               {this.props.children} ← Nơi nội dung nhận từ
18               component cha được
19               binding ra
20             </div>
21             <div className="modal-footer">
22               <button type="button" className="btn btn-secondary" data-dismiss="modal">Close</button>
23               <button type="button" className="btn btn-primary">Save</button>
24             </div>
25           </div>
26         </div>
27     )
28   }
29 }
```

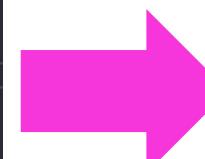
Kết quả hiển thị



Truyền dữ liệu thông qua innerHTML của component với thuộc tính this.props.children

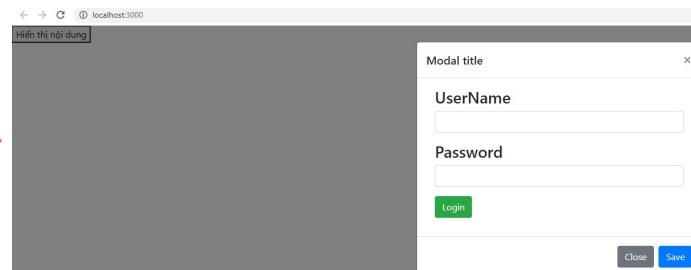
- Tương tự truyền dữ liệu sang thuộc tính props của component, reactjs còn cho phép ta truyền nội dung từ component cha sang component con thông qua phần **innerHTML** của component. Nội dung đó sẽ được hiển thị thông qua thuộc tính **this.props.children** từ component con. Thường ứng dụng khi giá trị cần truyền là 1 tag jsx hoặc 1 component nào khác.

```
JS App.js
video_cyberlearn > src > JS App.js > App > render
1 import React, { Component } from 'react' 8.3K (gzipped: 3.3K)
2 import DemoPropsChildren from './Props/DemoPropsChildren/DemoPropsChildren'
3 import FormLogin from './Props/DemoPropsChildren/FormLogin'
4 import FormRegister from './Props/DemoPropsChildren/FormRegister'
5
6 export default class App extends Component {
7
8   render() {
9     return [
10       <div>
11         <DemoPropsChildren>
12           <FormRegister /> ← Nội dung được gửi từ component app =>
13         </DemoPropsChildren> ← component DemoPropsChildren nằm ở phần
14
15         <button data-toggle="modal" data-target="#modelId">Hiển thị nội dung</button>
16       </div>
17     ]
18   }
19 }
20 }
```



```
video_cyberlearn > src > Props > DemoPropsChildren > JS DemoPropsChildren.js > render
1 import React, { Component } from 'react' 8.3K (gzipped: 3.3K)
2
3 export default class DemoPropsChildren extends Component {
4   render() {
5     return (
6       <div className="modal fade" id="modelId" tabIndex={-1} role="dialog" aria-labelledby="modelTitleId" aria-
7
8         <div className="modal-dialog" role="document">
9           <div className="modal-content">
10             <div className="modal-header">
11               <h5 className="modal-title">Modal title</h5>
12               <button type="button" className="close" data-dismiss="modal" aria-label="Close">
13                 <span aria-hidden="true">&times;</span>
14               </button>
15             </div>
16             <div className="modal-body">
17               {this.props.children} ← Nơi nội dung nhận từ
18               component cha được
19               binding ra
20             </div>
21             <div className="modal-footer">
22               <button type="button" className="btn btn-secondary" data-dismiss="modal">Close</button>
23               <button type="button" className="btn btn-primary">Save</button>
24             </div>
25           </div>
26         </div>
27     )
28   }
29 }
```

Kết quả hiển thị



Bài tập state props – thực hành giỏ hàng

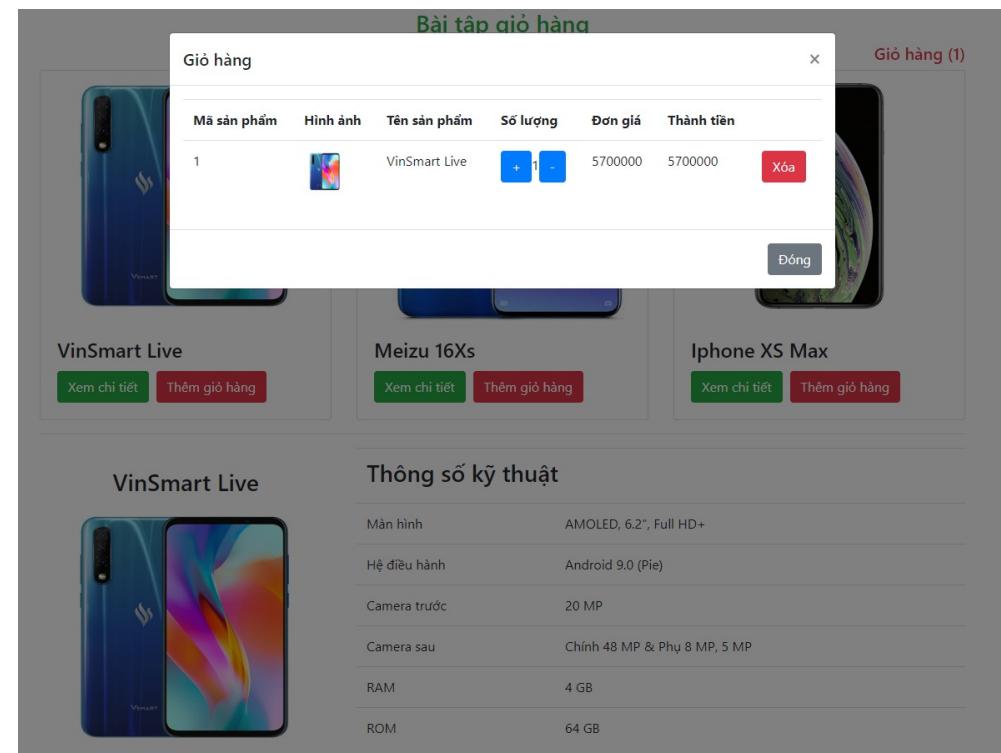
- Yêu cầu: Tạo cấu trúc component như bên dưới, xây dựng xử lý khi người dùng click vào nút xem chi tiết tại component ProductItem thì state của modal sẽ thay đổi. Như hình ảnh bên dưới

```
<ExerciseCart>  
    <Cart cartData={cartData} />  
  
    <ProductList productsData={products} addToCart = {this.addToCart}>  
        <ProductItem item={product} addToCart={this.props.addToCart}> </ProductItem>  
        <ProductItem item={product} addToCart ={this.props.addToCart}> </ProductItem>  
        <ProductItem item={product} addToCart ={this.props.addToCart}> </ProductItem>  
    </ ProductList >  
</ ExerciseCart >
```

Sử dụng link để lấy hình ảnh: <https://drive.google.com/open?id=1g15gVNroRB6e2vOJBkuMwgCg9byVcbzt>

Sử dụng data sau (Nhớ config lại so với thư mục hình ảnh của source_hiện tại):

```
mangSanPham = [  
    { "maSP": 1, "tenSP": "VinSmart Live", "manHinh": "AMOLED, 6.2, Full HD+", "heDieuHanh": "Android 9.0 (Pie)", "cameraTruoc": "20 MP", "cameraSau": "Chính 48 MP & Phụ 8 MP, 5 MP", "ram": "4 GB", "rom": "64 GB", "giaBan": 5700000, "hinhAnh": "./img/vsphone.jpg" },  
    { "maSP": 2, "tenSP": "Meizu 16Xs", "manHinh": "AMOLED, FHD+ 2232 x 1080 pixels", "heDieuHanh": "Android 9.0 (Pie); Flyme", "cameraTruoc": "20 MP", "cameraSau": "Chính 48 MP & Phụ 8 MP, 5 MP", "ram": "4 GB", "rom": "64 GB", "giaBan": 7600000, "hinhAnh": "./img/meizuphone.jpg" },  
    { "maSP": 3, "tenSP": "Iphone XS Max", "manHinh": "OLED, 6.5, 1242 x 2688 Pixels", "heDieuHanh": "iOS 12", "cameraSau": "Chính 12 MP & Phụ 12 MP", "cameraTruoc": "7 MP", "ram": "4 GB", "rom": "64 GB", "giaBan": 27000000, "hinhAnh": "./img/applephone.jpg" }]
```



Tổng hợp kiến thức

❖ Các bước làm bài tập với state props (Cách xác định state)

- 1. Bước 1 chia component và dàn layout**
- 2. Bước 2 xác định thành phần thay đổi của giao diện (state). State là gì ? Là **string** hay **number** hay **object** hay **array****
- 3. Bước 3 xác định component nào chứa cả giao diện và cả nút xử lý => đặt state và các hàm xử lý (Hàm chứa phương thức setState) tại đó.**
- 4. Truyền các hàm xử lý đến nơi chứa các nút xử lý**