***Redux***

**1        Redux là gì? Tại sao phải sử dụng redux?**

* *Redux là một công cụ quản lý trạng thái có thể dự đoán được cho ứng dụng Javascript.*
* *state của ứng dụng được lưu trong một nơi gọi là store và mỗi component đều có thể access bất kỳ state nào mà chúng muốn từ chúng store này.*

**2        Các thành phần chính của redux? Flow hoạt động của redux?**

* Có 3 thành phần chính của Redux: Actions, Store, Reducers.

**Flow hoạt động:**

*1. The View yêu cầu 1 action. Action Creator định dạng (format) yêu cầu và gửi lại*

*2. Action được gửi tự động (nếu bindActionCreators() đã được chuẩn bị) hoặc The View sẽ gửi*

*3. The Store nhận Action sau đó gửi State tree hiện tại và Action cho Root Reducer*

*4. Root Reducer chia State ra thành nhiều phần và gửi cho từng subreducers biết cách xử lý chúng*

*5. Subreducers tạo ra 1 bản copy từ phần nhận được và thay đổi trên bản copy. Sau đó gửi lại bản copy cho Root Reducer*

*6. Khi tất cả subreducers trả về các phần copies, Root Reducer ghép chúng lại tạo thành 1 update State tree và gửi lại cho Store. Store thay thế State tree cũ bằng State tree mới.*

*7. Store nói với The view layer binding là có State mới*

*8. The view layer binding báo Store gửi State mới cho mình*

*9. The view layer binding kích hoạt render view*

**3        Giải thích chức năng của các thành phần bên dưới:**  
 ***o        Redux Actions***

***Khái niệm:***

* Action là sự kiện.
* Là cách duy nhất để gửi dữ liệu từ ứng dụng đến store.
* Các action được gửi bằng phương thức store.dispatch().
* Các action là các đối tượng javascript
* Thuộc tính type để chỉ ra hành động sẽ thực hiện.
* Thuộc tính payload chứa thông tin dữ liệu cần được gửi đi.

***Ví dụ :***

export const getProduct = (categoryId) => {

return {

type : GET\_PRODUCT',

payload: categoryId

}

};

Action này trả về 1 plain object bao gồm

Type: Tên action

Payload: dữ liệu cần gửi đi

Tại nơi gọi action sẽ có mã như sau:

Dispatch(getProduct(categoryId));

***o        Redux Reducer***

***Khái niệm:***

* Reducer là các hàm pure function lấy trạng thái hiện tại của ứng dụng.
* Reducer thực hiện một hành động và trả về trạng thái mới.
* Dựa trên hàm reduce trong javascript,  trong đó một giá trị được tính từ nhiều giá trị sau khi thực hiện chức năng gọi lại.

***Ví dụ:***

const initialState = {

products: []

};

export default (state = initialState, action) => {

switch (action.type){

case GET\_PRODUCT':

return {

…state,

products: action.payload

}

default:

return state;

}

};

Có thể coi reducer là một bộ chuyển đổi với

Input là state hiện tại, action

Output là state mới tương ứng

***o        Redux Store***

***Khái niệm:***

* Store lưu giữ trạng thái của ứng dụng.
* Mỗi reducer sẽ xử lý và trả về state. Tập hợp các state này sẽ thành cây state mà Store nắm giữ.
* Chỉ có một Store

***Ví dụ:***

import {combineReducers} from 'redux'

import todos\_reducer from './todos'

import get\_product\_reducer from './get\_Product\_filter'

export const rootreducer = combineReducers({todos: todos\_reducer, get\_product: get\_product\_reducer});

Tại nơi khai báo và sử dụng store.

Tạo store và truyền cho Provider để tạo ra một mạng lưới kết nối

const store = createStore(rootreducer, window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_ && window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_());

<Provider store={store}>

<App/>

</Provider>, document.getElementById('root')

***o        Redux Middleware***

***Khái niệm:***

* Middleware cho phép chúng ta can thiệp vào giữa thời điểm dispatch một action và thời điểm action đến được reducer.
* Một middleware function là 1 function return 1 function return 1 function
* Một số thư viện để làm việc với Middleware như: redux-thunk, redux-saga, redux-observable