MỤC TIÊU:

Kết thúc bài thực hành này bạn có khả năng

- ✓ Redux
- ✓ React với redux
- ✓ Redux-thunk

NỘI DUNG

Bài 1: Redux

Sử dụng codepen.io thực hiện bài tập ví dụ về Redux

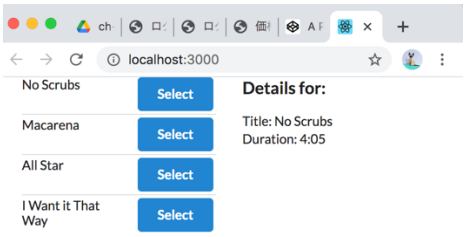
```
const createBag = (withdraw) => {
  return {
    type: 'deposit',
    payload:{
      withdraw:withdraw
    }
  };
};
```

```
//reducers
const bagReducers = (moneyofbag = 100, action) => {
  if (action.type === 'deposit') {
    return action.payload.withdraw + moneyofbag;
  }

  return moneyofbag;
};
```

```
const {createStore, combineReducers} = Redux;
const claimDepartments = combineReducers ({
  moneyofbag: bagReducers,
});

const store = createStore(claimDepartments);
store.dispatch(createBag(100));
console.log(store.getState());
```

Bài 2: React-Redux

- Tạo app project có tên là songs với cấu trúc, mô tả và hướng dẫn như sau
- Yêu cầu



- Cấu trúc

/src	
/actions	chứa các files liên quan action creators
/components	files liên quan components
/reducers	files liên quan reducers
index.js	cài đặt cả react và redux trong app

- Hướng dẫn

index.js

src > actions > JS index.js > ...

```
1   // Action creator
2   export const selectSong = song => {
3     // Return an action
4     return {
5       type: 'SONG_SELECTED',
6       payload: song
7     };
8   };
```

index.js

src > reducers > index.js > ...

```
import { combineReducers } from 'redux';

const songsReducer = () => {
  return [
    { title: 'No Scrubs', duration: '4:05' },
    { title: 'Macarena', duration: '2:30' },
    { title: 'All Star', duration: '3:15' },
    { title: 'I Want it That Way', duration: '1:45' }
  ];
};

const selectedSongReducer = (selectedSong = null, action) => {
  if (action.type === 'SONG_SELECTED') {
    return action.payload;
  }

  return selectedSong;
};

export default combineReducers({
  songs: songsReducer,
  selectedSong: selectedSongReducer
});
```

SongList.js 1

src > components > SongList.js > default

```
import React, { Component } from 'react';
import { connect } from 'react-redux';
import { selectSong } from '../actions';

class SongList extends Component {
  renderList() {
    return this.props.songs.map(song => {
      return (
        <div className="item" key={song.title}>
          <div className="right floated content">
            <button
              className="ui button primary"
              onClick={() => this.props.selectSong(song)}
            >
              Select
            </button>
          </div>
          <div className="content">{song.title}</div>
        </div>
      );
    });
  }

  render() {
    return <div className="ui divided list">{this.renderList()}</div>;
  }
}

const mapStateToProps = state => {
  return { songs: state.songs };
};

export default connect(
  mapStateToProps,
  { selectSong }
)(SongList);
```

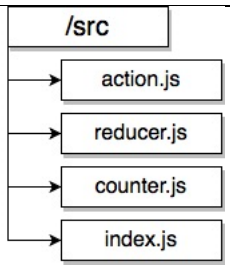
SongDetail.js 1

src > components > SongDetail.js > ...

```
import React from 'react';
import { connect } from 'react-redux';

const SongDetail = ({ song }) => {
  if (!song) {
    return <div>Select a song</div>;
  }

  return (
    <div>
      <h3>Details for:</h3>
      <p>
        Title: {song.title}
        <br />
        Duration: {song.duration}
      </p>
    </div>
  );
};

const mapStateToProps = state => {
  return { song: state.selectedSong };
};

export default connect(mapStateToProps)(SongDetail);
```

Bài 3: Middleware

- Tạo project có tên là middleware có cấu trúc như sau



- Mô tả

- Hướng dẫn

```
// The action creators
export const increment = () => ({ type: "INC" });
export const decrement = () => ({ type: "DEC" });
export const reset = () => ({ type: "RESET" });
```

```
// The reducer updates the count
const initialState = {
  count: 0
};
export const reducer = (state = initialState, action) => {
  switch (action.type) {
    case "INC":
      return { ...state, count: state.count + 1 };
    case "DEC":
      return { ...state, count: state.count - 1 };
    case "RESET":
      return { ...state, count: 0 };
    default:
      return state;
  }
};
```

Counter.js actions.js index.js reducer.js

```
import React from "react";
import { connect } from "react-redux";
import { increment, decrement, reset } from "./actions";

function Counter({ count, dispatch }) {
  return (
    <div>
      <h1>{count}</h1>
      <button onClick={() => dispatch(decrement())}>-</button>
      <button onClick={() => dispatch(increment())}>+</button>
      <button onClick={() => dispatch(reset())}>reset</button>
    </div>
  );
}

const mapStateToProps = (state) => ({
  count: state.count
});

export default connect(mapStateToProps)(Counter);
```

Counter.js actions.js index.js reducer.js

```
import React from "react";
import ReactDOM from "react-dom";
import { Provider } from "react-redux";
import { createStore } from "redux";

import Counter from "./Counter";
import { reducer } from "./reducer";

// The store holds the data
const store = createStore(reducer);

// The App renders everything
function App() {
  return (
    <div className="App">
      <Counter />
    </div>
  );
}

// The Provider passes the store through the app
// so connect() can access it
const rootElement = document.getElementById("root");
ReactDOM.render(
  <Provider store={store}>
    <App />
  </Provider>,
  rootElement
);
```

Bài 4: Async Middleware Thunk

- Tạo project có tên là blogs có cấu trúc như sau

- Hướng dẫn

index.js 1

src > actions > index.js > ...

```
1  import jsonPlaceholder from '../apis/jsonPlaceholder';
2
3  export const fetchPosts = () => async dispatch => {
4    const response = await jsonPlaceholder.get('/posts');
5
6    dispatch({ type: 'FETCH_POSTS', payload: response });
7  };
8
```

index.js 1

src > reducers > index.js > ...

```
1  import { combineReducers } from 'redux';
2
3  export default combineReducers({
4    replaceMe: () => 'hi there'
5  });
6
```

App.js 1

src > components > App.js > ...

```
1   import React from 'react';
2   import PostList from './PostList';
3
4   const App = () => {
5     return (
6       <div className="ui container">
7         <PostList />
8       </div>
9     );
10  };
11
12  export default App;
13
```

PostList.js 1

src > components > PostList.js > PostList

```
1   import React from 'react';
2   import { connect } from 'react-redux';
3   import { fetchPosts } from '../actions';
4
5   class PostList extends React.Component {
6     componentDidMount() {
7       console.log(this.props.fetchPosts());
8     }
9
10    render() {
11      return <div>Post List</div>;
12    }
13  }
14
15  export default connect(
16    null,
17    { fetchPosts }
18  )(PostList);
19
```

JS index.js 1 ×

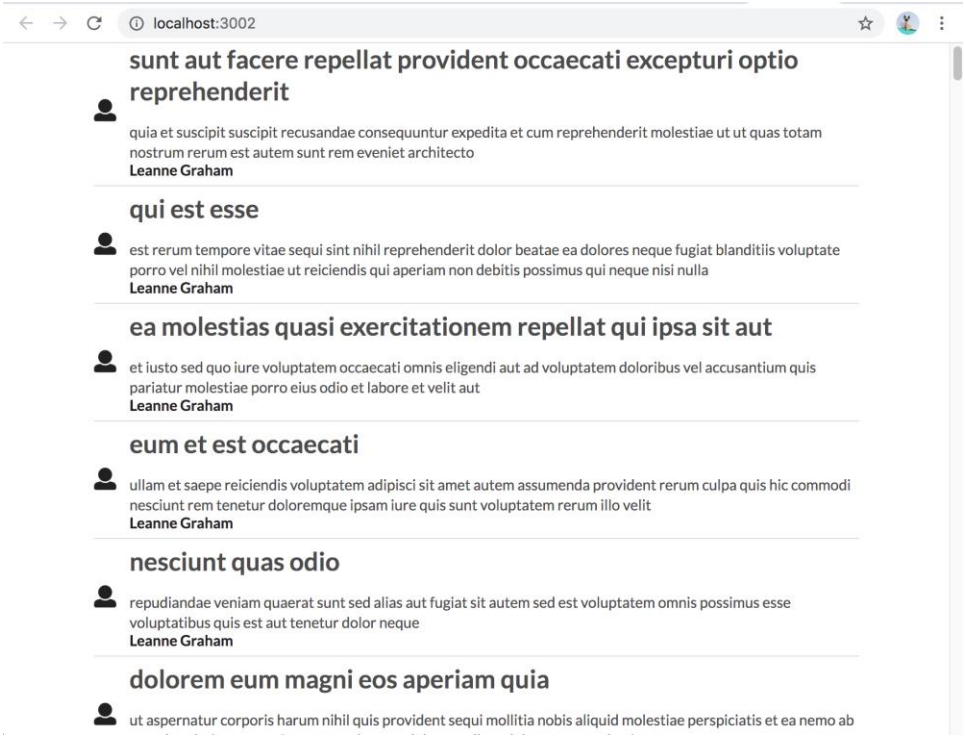
src > JS index.js > ...

```
1   import React from 'react';
2   import ReactDOM from 'react-dom';
3   import { Provider } from 'react-redux';
4   import { createStore, applyMiddleware } from 'redux';
5   import thunk from 'redux-thunk';
6
7   import App from './components/App';
8   import reducers from './reducers';
9
10  const store = createStore(reducers, applyMiddleware(thunk));
11
12  ReactDOM.render(
13    <Provider store={store}>
14      <App />
15    </Provider>,
16    document.querySelector('#root')
17  );
18
```

Bài 5:

- Giảng viên cho thêm

  Gợi ý: Hoàn thành project blogs



*** Yêu cầu nộp bài:

SV nén file (*hoặc share thư mục google drive*) bao gồm các yêu cầu đã thực hiện trên, nộp LMS đúng thời gian quy định của giảng viên. KHÔNG NỘP BÀI COI NHƯ KHÔNG CÓ ĐIỂM.

--- Hết ---