**What is react**

* A JavaScript library for building user interfaces
* React JS is the most popular JS library in today’s times

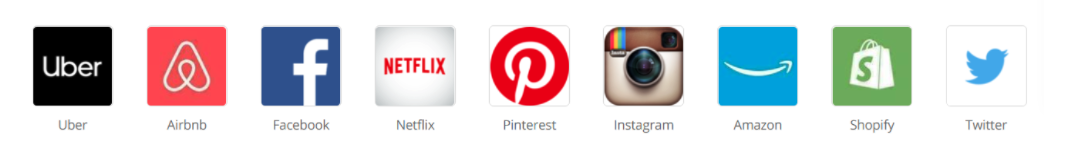
### Declarative code for developers

### Component-Based UIs

### Learn Once, Write Anywhere

* Most popular for SPA
* Easy and flexible to learn and use

**Who uses React Js**

****

**History of React**

React was created by Jordan Walke, a software engineer at Facebook, who released an early prototype of React called "FaxJS". He was influenced by [XHP](https://en.wikipedia.org/wiki/XHP), an [HTML](https://en.wikipedia.org/wiki/HTML) component library for [PHP](https://en.wikipedia.org/wiki/PHP). It was first deployed on Facebook's [News Feed](https://en.wikipedia.org/wiki/News_Feed) in 2011 and later on [Instagram](https://en.wikipedia.org/wiki/Instagram) in 2012.It was open-sourced at JSConf US in May 2013



**Before React**

* javaScript core
* ES6 with all modern javaScript
* Frontend building experiences
* API use and maintains
* Webpack & babel env

**Install & Setup React JS**

* Install latest version of node js with **yarn** and **npm**
* Yarn is recommended for react developer
* **Then start to install react app**   
  npx create-react-app app-name
* **Start your react App**   
  npm start   
  yarn start

**React app structure**

* Clean react app first
* Core template file   
  public/index.html
* Entry javaScript file   
  src/index.js
* Application file   
  src/app.js
* **Work flow of react env**

**Application Structure**

* Pages
* Components
* privateRoutes
* fakeData
* utility
* Redux / contexts / reducers / recoiles
* Assets

**JSX**

* javaScript XML
* Html code in JS
* Use any js from { js expression }in JSX
* Style JSX elements
* **Rules of JSX**  
  use a single root / fragment root  
  For multiline use empty fragment   
  Any js code expression use { }  
  Avoid semicolon in JSX   
  For conditional avoid {} for statement

**team member react app**

* Setup a JSON Server for developers
* Install axios for api response
* Create a data for developers and use it
* Design a team member template by bootstrap and boxicon
* Render all data in react apps

**React Components**

* React Components are reusable container / template
* **Component type -** there are 2 types of component   
  **Functional component**  
  **Class** **Component**
* **Functional** component   
  function Student (props){  
   return(  
   your JSX code goes here   
   );  
  }
* **Class** Componentclass Student extends React.Component {  
   constructor(props){  
   super(props);  
   }  
   render(){  
   return(  
   your jsx code goes here   
   );  
   }  
    
  }
* **Props drilling for data flow**   
  For data pass to a component we need props drilling

**<Student props1={ value2} props2={ value2 } />**

* **Get props value in functional component** function Student (props){  
   return(  
   your JSX code goes here   
   );  
   }
* **Get props value in class components**   
  class Student extends React.Component {  
   constructor(props){  
   super(props);  
   }  
   render(){  
   return(  
   your jsx code goes here   
   );  
   }  
    
  }
* **CSS styling - inline**   
  <h1 style={ { color : ‘red’, background-color:black; } }></h1>  
    
  const heading = {  
   color : ‘red’,  
   backgroundColor : ‘black’  
  }  
  <h1 style={ heading }></h1>
* Import css or sass file for styling

**React Events**

* Just like HTML DOM events, React can perform actions based on user events
* React has the same events as HTML: click, change, mouseover etc.
* Set a event to a elements with handler   
  <button onClick={ handlerFunction }></button>
* Set a event with args to a elements  
  <button onClick={ (e) => handlerFunction(value) }></button>

**State Management**

* A template dynamic data controlled by react a state
* State is a system to pass data one component to another
* We can manage state by useState hook
* We can also manage state by   
  useState   
  Context API   
  Reducers   
  Redux
* **Declare a state by useState hook**   
  const [ counter, setCounter ] = useState(‘default value’);
* Create a counter project with state management
* Create a alert management project with state management
* Create a data loading state from **JSON Server API** for data management

**useEffect hook**

* For render effect control with effect
* Reload any api response or data change after a successful state change
* **Run on every render**   
  useEffect( () => {  
    
  });
* **Run on first render**   
  useEffect( () => {  
    
  }, []);
* **Run on first render** & also run when state value change   
  useEffect( () => {  
    
  }, [props]);

**Form Data State**

* Create a form with different fields
* Now create state for all inputs fields
* Now set value fields value and state update   
  const [ name, setName ] = useState(‘’);  
    
  <input type=”text” value={ name }>  
  <input type=”text” value={ name } onChange={ handleNameChange } >
* Update input fields value   
  <input type=”text” value={ name } onChange={ (e) => setName( e.target.value ) } >
* Create a Form validation project

**React Bootstrap**

* Use **Bootstrap** in react nicely
* Fully functional **Bootstrap** in React
* Component elements use
* You must know the core bootstrap for professional use
* **Install React Bootstrap with Bootstrap**   
  -> npm install react-bootstrap bootstrap   
  -> yarn add react-bootstrap bootstrap
* **Import Bootstrap stylesheet in index.js file**  
  -> import 'bootstrap/dist/css/bootstrap.min.css';
* Now Load any components to react app and use it
* **Use react bootstrap components**   
  Modal   
  Alert   
  Table   
  Form   
  Card   
  Offcanvas   
  Dropdown   
  Grid
* Create a complete **developers** CRUDS apps by using React Bootstrap, JSON Server, Modal system
* Create a complete Online Product CRUDS apps by using React Bootstrap, JSON Server, Modal

**React Router Dom**

* Manage route for **SPA**
* Also powerful for **nested** route managements
* Use in mobile apps for **native apps development**
* Too easy to **maintain** for large apps
* **Install React Router Dom**-> npm install react-router-dom@6   
  -> yarn add react-router-dom@6
* **Create a basic React Router**   
  <BrowserRouter>  
   <Routes>  
   <Route path=’ ’ elements={} />  
   </Routes>  
  </BrowserRouter>
* **Create a navigation menu with the router**<Link to=’ ’ >
* **Add a 404 page to the router**<BrowserRouter>  
   <Routes>  
   <Route path=’ ’ elements={} />  
   <Route path=’\* ’ elements={} />  
   </Routes>  
  </BrowserRouter>
* **Add NavLink in navigation**-> Replace **Link** to **NavLink** Component  
  -> add a custom CSS for active nav item styles
* **Link Object Interface**-> pathname   
  -> search  
  -> hash   
  -> state   
  <Link to={{  
   pathname: ‘/path’,  
   search: ‘?s=haq’,  
   hash: ‘#name’,  
   state: { stateName: value }  
  }} />
* **React router params**
* **Nested Route**-> Nested routes are used for **components** to **components** **routing managements**   
  -> syntax of nested route   
  <BrowserRouter>  
   <Routes>  
   <Route path=’ ’ elements={} />  
   <Route path=’\* ’ elements={} />  
   <Route path=’/dashboard ’ elements={} >  
   <Route path=’admin ’ elements={} />  
   <Route path=’password ’ elements={} />  
   </Route>  
   </Routes>  
  </BrowserRouter> -> Use **<OutLet/>** for load nested components
* Now create a **single page online store application**  with product and single product
* Create a **facebook timeline** project with react router dom
* **Porto theme** clone by react with router dom and bootstrap