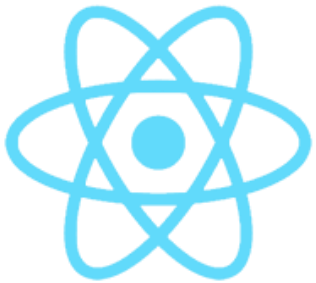
YASH Technologies

React Component & data flow with props



React

Created by : Pankaj Sharma

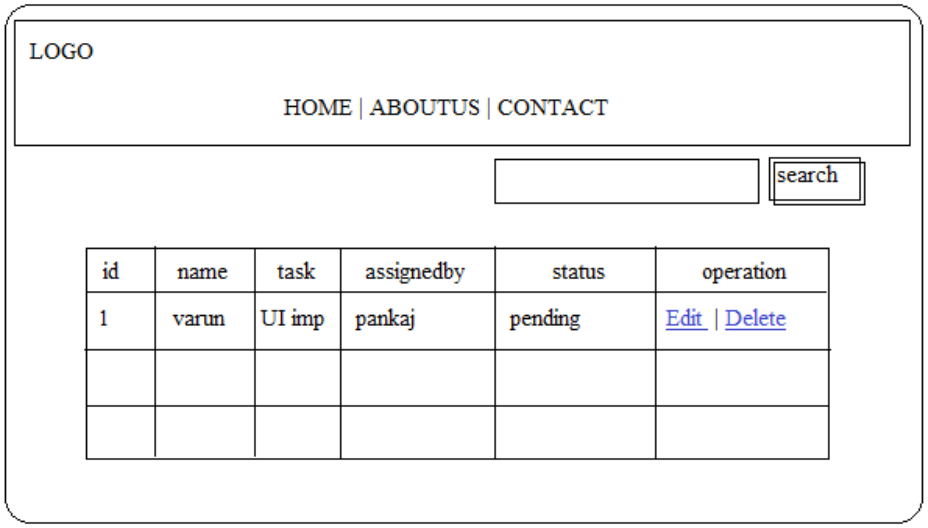
Topics to be covered

- What is React Component
- How to create React Component
- Rendering a component
- Extracting component
- Using props
- Data flow with props
- Practical Exercises on props
- Creating list in React with .map and .filter
- Practical Exercises on .map and .filter

What is React Component

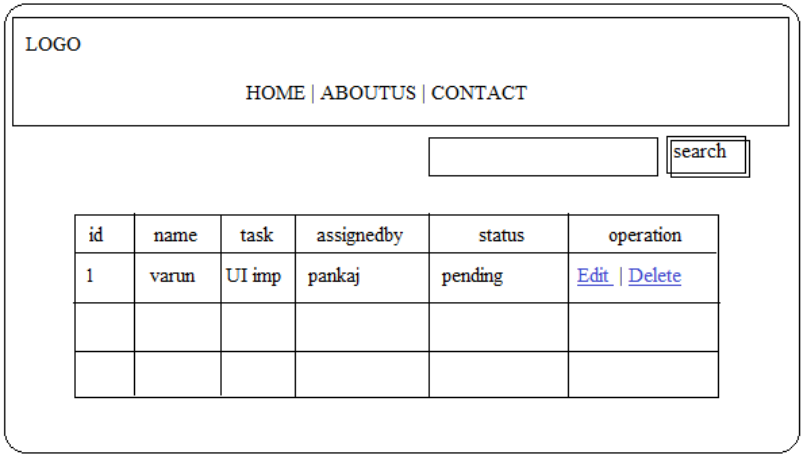
We will see the nature of React Component and will cover the details around creating React components.

- React Component is typically a single view of a user interface that is divided up into logical parts or branches.
- The tree becomes the starting component (e.g. a layout component)
- Each branch will become the sub-component that can be divided further into sub-components.
- Advantage will be the organized UI and it allows data and state changes to logically flow from the tree to branch, then sub branches.



How to create and render React Component

We will be creating below structure. UI will be latter improved.




```
Hello-app/src/index.js
-----------------------------------------------
import React from 'react';
import ReactDOM from 'react-dom';
class IssueContainer extends React.Component{
render(){
return <div>
          IssueContainer
        </div>
}
ReactDOM.render(<IssueContainer/>,document.getElementById("root"));
```

How to create and render React Component

Add Remaining Code

Hello-app/src/index.js

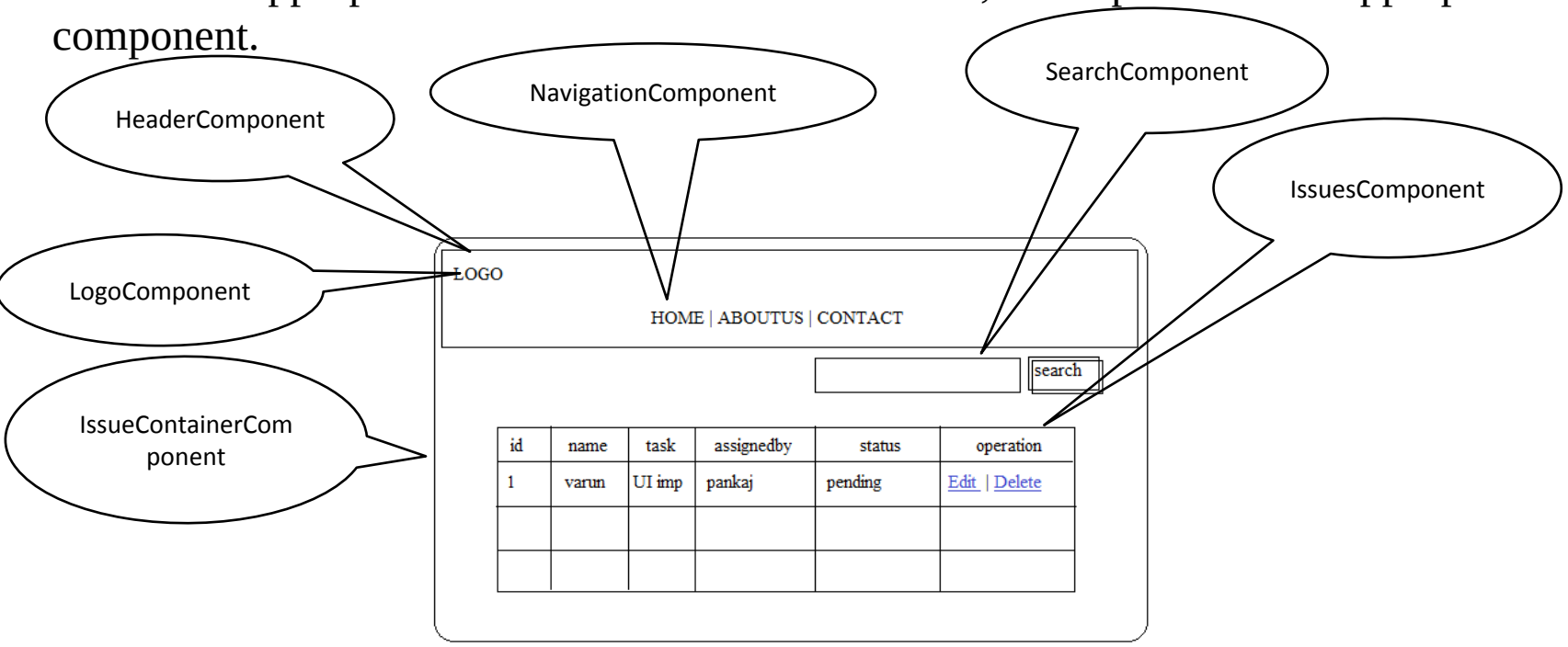
```
import React from 'react';
import ReactDOM from 'react-dom';
class IssueContainer extends React.Component{
          render(){
                    return (<div>
                              IssueContainer
                              <div>
                                 Header
                              <div>
                                  Logo
                              </div>
                              <div>
                                  HOME | ABOUTUS | CONTACT US
                              </div>
                              </div>
                             <div>
                              <form>
                                  <input type="text" placeholder="search site"/>
                                  <button>Search</button>
                              </form>
                             </div>

Continue.....
```

How to create and render React Component


```
Hello-app/src/index.js
-------------------------------------------------
<div>
          <table>
             <tr>
                <th>#</th>
                <th>Name</th>
                <th>Task</th>
                <th>Assigned By</th>
                <th>Status</th>
                <th>Opetation</th>
             </tr>
             <tr>
                <td>1</td>
                <td>Varun</td>
                <td>UI Improvement</td>
                <td>Pankaj</td>
                <td>Pending</td>
                <td><a href="">Edit</a> | <a href="">Delete</a></td>
             </tr>
          </table>

</div>
</div>
)
}
}
ReactDOM.render(<IssueContainer/>,document.getElementById("root"));
```

Extracting Component

Assignment : Visualize each part as a separate component. Create Separate component, extract the appropriate code from the IssueContainer, and replace it with appropriate component.



Extracting components might seem like grunt work at first, but having a palette of reusable components pays off in larger apps. A good rule of thumb is that if a part of your UI is used several times (Button, Panel,ProfileImage), or is complex enough on its own (App, FeedStory, Comment), it is a good candidate to be a reusable component.

Props & DataFlow with props

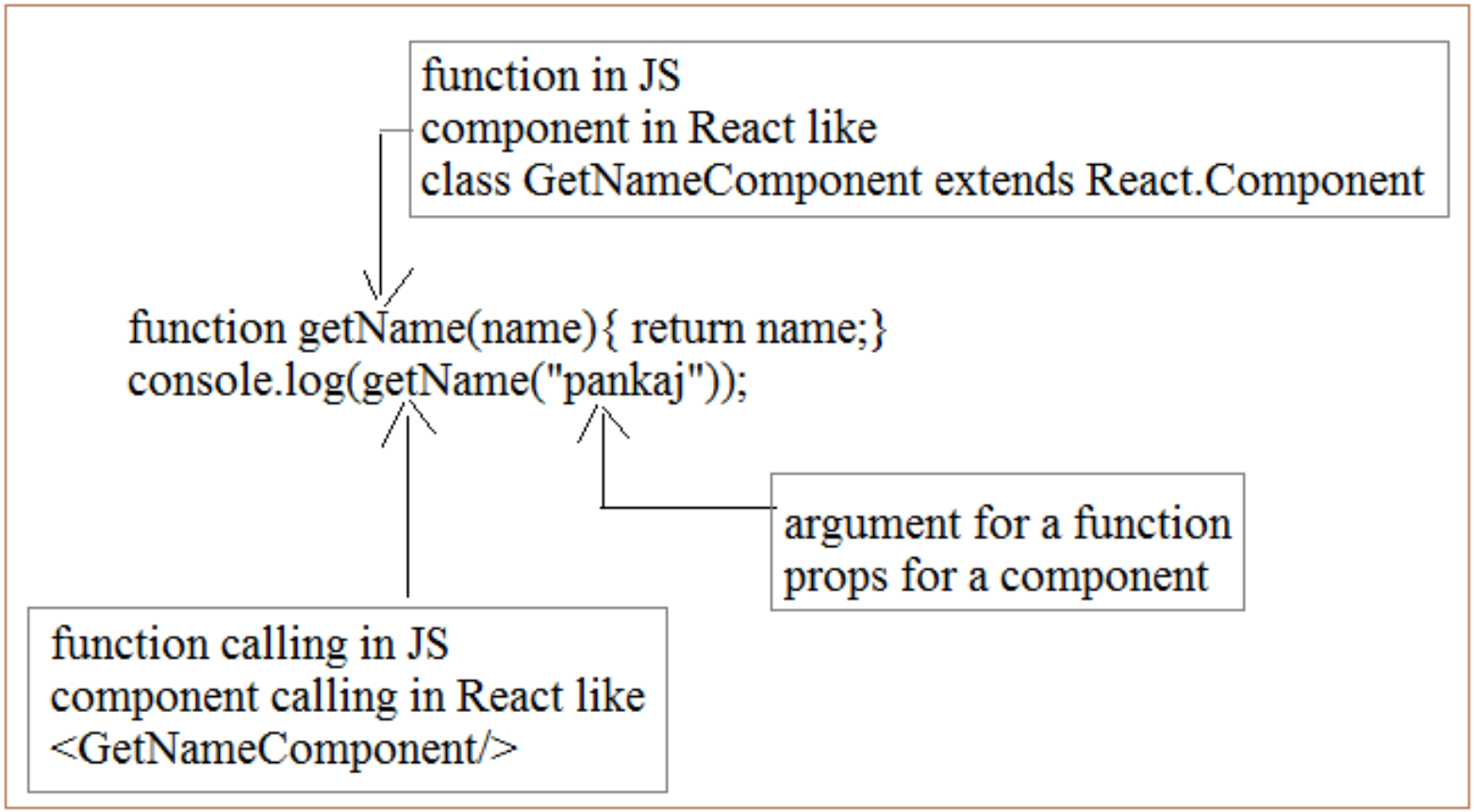
- We know how to create Component.
- Now we will see how to pass information to our Component.
- Passing information to component is just like passing information to function.
- React is good at managing state, because there is a simple system for passing data from one component to another child component and that system is through **“props”**
- **Props are to component what arguments are to function**

Assignment : Create a function in javascript browser console to accept name of the user and return the provided name, verify your function in console.log();

Solution:

```
function getName(name){ return name;}
console.log(getName("pankaj"));
```

Props & DataFlow with props



Props & DataFlow with props

Example of using props.
Create GetNameComponent, that should display the Hello! <username> as the name of user is provided.

```
propdemo/src/index.js
-----------------------------------------------
import React from 'react';
import ReactDOM from 'react-dom';

class GetNameComponent extends React.Component{
render(){
            return(
                        <h2>Hello ! {this.props.name}</h2>
                  );
        }
}

ReactDOM.render(<GetNameComponent name="Pankaj Sharma"/>,document.getElementById("root"));
```

When we use the component, we pass the name as attribute. This attribute can be accessed inside the component using ***this.props.name***

Practical exercises on props

Assignments 1 : Below are some of the assignments based on passing the correct props and rendering them.

Name : Pankaj Sharma

UserName : pankaj

propdemo/src/index.js

```
class Badge extends React.Component{
render(){
return(
<div>
<img src={this.props.img}/>
<h2>Name :{this.props.name}</h2>
<h3>UserName :{this.props.username}</h3>
</div>
);
}
}

ReactDOM.render(<Badge
name='Pankaj Sharma'
username='pankaj'
img='./profile.jpg'/>,
document.getElementById("root"));
```

Note : add profile.jpg in public folder.

Practical exercises on props

Assignments 2 : let us refactor the code as shown below.

propdemo/src/index.js

```
var USER_INFO={
name:'Pankaj Sharma',
username:'pankaj',
img:'./profile.jpg'
}

class Badge extends React.Component{
render(){
return(
<div>
<img src={this.props.user.img}/>
<h2>Name :{this.props.user.name}</h2>
<h3>UserName :{this.props.user.username}</h3>
</div>
);
}
}

ReactDOM.render(<Badge user={USER_INFO}/>,
document.getElementById("root"));
```

Practical exercises on props

Assignments 3: we want same output. This time you need to convert image, Name and UserName section in respective Components.

Note : try to use the props main component as well as sub components.

Practical exercises on props

Solution 3:

propdemo/src/index.js

```
var USER_INFO={
name:'Pankaj Sharma',
username:'pankaj',
img:'./profile.jpg'
}
class ProfileImageComponent extends
React.Component{
render(){
return(
<img src={this.props.img}/>
);
}}

class ProfileNameComponent extends
React.Component{
render(){
return(
<h2>Name : {this.props.name}</h2>
);
}}

Same for ProfileUserNameComponent
```

```
class Badge extends React.Component{
render(){
return(
<div>
<ProfileImageComponent
img={this.props.user.img}/>
<ProfileNameComponent
name={this.props.user.name}/>
<ProfileUserNameComponent
username={this.props.user.username}/>
</div>
);
}
}
ReactDOM.render(<Badge user={USER_INFO}/>,
document.getElementById("root"));
```

Creating list in React with .map and .filter

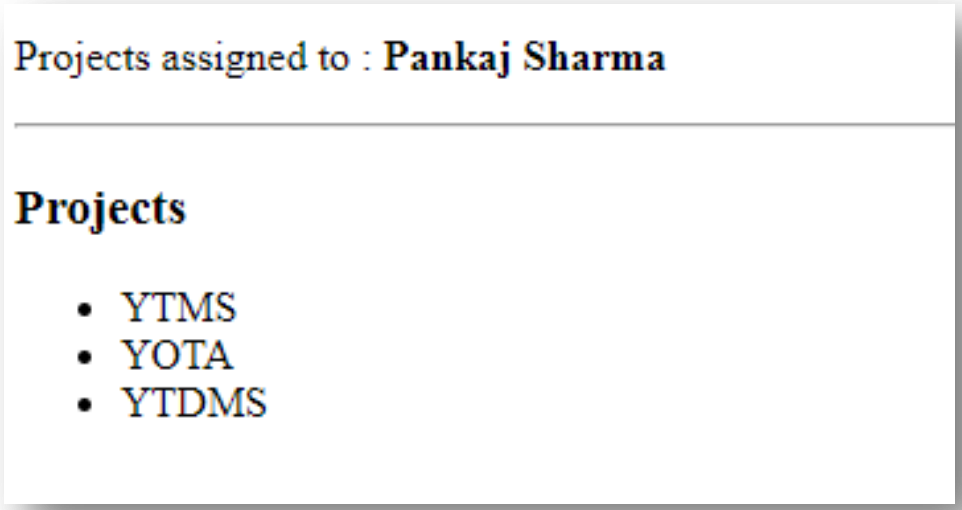
Assignment :
Create a basic javascript code that should add 10 in each number of the array number.
var numbers=[10,20,30,40,50];
Output must be:
20 30 40 50 60

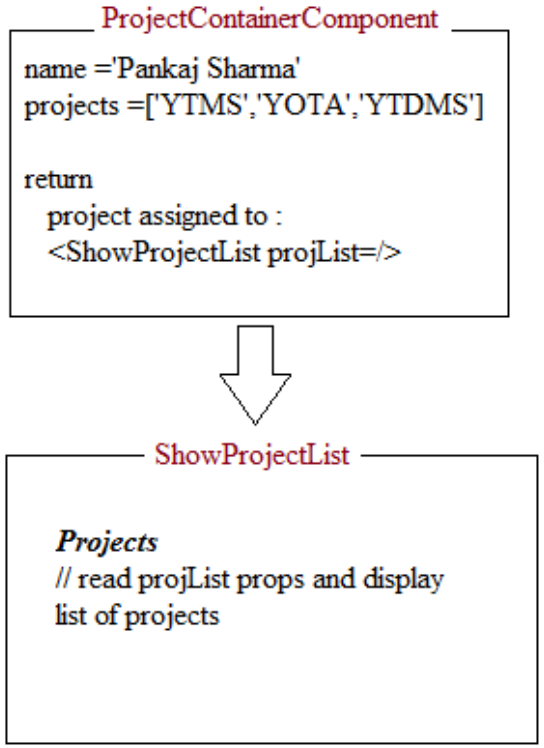
```
var numbers=[10,20,30,40,50];
var numberplusten=numbers.map(function (num){
        return num+10;
});
console.log(numberplusten);
```

Creating list in React with .map and .filter

Let's see how we can use map function to build a list for our UI in React. We will be having two components for this. One is parent component and one is child component. Parent will pass down the data to child as props.

Assignment : We want to show the list of projects that are assigned to manager.





Creating list in React with .map and .filter

propdemo/src/index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
class ProjectContainer extends React.Component{
render(){
var name='Pankaj Sharma';
var projects=['YTMS', 'YOTA', 'YTDMS'];
return(
<div>
<p>Projects assigned to : <strong>
{name}</strong></p>
<hr/>
<ShowProjectList projList={projects}/>
</div>
);
}
}
```

Continue. . . .

```
class ShowProjectList extends React.Component{
render(){
return(
<div>
<h3>Projects</h3>
<ul>
{
this.props.projList.map(function (project){
return <li>{project}</li>;
})
}
</ul>
</div>
);
}
}
ReactDOM.render(<ProjectContainer/>,document.getEl
ementById("root"));
```

Use of .map and .filter

.filter is same as **.map,** but instead of returning new array after you've modified each item in the array, **.filter** allows you to filter out certain items in an array.

For example we want all the Projects that Starts with letter 'Y', this will be the list of internal projects.

Before looking at React version, let us first see it in JavaScript.

```
var projects=['YTMS', 'YOTA', 'YTDMS','Ketterpiller','KK
Sons','Moci','Sigma'];

projects.filter(function(project){
    return project[0]==='Y';
});
```

Check it in console. This will return the list of all the projects starting with letter 'Y'.

Now let's check it in React.

Use of .map and .filter

propdemo/src/index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
class ProjectContainer extends React.Component{
render(){
var name='Pankaj Sharma';
var projects=['YTMS', 'YOTA',
'YTDMS','Ketterpiller','KK Sons','Moci','Sigma'];
return(
<div>
<p>Projects assigned to : <strong>
{name}</strong></p>
<hr/>
<ShowProjectList projList={projects}/>
<hr/>
<InternalProjectComponent projList={projects}/>
</div>
);
}
}
```

propdemo/src/index.js

```
class InternalProjectComponent extends
React.Component{
render(){
return(
<div>
<p><h3>Internal Applications</h3></p>
<ol>
{this.props.projList.filter(function(internalProje
ct){
return internalProject[0]==='Y'
}).map(function(intProj){
return <li>{intProj}</li>;
})}
</ol>
</div>
);
}
}
```

ShowProjectList component will be same as previous one.
ReactDOM.render() statement will be same.

How .filter is used with map

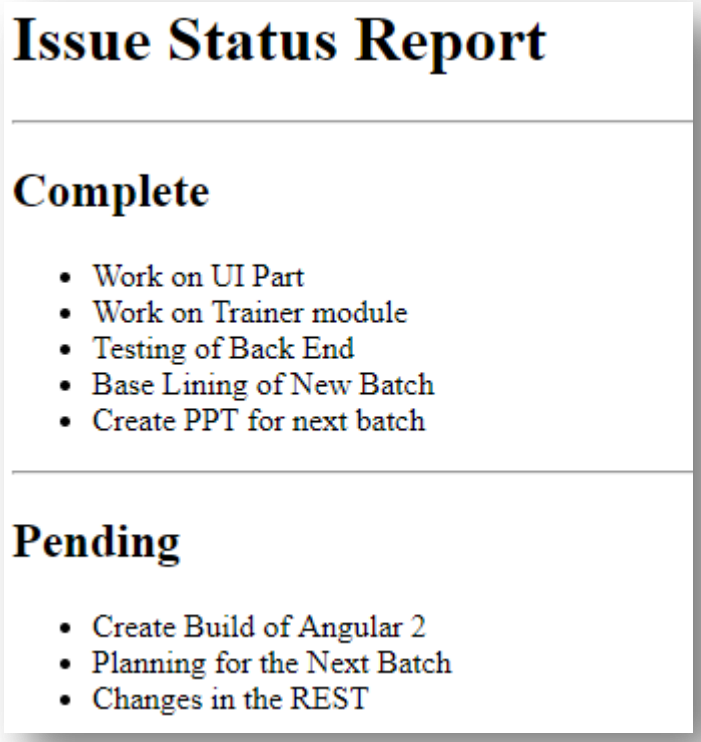

```
{
this.props.projList.filter(function(internalProject){
return internalProject[0]==='Y'
})

.map(function(intProj){
return <li>{intProj}</li>;
})}
```

This will return the projects or objects that satisfies the condition.

This will take one value or object from the list of values and return statement will prepare the list and display data.

Practical Exercise on .filter and .map

Assignment : Assume that you have a list of issues , some of the issues are completed and some are not. Issues can be represented as a list of Issue which holds two values as name of the issue and status of the issue. Name will be the string value and status will be the boolean value.



Practical Exercise on .filter and .map

propdemo/src/index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
class Issues extends React.Component {
render() {
return ( <div>
<h1>Issue Status Report</h1><hr/>
<h2>Complete</h2>
<ul>
{
this.props.IssueList.filter(function(issue){
return issue.status=== true
})
.map(function(issue){
return <li>{issue.name}</li>;
})
}
</ul><hr />
<h2> Pending </h2>
<ul>
{this.props.IssueList.filter(function(issue){
return issue.status=== false
}).map(function(issue){
return <li>{issue.name}</li>;
})}
</ul>
</div>)}}
```

Continue. . .

```
ReactDOM.render(
<Issues IssueList={[
{ name: 'Work on UI Part', status: true },
{ name: 'Work on Trainer module', status: true
},
{ name: 'Create Build of Angular 2', status:
false },
{ name: 'Testing of Back End', status: true },
{ name: 'Planning for the Next Batch', status:
false },
{ name: 'Base Lining of New Batch', status: true
},
{ name: 'Changes in the REST', status: false },
{ name: 'Create PPT for next batch', status:
true },
]}
/>,
document.getElementById('root')
);
```

Thank You!

Email: info@yash.com
Web: www.yash.com

Created by : Pankaj Sharma