**Day 1**

**React JS Training :**

**JavaScript : ES5**

**Web Application**

https://[www.google.com](http://www.google.com) URL

req(http/https)---🡪

Client Server

🡨---res(http/https)-----

HTML/HTML5

Web pages or Static web application.

CSS

**Actual Contents : HTML**

Cascading Style sheet :

Separation of concern

Presentation style : **CSS**

ES5 : ECMA Script : Concept.

JavaScript is a implementation of ES5.

JavaScript ES5: JavaScript was object based interpreter scripting language.

Object base Vs Object oriented

JavaScript was use to do validation on client side.

On html programming concept.

Variables

Data types

If statement

Looping

Function and events.

Pre-defined object

User-defined object

HTML 🡪 Content

CSS 🡪 Presentation

JavaScript 🡪 Action on content

CSS :

Inline CSS :

<tagName style=”property:value”></tagName>

Internal CSS

In head tag

<style>

Selector {property:value}

</style>

External CSS

ES5 :

JavaScript function syntax

JavaScript : **function and events**

Event : it is interaction between user and components(html tags p, h1, form, div, span, html form tags).

Event is providing bridge between html and javascript.

In JavaScript all event start with prefix

On followed by eventname

onClick

onDblclick

onMouseOver

onMouseOut

onKeyUp

onKeyDown

onSubmit

onFocus

onBlur

etc

DOM : Document Object Model : all html tag is known as DOM elements.

DOM provided set of API (Application Programming interface) which help to read, write and update dom elements.

jQuery : jQuery is a external JavaScript function which help to do DOM Operation.

Angular Framework : It is a framework which internally follow MVC : Model View and Controller. Google company.

React JS : React JS is a library not a framework. In MVC React JS use View.

React JS Application without Node JS

React JS Application with Node JS

Vue JS :

Node JS : Node JS is not a library not a framework. It is a runtime environment for JavaScript library or Framework.

Node JS provide lot of pre-defined module.

Few modules are local

Few modules we have to download.

With the help of those modules JavaScript become Server Side scripting language.

JEE

Spring boot

Asp.net

Python

Php

Node JS

MEAN Stack or MERN Stack or MEVN

MongoDB Express Module Angular/React and Node JS

Framework internally follow standard.

70 to 80% task taken care by framework.

Framework are heavy

Framework also known as Template or protocol.

Implementation of Design pattern is taken care by Framework.

Library :

Light weighted. Doesn’t follow any standard.

Doesn’t follow any design pattern.

**Npm : node package manager.**

Using npm you can download external modules provided for node js.

Express module : REST API using JavaScript

Db module : we can connect mongo db or my sql database.

Rest Full Web Service :

Backend Front end

Spring Boot Java

Php .net

Asp.net Angular/ReactJS

Python

Node JS : Express Module REST API

Service Provider : JSON /XML

Npm command.

Node JS program we can’t write document or window object.

**create-react-app**

create-react-app is a external module which help to create the create js program with sample code with all dependencies for that application.

npm : node package manager. It is like a mvn in Java. Npm help to download or install/uninstall any external module.

npm install moduleName : install locally

npm install moduleName –g : install globally

Now to create the react js program

Command

Open the command prompt

**node --vesion**

**npm --version**

**npm install create-react-app –g**

**or**

**npm install create-react-app**

This command is use to create the project

**create-react-app project-name**

After project created successfully

Move inside a project directory

**cd project-name**

To run the project

**npm start**

Please open the project in VSCode

**React JS mainly is use to create the SPA (Singe Page Application).**

**AJAX**

**MPA: The whole page loaded.**

**SPA :** we are loading only part of the web page rather than whole page.

JavaScript function

Basic JavaScript

function App() {

code …..

return variableName;

}

document.write(“Welcome”);

document.write(“<font color=red>Welcome</font>”)

ES5 React JavaScript Function (Component)

function App() {

code …..

return HTML Code

)

React JS using JSX : **JavaScript and Extension**

JSX :Inside a JavaScript we can write html and as well javascript code.

**Index.html View**

Intermediate file is

index.js

**App.js (Component )**

**Component : Component is use to control the view or part of the View.**

**In React JS We can create two types of Component**

1. **Functional style component**
   1. **Using normal function style**
   2. **Using arrow function style**
2. **Class style component**

**In class style component we can use life cycle methods. But in functional component style we can’t use life cycle methods.**

Whenever we use ES6 Features as well as JSX concept. It require transpiler which help to covert JSX to ES5 JavaScript code.

React JS use Babel Transpiler to convert ES6 as well JSX code.

In React JS we can use two type of variable.

1. State : it is private to that component. Which help to describe the component behaviour. State can change using setState() function/methods.
2. Props : it is use to pass the value from one component to another component like Data sharing. Props can’t change. It is also known as immutable property.

Constructor : it is a type of special function or methods. Which help to create the object.

Points.

1. In ES6/Typescript to write the constructor we have to use constructor keyword for that function.
2. It will call automatically when we create the object. But in React JS object creation taken care by Babel.
3. In React constructor is use to set props and initialization of state in Class component style.

**P**

**H1**

**Font**

**Using Angular as well React JS we are creating user-defined tag. So whenever we use user-defined background that logic we want to execute that code we can write using function or class component style.**

**Function Component**

**Function component takes are props as parameter (optional) and return the JSX (static and dynamic).**

**Class Component**

Class component contains life cycle methods, constructor, state and props.

Retrieve array data from component using parent and child relationship.

**Event with Login Page Example**

**Share the data between component – Parent – Child and Child – Parent Relationship.**

App

JSX

this.state.id =1

Employee Person

Id:100 id:10

this.state.id 100 this.state.id 10

JSX JSX

this.props.id 1 this.props.id 1

React Component Life cycle :

Life cycle divided into three parts.

1. Initialization phase : Only one time
2. Update phase : again and again when you do update state
3. Destroy phase : at last Only one time

**Refs : ref attribute is use to access the DOM elements values.**

**Creating the React Js project with typescript**

**create-react-app sample-demo –-template typescript**

**if you get the error in App.tsx file**

**open the tsconfig.json file**

 "jsx": "react-jsx"

Replace by

 "jsx": "react"

**Day 2 :**

**Load the json Data from static file**

**Call Fake Service using axios (URL ) Get Resources**

**Using fetch() purely JavaScript base upon the AJAX**

**Using axios third party plugin**

**Both method return Promise object.**

**Sample Rest Service API created using Node Express Module**

**To call Rest Service API for Get,Post, Put and Delete methods.**

**Redux**

**Action**

**Store**

**Reducer**

**Basic Operation**

**CRUD Operation**

**Routing**

**Using Path**

**Using Link**

**Using Button (programmatically)**

User Defined Promise Object

var obj = new Promise((resolve,reject)=> {

//resolve("Successfully Done"),

reject("Error Generated...")

})

obj.then(data=>console.log(data)).catch(error=>console.log(error));

console.log("Done1")

console.log("Done2")

console.log("Done3")

console.log("Done4")

console.log("Done5")

using fetch() and axios

Web Service :

HDFC HSBC

Java .net

Web Service : giving the Service for web application when both service running using different technologies.

SOAP Web Service : Simple Object Access Protocol :SOA

Consume and Produce only in the form of XML.

WSDL : Web Service Description Language. Only in the form of XML

Rest Web Service

We can consume as well produce data in any format XML as well as Non xml (JSON or text or steam format).

WADL : Web Application Description Language.

Swagger API : To create proper documentation for Rest full web service.

**axios:** It is third party plugin which help to call Rest API using get(), post(), put() and delete() etc.

We have to install this module

npm install axios --save

Move inside express module folder

**npm install**

**node allHttpMethods**

axios.get(“url”).then().catch()

axio.post(“url”,jsonData).then().catch()

axio.post(“url”,{}).then().catch()

axios.put(“url”,jsonData).then().catch()

axios.delete(“url/pathParam”).then().catch()

http: angular 2 return type is promise

HttpClient Angular 4 10 return type is Observable.

**Redux Concept**

A -🡪 Parent

M Component

React is library not a framework. By default React doesn’t follow standard.

According to MVC React is View

Redux is a state management tool.

With the help of redux we can keep the state in centralized place ie Store

Few component store the data in store, few component retrieve from store, few component update state data in store, few component delete the data from store.

Redux are three building parts

1. Action
2. Store
3. Reducer
4. State (Data)

Action : Simple Action are events. They are the only way we can send the data(state) from you component to Redux Store. Those event may be life cycle event or base user-defined event when interact with any html DOM elements.

Store : The store hold the application or component data(state). There only one store in Any Redux application. You can access the state stored, update, delete or retrieve the state (data) through helper methods.

Reducer : Reducer are pure JavaScript functions that take the current state of the application to perform an action of state(date) and return the new state(data) values. Where reducer is responsible to change the State(data) in store.

State: State which hold the value base upon the type of state variables.

Normal Database

State : Actual data id, name, salary

Store : Database (MySQL, Oracle, MongoDB)

Action : Insert/Delete/Update/Retrieve : Query

Reducer : Table

function dis(id,name,salary,...tech) { // REst operator ...variableName it must in function parameter, 0, 1 or many

console.log(id);

console.log(name);

console.log(salary);

console.log("Size "+tech.length);

}

dis(1,"Raj",12000);

dis(1,"Raj",12000,"C");

dis(1,"Raj",12000,"C","C++");

dis(1,"Raj",12000,"C","C++","Java",".net");

let techologies =["C","HTML","JS","oracle"];

dis(1,"Raj",12000,techologies);

dis(1,"Raj",12000,...techologies); //spread parameter

Without Redux

import React from 'react';

class App extends React.Component {

  constructor(props){

    super(props);

    this.state = {id:100,name:"Ramesh",salary:12000};

  }

  increment=()=> {

    //let salary = this.state.salary;

    //salary = salary+1000;

    //this.setState({salary:salary})

    this.setState({...this.state,salary:this.state.salary+1000})

  }

  decrement=()=> {

    //let salary = this.state.salary;

    //salary = salary-1000;

    //this.setState({salary:salary})

    this.setState({...this.state,salary:this.state.salary-1000})

  }

  render() {

    return(

    <div>

      <h2>update Salary with Redux</h2>

      <p>Id is {this.state.id}, Name is {this.state.name}, salary is {this.state.salary}</p>

      <input type="button" value="increment" onClick={this.increment}/>

      <input type="button" value="decrement" onClick={this.decrement}/>

    </div>)

  }

}

export default App;

Limitation of this id,name,salary are local this App component. If we want to do any changes on those state if have access from same component or else we can take the help of props only if two component are parent and child component relationship.

With Redux

First install the redux and react-redux

**npm install redux –-save**

**npm install react-redux –-save**

create the Store folder and create the **reducer.js** file

Now we have take help of Provider API which is part of react-redux which allow as to inject the Global store for the Higher Component ie App Component.

Then create the create the Store. To create the Store there is pre-fined function createStore() part of redux

Which we have to import in index.js

Two function which help to access the global data from a store

mapDispathcToProps

mapStateToProps

If you want to create the component using **CLI**

**npm install create-react-component**

**create-react-component component-name**

Web Service application : sessionId (Cookies)

JWT (Json Web Token)

JavaScript

1st

localStorage Backend unique Id

sessionStorage

sessionStorage.setItem(“key”,value);

sessionStorage.getItem(“key”)

sessionStorage.removeItem(“key”);

React

Context API React JS

Third party

redux store

**ES6 Features :**

let const,

types of loop

in loop, of loop

type of functions

arrow function

ES6 Class Features

**React Routing :**

SPA : all data display using components.

Routing is use to navigate from one component to another complete template (JSX) base upon the path provided in routing.

All routing API is part of react-router-dom

Step install react routing module

**npm install react-router-dom –-save**

To enable routing features we have wrap for main module as Browser module App.

**Route** : which display the component JSX details base upon the path pass by URL or hyperlink or button.

**Url Pattern :** This technique is use render the same component base on its url like and different values.

**Day 3**

redux-persist : storing the data in browser memory

redux-devtools : configure

Next :

MCQ : Test

**Redux-persist : this is external module part of redux which help to store the data permanently in browser memory.**

**Depending upon the type of storage.**

**Storage internally it use the localStorage.**

**Install the redux-persist**

**npm install redux-persist –-save**

**index.js**

import React from 'react';

import ReactDOM from 'react-dom';

import './index.css';

import App from './App';

import {combineReducers, createStore} from 'redux';

import reducer from './Store/reducer'   // single reducer

import {Provider} from 'react-redux'

import {persistReducer,persistStore} from 'redux-persist';

import storage from 'redux-persist/lib/storage'   //web storage internally use localStorage

import { PersistGate } from 'redux-persist/integration/react';  // import

const persistConfiguration = {

  key:"root",

  storage:storage

}

/\* const multReducer =      combineReducers({

                    employeeReducer,productReducer

        })\*/

const PersistReducer = persistReducer(persistConfiguration,reducer);

const storeInfo = createStore(PersistReducer);

const PersistStore = persistStore(storeInfo);

ReactDOM.render(

  <React.StrictMode>

    <Provider store={storeInfo}>

    <PersistGate persistor={PersistStore}>

          <App />

    </PersistGate>

    </Provider>

  </React.StrictMode>,

  document.getElementById('root')

);

**Redux dev tool** : It is a development tool which help to keep the track about your state change and action fire on that state in redux store.

**npm install redux-devtools-extension**

**Limitation of React JS Application**

React JS is a Client Side Rendering Library.

Angular JS is a Client Side Rendering Framework.

React JS, JavaScript, Angular, Vue, jQuery js.

JSX : Babel

TypeScript : typescript

If we want do server side rending.

Asp.net

Php

Python

Jsp

Next JS : Next JS is use to create the Server side rending and static web page without server less like tomcat, web logic etc.

Limitation of Client Rendering

SEO (Search Engine Optimization).

Server Side Rendering

1. Faster page experience
2. We can improve SEO

Node JS Server : http or Express JS

**Event driven architecture**

Callback and asynchronous concept

Normal Server : Tomcat : multithreading.

Thread pool : 1000, 10,000 etc

http/https -🡪req

Client Server html static or dynamic

Res(http/https)---

**React JS**

Features of next JS

1. Server side rendering
2. Typescript support
3. Automatic routing
4. Dynamic components.
5. File base routing

Next js sample project

CLI

create-react-app react js

**create-next-app next js**

**npm install create-next-app –g**

**npm install create-next-app**

**create-next-app project-name**

After created the project

Move insider a project

cd project-name

to run the project

**npm run dev**

To run the application we have to hit the command as

<http://localhost:3000>