**React**

JavaScript library which is used to make interactive user interfaces.

HTML for creating content

CSS for style content.

JavaScript for generating dynamic content/ generating live content/ for events also.

**Advantages of using react**

There are the following advantages

Increase performance.

Decrease the line of code.

Faster development.

**How does react work- react concept**

Component-based development library.

Component is a piece/ part of UI- theory wise.

Create components separately.

Every component has its own/separate HTML (JSX), CSS, and JS.



Component reusability.

Components can be used again on the same page and again but with different data.

Components cannot be dependent on each other

**Setup react project**

**Node:**

(Provides)It is a Java script environment/runtime.

Js runtime = running environment for JS.

**Note:** run JavaScript anywhere on PC without using any HTML.

Brower by default running environment for JavaScript.

Note: if we want to run JavaScript outside the browser, we need node JS.

Download and install node

>node –version

>node -v

**NPM (Node Package Manager):**

There are a lot of libraries, and React is one of them.

NPM is a tool that you get when you install node js to help you install, uninstall, and manage libraries in your project.

**NPX (ode Package eXecute):**

Sometimes don’t you want to download the tool and directly use the library, then use npx. Npx executes the package from the internet without even downloading to the pc

**Create React project**

>cd react project

>npm create-react-app myapp

>cd myapp

>npm start

>ctrl+c

>npm run build

**Folder structure**

**Package.json: -** just an overview of libraries

It is the heart of any JS project.

The configuration file of any js project.

Keep track of project information.

**Package-Lock \_JSON:**

Extended version of pakage.json

Information about dependencies/libraries in details

**Node Module:** contains dependencies

All dependencies mentioned in package.js are present in the node module folder.

Libraries are also called modules.

Modules are simply nothing but collections of function.

The library is nothing but a collection of modules

>delete node module folder

>npm install

**Public (Index.htm):**

Note: The public folder keeps assets.

Two components have a single js file.

**SRC Folder:**

Contains components.

Component is nothing but a function returning some html that’s all – coding wise.

**Guideline for component**

Comp file name should be started with capital & follow camel case.

The component name should be as component file name.



**Creating Component**

Component is nothing but a function returning some html that’s all – coding wise.

import './Header.css';

function Header() {

  return (

    <div className="header">

      <h1>this is header</h1>

    </div>

  );

}

export default Header;

Creating function by extension rafc

function return jsx (JavaScript XML) not html.

Should be one parent/wrapper element like div section main <> etc.

**JSX Guidelines-**

1. Attribute should be in camel case conventional

Like class= className

1. Tag should be closed
2. Self-closing tag like img and input should be closed.
3. To write java script inside jsx use {} brackets.

**Babel-**

This is a library & compiler which is responsible for compiling this jsx into java script.

Like <h1></h1>

Babel compile/transpiler this jsx into js

document. createElement(‘h1’)

**difference between js/jsx**

js is a programming language and jsx is way of writing java Script in terms of html.

**Styling Component**

1. **CSS file**

**Header.css**

.header {

  border: 5px solid green;

}

**Import CSS file**

import './Header.css'; //importing

function Header() {

  return (

    <div className="header">

      <h1>this is header</h1>

    </div>

  );

}

export default Header;

1. **Note:**
2. we can apply class name to child component and style in App.css too
3. App.css is only work for App Component and its Childs components.
4. **inline css**

import "./Header.css";

function Header() {

  return (

    <div className="header">

      <h1 style={{ backgroundColor: "yellow", color: "white" }}>// inline css

        this is header

      </h1>

    </div>

  );

}

export default Header;

**Style Priority**

Inline-highest

Separate CSS between App.js

separate>App.js

**Note index.css -global css**

We can write global CSS for whole application.

**Component export and import**

****