1. install
   npm install -g create-react-app
2. Create the project
   create-react-app <Project Name>
3. To run the project
   npm start
   This command will execute the project and the output will be displayed on
   localhost:3000 Port number 300

Project structure:
1. node modules: This folder contains the modules required for project
2. public: It contains the images which we use in the project.
3. gitignore: Whatever we don't want to push on the git repository, We make
   it's entry in this file.
4. package.json: It contains the dependencies required for the project.
5. package.lock.json: It contains the detailed list of dependecies.
6 src : In this folder we create the components.

1. React uses the virtual DOM.
2. When the page is loaded on the browser, some memory gets reserved in the RAM.
3. In this memory the html elements are placed in the tree like structure.
4. This memory is called as DOM.
5. DOM is created only once at the page loading time, at the time virtual DOM is also created.
6. Using this virtual DOM, internally some part of actual DOM get updated.
7. The process of identifying the difference between DOM and virtual DOM is called as diffing.
8. Using the diffing actual DOM get updated.

1. When we create the React project, by default 2 components are created.
2. App and Index.
3. index.js is the first file which get displayed on the browser.
4. Every component in React return the JSX code(Javascript+html)
5. Every component contains the return statements.
6. In React we can create the component using function as well as class.
7. Index.js is the top level component.

Creating the function components:
steps:
1. create a file in the src folder with the name as of component name.
2. In this file create the function which will return the html code(JSX)
3. This function will act as component.
4. The name of the file and the name of funbction must be same while
   creating the function component.
5. It is recomended to create the separet stylesheet file
   for every created component with the same name as of component.

6. To create the function component we should import React i that file as
   import React from "react"
7. After creating the function xomponent, we must export it, then only we can
   use it in other component.
8. Component name should start with capital case.

Class component:
1. create the .js file with name of component.
2. in this file import React and component

3. create the class with same name as given to file.
4. extends this class with Component.
5. Create the render() function in this class, this render() function should
   return the JSX code(html+js).
6. export the class so that it can be imported and used in other components..

Props:
1. It is used to pass the data from parent component to child component.
2. Component is the basemost class in React, but when display the component
   index.js is the base component.
3. We can add the user define component directly inside the index.js also.
4. When we add user defined component inside the App.js component then
   App.js becomes the parent component of user defined component.
5. We pass the paremeter from parent component to child component at the time of
   adding the child component.
6. props are read-only, child component can only read the data received from
   props.
   Example: refer Cart and Payment component.

State:
1. state is the component's own data.
2. state is initialized inside the constructor of the component only.
3. state can be updated using setState() method.
4. A component with some state is called as stateful component.
5. State data belongs to component while props data comes from parent component.
6. To access the state we make use of "this.state.<name>".
   eg. this.state.empNo
7. To modify the state we make use of setState() method.

   Example : refer Employee component

Conditional Renendering:
1. Whenever we want to display a perticular UI on the page depeneding on some
   condition, we make use of conditional renedering.
2. In this approach we create multiple components and display them
   based on conditions.
   eg. if login is successful then display Home component else display
      Login Component.
Example : refer MyApplication, Login, Home Component.

Data binding:
1. It is a flow of data between component and it's UI part(render function).
2. Data binding is of 2 types
   1. One way data binding
   2. Two way data binding

3. One way data binding is implemented using state.
4. In two way data binding data flows from both component to UI and UI to
   component.
5. Two way data binding is implemnted using state as well as event binding.

Hooks:
1. It is new concept introduced in React
2. Using the concept of hooks we can create the state inside the

function components.
3.Earlier we were not able to create the state inside the function components
4. To create the state in the function components we make use of useState() hooks.
5. Hooks are like functions.
6. Hooks are predefined functions in React.
7. useState() hook return 2 values, the first value is initial value of state and the second value is the name of function. Using this function we can modify the value of state.
8. Later we have to create the funcion with this name and we can call it on any event.
9. useState() hook is same as this.setState() in the class component.
10. Function components are used to minimize the code size.
 eg. If we implemnt the state and if w want to modify it, then we need to create the constructor, inside constructor we have to create the state. Then we have to creat ethe separate function to modify the state. On event we have to invoke it.
 But un function component, we can use useState() hook and no need to create teh separate function, only we declare the function at the event declaration.
11. We need to pass the initial value of state as parameter to useState() hook.

useEffect() hook:
1. Is is used to update the metadat of the page.
2. This hook takes adddional function as parameter.
3. In this function we write our logic to change the metadata.
4. This hook gets executes only once at the page loading time.

Higher Order Component(HOC):
1. When we want to reuse the logic of exsisting component, we make use of HOC.
2. HOC is a technique to create the new Coponent using exsisting component.

Steps:
1. Create the annonymous function component.
2. Create an annonymous arrow function. In this arrow function create the annonymous class This function should accept the exsisting component as parameter.
3. We can pass only annonymous function component as a parameter to create HOC.
4. HOC is annonymous function which accept new logic(which is declared in annonymous function component) as input and add exsisting logic to it and return new Component.
5. We create annonymous function which contains new additional logic which we want to add to exsisting logic.
6. We assign this annonymous function component to a variable. And we pass this varibale as argument to the Base component(component with exsisting logic).
7. We can create the new logic only using annonymous function component not using class component.
IMP:- If we want to pass the state data of one component as props to another component, it can be done only using HOC.
8. HOC internally uses inheritance.

Routing :
1. Routing is used to develope the single page application.
2. Using the concept of routing we can navigate from one component to another

component.
3. To implement the Routing in React we use below packages
    1. react-router
    2. react-router-DOM
    3. react-router-native
4. react-router package is used in any application(web/mobile)
5. react-router-dom is used in browser based application
6. reat-router-native is used for mobile app.

Installation: npm install -g react-router-dom

Steps to create Routing app:
1. Create the component which will participate in routing.
2. Import {Route, BrowserRouter} from react-router-dom in the index.js file
3. Declare the Routes for the components in index.js file.
4. <Route> element has 2 properties from version 6. 1st is "path" and second property is "element". In this element property we specify the component name or any html code which we want to display when the path is invoked.
5. Create the Navigation component.
6. In this Navigation component import {Link} from react-router-dom.
7. In the Navigation component we create the Link to different components.
8. There is "to" property in the <Link>, the value given to "to" propery must be same as name given in the path property at the time of declaring Route in index.js file.
9. Or we can simply reate the Link in any other component also. If we dont want to create the separate Navigation component.
10. After writting the navigation logic we need to write <Outlet/> element.
11. Import this <Outlet/> element from 'react-router-dom' in the file in which we declare navigation logic.
12. This <Outlet/> element displays the invoked component(or html).
13. We need to create the Routes in the index.js file under <BrowserRouter>
14. Under this <BrowserRouter> we need to create <Routes> element, and under this <Routes> element we create the routes using <Route> element.
15. To use these we need to import {Route, BrowserRouter,Routes} from react-router-dom in the index.js file

Example : refer Cart, Admin, BankAccount

Child Routing(Nested routing) :
1. In Application we group the related components together.
2. To implement among the parent child component we implement child routing
3. When we want to create the Child route we declare the route of child under the parent's <Route> element.
    e.g.
```
<Route path="product" element={<Product/>}>
    <Route path="book" element={<Book/>}/>
    <Route path="mobile" element={<Mobile/>}/>
</Route>
```
Here Book and Mobile are the child routes of Product. It means when user will click on localhost:/3000/product , then only the links for these child routes will be displayed.

4. When we create the <Link> the "to" property shiuld start with parent route path followed by child route path.
    eg. <Link to="/Product/book">Book </Link>
    here /Product is path of parent route and /book is path of child route.

Passing parameter to child routes:
1. In react we can pass parameters to child routes.
2. To pass the parameter to the child route we declare that route as child route under the roure of parent.
3. IMP: When we want to pass the parameter to child route component, in this case that child route component should be function component. Because we acess these parameter passes with route using useParams() hooks. And we can use any hooks only inside function component.
4. While declaring the path for such a child component which accept the parameter we have to create the path starting with only parameter name.
   eg. <Route path=":title" element={<Productdetails/>}>
5. While creating the <Link> for such child we have to create it as starting with parent route path followed by child route path. and followed by the parameter to be passed. Here we have to combile the string path and value. Here we make use of JSX to create the <Link>
6. To access the parameter coming with route, Inside the child components we need to import the useParams() hook from "react-router-dom"
7. We access the parameter using useParams() hook as below
   let params=useParams();
   <h1> Parameter received as : {params.title} </h1>
   here id is name which we have given in the path at the time of creating <Route> for child. The name used at both places must be same.

Forms:
There are 2 types of forms in React:
1. Controlled forms
2. Uncontrolled form.

Controlled form.
1. This type of form manages the state of form.
2. It manages the user input in the component.
3. This type of form uses state and setState() to manage the state of form.
4. setState() is triggered using event handller.
5. Every state chage will have associated handler function.

Steps:
1. Create the component
2. Create the state in the component as per requirement(eg. name,password)
3. Create the <form> with required form controls
4. create the appropriate event handlers.
5. Bind the event handlers in the constructor using bind() function.
6. Call these appropriate handlers from perticular form control.
7. Execute.

Validation steps:
1. In case of form validation we display perticular messages when condition is not satisfied.
2. To display these error messages, we create the error object as state.
3. In this error obect we create the different state variable which will hold the perticular error messages. eg. errorOfName,cerrorOfPassword.
4.

Uncontrolled form:

1. In controlled form form data is controlled by react component
   with the help of event handlers.
2. Because of this length of code may increase, due to creation of
   multiple event handlers.
3. To overcome this issue, we make use of Uncontrolled form.
4. In the uncontrolled form we retrieve the form values using
   DOM

5.To retrieve the form values from DOM we use "refs" keyword.

Axios:
1. We can share the data within component using data binding.
2. We can share data among multiple components using services.
3. Axios is open source third party api.
4. Axios provides HttpClient service(request,response) to the component.
5. Axios is promised based api.
6. Axios uses XMLHttp request for communication.
7. This XMLHttp request is asynchronous requst.
8. Promise object has 2 outcomes
    1. success(Resolved)
    2. failure(Rejected)
9. installation:
   npm install --g axios --save-dev
10. In the component in whihc we want to use axios, we need to import axios
11. We create the axios obect, using this object we can call different
    http methods as
    axios.get(url)
    axios.post(url)
    axios.delete(url)
12. As axios is promise based we use axios as below

```
axios.get("www.json.com").then(response)
{
    here we will write logic.
    This part will be executed only if request is successful
}.catch(error)
{
    Here we will handle the error.
    This part will be executed only if request fails.
}
```

Testing:
1. Unit testing: It breaks the code into small parts.
2. The unit testing is generally done by the dvelopers.
3. In unit testing we check for the business logic and functionalities.
4. To perform the unit testing in React we use JEST
5. installation:
   npm install jest

Steps:
1. create any .js file whose functionality you want to test
2. Create the test file as <file_name>.test.js
3. Create some function in .js file and export it.
4. in .test.js file import the exporeted function using require.
5. in .test.js file we use test() function for writingtest cases.
6. Inside this test() function we use expect() function and other function such
   as toBe() or toHaveContain() or toHaveProperty() etc.

7. test() function has 2 parameters. 1st is message and 2nd is callback function.
8. In expext() function we call te function which we want to test.
9. The on expext() function we call toBe() or other result matching functions.
10. To run the test cases we use npm test command
11. toContain() function is used for array testing.

Sometimes we need to nned to perform some tas before start or end of testing
or before and end of indivisual tast case.
It is done using below function:

1. beforeAll(): calls the callback function before any test case
2. afterAll(): calls the callback function after all test case
3. beforeEach(): calls the callback function before each test case
4. afterEach:() calls the callback function after each test case