Bootstrap :→ A css frame work. with predefined libraries, which needed to be included with our code and they are ready to use.

Overwriting of codes requires advance knowledge.

So for fast-work we prefer bootstrap.

It's used to avoid cumbersome css code.

Vanilla css :→ Writing everything on your own.

Tailwind css :→ utility framework where we can give our twist to the code.

Bootstrap
It's also used to build responsive mobile-first sites

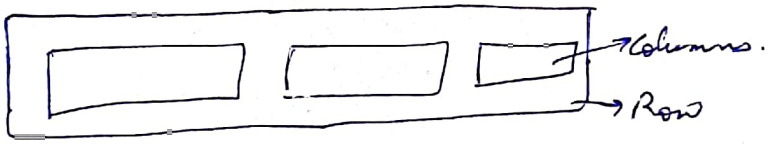
mobile-first means layout changes with change in size of the screen.
Note :→ mobile → moving here

It covers 3 things

① General Default style → Reboot (set cross-browser), Typography (font size & family) weight
② Pre-defined css, classes → Default responsive style
③ Grid layout.

Grid layout :→ We can think of our page as rows and columns which are defined by bootstrap.

suppose for different screens we want different layout of Columns, we can define it easily using bootstrap.



Pre-defined css classes.

eg: alert messages :→ with different bg colors.

buttons :→ In various colours

forms :→

Navigation bar :→

card/images :→

How to use Bootstrap :→ Go to → getbootstrap.com → Click get started → on that page we'll

1) get instruction how to add it to our project, the easiest way is by using CDN links

1st way.

CDN → highly fast / often importing from online servers

CDN is faster than your local servers so CDN is preferable.

Agile methodology.

↓

Agile → means fast.

scrum-master

1) What problem it solves.
2) Who is going to use it.

MERN MEAN

1. Visibility of System Status
2. Error ~~detection~~ Prevention
3. Recognition rather than recall
4. Aesthetic & Minimalist design
5. Ease of use
6. User-Control and freedom
7. Consistency & Standards
8. Match between System and Real world
9. Help and documentation
10. Flexibility and efficiency of use.

Foundation and

✓ Vanilla CSS

⇒ Traditional CSS, where we write codes on our own

→ Everything in our hand

→ Takes time

→ Full control

→ No unnecessary code.

Bootstrap

→ components framework.

→ We save time

→ takes space/memory

→ fast/Rapid development.

→ No need to be a CSS expert.

Tailwind CSS

→ utility libraries

→ We can add our twist too.

→ Fast development.

→ No need to Knowledge on scratch.

Website

→ Search → Restaurant ~~food~~ food

→ Buying ~~Ordering food~~ ~~category~~.

→ Choosing ~~specification of~~ spices

Digital Menu-card.

↳

Pros advantages

→ Lot of work

→ Danger of writing bad code

→ Slow

Pros

→ No overriding over code (It is possible but requires advance knowledge).

→ Unnecessary code/memory consumption as you are importing whole library and using just the part of it.

Pros

→ Unnecessary code

→ little control as compared to Vanilla but more control w.r.t Bootstrap.

	Specialities	Drawbacks
→ Javascript	—	Long codes for small tasks
It's a framework → JQuery	—	Not scalable
It's a framework → Angular JS	—	Not scalable, Not migratable
It's a Library → React JS → Scalable, easily migratable, easy maintainance, Faster Rendering (virtual dom)	Provides Stable Code.	—

Front end/Client Side :→ The part to which the user interacts directly.
Frontend could be languages, frameworks, libraries.

Back-end/Server Side :→ It stores and arranges data and make sure everything works fine on the Client Side.
Backend could be languages.

NOTE :→ Javascript is used as both front-end as well as backend.

React is developed by and maintained by facebook.

↓

It's a javascript library for building user interfaces using Component-Based Architecture.

It's a library not a framework.

Module is a small part of Package.

Keep your code DRY [Don't Repeat yourself]

The degree of freedom a library or framework gives to the developer is called "Opinioned"

Node JS is a platform.

Package manager means from where you can easily take/install new package or make your own and give it to the community to use

eg: Anaconda is a distributer as well as Package manager as you can easily install any new package from Anaconda

packages & dependencies.

Package are small part of library they are i.e they are also reusable codes.

Library	Framework
→ Pre-written codes by someone	→ Pre-written codes by someone
→ Developer are in charge of the flow when to use, where to use where to call the library	→ The framework is in charge of the flow, you just your code where you want but the call of code is in hands of framework.
eg:→ ~~javascript~~/ JQuery we used to call it at the end of HTML code and it looks	

dependencies :→ They are again Packages but they require another package on top of them to work

npm is the package manager for nodeJS

npm → node Package Manager

npx → node package executive.

It comes with npm only.

Environment :- Node Js is a Javascript runtime Environment.

↓

- ⊙ Text Editor :- Where you write your code
- ⊙ Interpreter :- Converts program line by line in Machine language
- ⊙ Compiler :- It converts whole program in one go.

You require the above 3 to create an environment for your work.

Similarly to work with library like React Js we require and environment like Node Js.

3tier Architecture :- How normal environment works

⊙ User or Client → ask → PHP or server side languages → sends request → computer file system

② PHP or server side language waits while the file system opens and reads the file.

③ Returns the Content to the Client

④ Ready to handle another request

Both are 3 tier but Node Js works fast and for React to work properly Node Js is must

How node Js environment works

⊙ Client → task sends request → ~~Node Js~~ Node Js → File System

⊙ Ready to Handle next request. i.e Node Js environment eliminates waiting and continues without rest

When the file system has opened and read the file, the server returns content or data to the Client

Hence Node Js is fast.

#ReactJs or React or React.js → It's a javascript library for building User-Interface.

⇒ It's a component based library to design UI.

⇒ Component based approach
⇒ Declarative approach.
⇒ DOM updates are handled easily.
⇒ React is Scalable kyoke app ek chaey kar component banake use karne hoo.

1) Pre-requisite → HTML, CSS, javascript, ES6 features.
Basic understanding using npm

→ VS-code-icon 'javascript ES6'

We can split any webpage into components.

⇒ A react component is just the function.

#React Dev tools :→ It's a Chrome devTools extension for the React Javascript library. It allows us to inspect the react component hierarchies in the Chrome Developer tool.

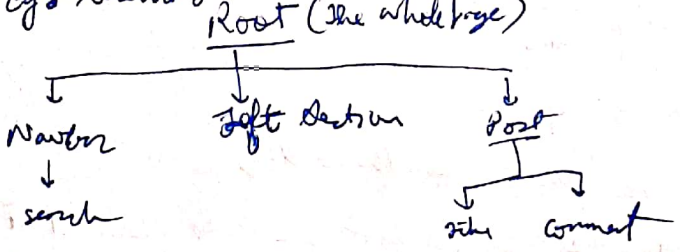
✓ extension hai jo batata hai kii page mai React use hua hai ya nhi or hua hai toh component kii heirarchy kya hai. → (Isse kya benefit hoga? How we use it) ?

JSX [Javascript Syntax extension or Javascript Extension] → this is neither a String nor an HTML.

→ It is somewhat like a template language but it comes with full power of Javascript. JSX produces react elements. # ? can we use JSX as extension

Component Heirarchy

eg:→ Facebook eg.



* We draw our own component Hierarchy when we make our project.

Project Process
1) Style guide
2) Wireframe
3) Component Hierarchy
4)

HTML file will contain
only these two things in the body
i.e.
<noscript> you need to enable javascript to run this app</noscript>

# index.js
import React from 'react';
import ReactDOM from 'react-dom';

In Command prompt
C:\> create-react-app <project name>

package.json :→ Contains the meta data about our project.

1st go to that project in command line then.
npm start [To start our development Server] Q) Why to Run cmd prompt as administrator?

Babble → It is modern Javascript compiler which browser can understand.

Simple javascript → when we use var React = require("react"); [with this command babble automatically comes]
or import React from 'react'; → (modern javascript ES6 code.)

#
ReactDOM.render (kya dikhana hai, kaha dikhana hai, composite funct^n)
eg: method is a part of React-DOM
ReactDOM.render (<h1> Hello World </h1> , document.getElementById("root"));

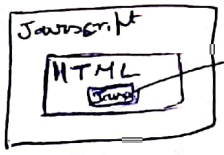
↑
Javascript expression.
or
JSX / Javascript XML

When you want to use JSX
React importing is compulsory.

 1,2,3
OR use [ ]

if version of react is greater than 16. something
.render method takes only 1 JSX element at a time so if we have to write multiple JSX element we must enclose our element in a single .....

#JSX



{ } → to use javascript again in HTML we need to use curly braces.

eg: index.js → file name

```
import React from 'react';
import ReactDOM from 'react-dom';
const Fname = 'Todesh';
ReactDOM.render (
    <h1> My name is Fname <h1>,
    document. getElementById ('root'));
```

Output on Screen → My name is Fname

because we are using javascript again in HTML like this i.e JSX.

So

```
import React from 'react';
import ReactDOM from 'react-dom';
const Fname = 'Todesh';
ReactDOM.render (
    <h1> My name is {Fname} </h1>,
    document. getElementById ('root')
    );
```

Now Output → My name is Todesh

Note :- { } → In these curly braces we can only use expression → something which returns single value like Math. random (), 5+2, 3×2 etc. but not Statements like if else or loop etc. Remember this

#Template Literals in JSX

Where we can use strings and javascript variables or expressions together.
Use backtick (` `)

eg.

```
import React from 'react';
import ReactDOM from 'react-dom';
const Fname = 'Todesh';
const Lname = 'Shaikh';

React.render (
    <>
    <h1> ~~my name is~~ {`My name is ${Fname} ${Lname}`} </h1>

    </>,
    document. getElementById ('root')
    );
```

React Fragment

```
import React from 'react';
import ReactDOM from 'react-dom';

ReactDOM.render(
  <React.Fragment>
    <h1> Hello Todeeb </h1>
    <h2> yes Nipun </h2>
    <p> OK lets begin </p>
  </React.Fragment>,
  document.getElementById("root")
);
```

No need to write React.Fragment too 😊

So use React.Fragment because it is little faster plus takes less memory compared to div or array.

or

now if we inspect we'll get.

```
<div id='root'>
  <h1> Hello Todeeb </h1>
  <h2> yes Nipun </h2>
  <p> OK lets begin </p>
</div>
```

Note ↑ no extra coming.

```
import React from 'react';
import ReactDOM from 'react-dom';

ReactDOM.render(
  <div>
    <h1> Hello Todeeb </h1>
    <h2> yes Nipun </h2>
    <p> OK lets begin </p>
  </div>, document.getElementById("root")
);
```

☆ When you write the above code and inspect the web page you'll notice this

```
<div id='root'>
  <div>
    <h1> Hello Todeeb </h1>
    <h2> yes Nipun </h2>
    <p> OK lets begin </p>
  </div>
</div>
```

↙ div inside div

That is an extra div comes in the root div which will be problematic so avoid this issue we use React dot Fragment which is given above the line

Problem will be when we use grid with bootstrap, it creates problem.

JSX

Uses camel case when we two words are there

① Page automatic Refresh ?

① ./style.css ko ese kyo likha poora path nhi diya.

PROPS

State.

means "properties"

It is used to pass data from Parent to Child

They are like arguments of a function. props are immutable i.e we can't change them

State

But we need something mutable

So state is something which can be edited.

stateless or Functional Component

↓

Those static component which will be just showed on the screen. they wont interact or actions.

They are also called as pure components i.e they just show data, take data from parent but just show them statically.

i.e Read only time.

(Sir ne joo karwaya)

Statefull Components / Class Components

↳ They are dynamic ES6 class.

(Aapane joo karwaya)

Props (Properties)

Basically like C++ Classes and Objects

Here we create our own customised components and attributes and using props we call them in the child with help of dot (.) operator.

eg:
```
import React from ...;
import ReactDOM from ...;
import Card from '/...';
ReactDOM.render(
  <>
    <Card name='sohail'
          Rollno='52'
          img='----'/>
    <Card name='Todeeb'
          Rollno='82'
          img='----'/>
    :
  </>,
  document.getElementById('root')
);
```

parent

~~export default Card~~

```
function Card(props) {
  <div className=>
    {props.name}
  </div>
  <h2>{props.Rollno}</h2>
  <img src={props.img} alt="" className="" />
}

export default Card;
```

Child where we call objects

#JSX attributes :→

eg: In HTML
lang
Class
Id
Style
contentEditable
etc

but in JSX we follow camelCase when two words are there
eg:
contentEditable

#CSS in React.

① using external CSS, internal CSS and inline CSS
② using Bootstrap.

#Google fonts

go to Google fonts, Copy fonts link paste in HTML file of ReactApp.
and
copy CSS of google fonts paste it in css file of reactapp
where every you want to use

#Inline CSS in React

you need to create objects for using it. and each object should be in {{}} and camelCase for 2 words.

eg:

```
const heading = { color: '#fa9191',
                  textTransform: 'capitalize',
                  margin: '70px 0',
                }
```

note: Here , is there in css we used to but ; and ' ' everything is in double or single quotes.

How to use

✓ `<h1 style = {{color:'#fa9191', textTransform:'capitalize'}} >`

or

✓ `<h1 style = {heading} >`

both works.

#Import & Export in ReactJS.

default → could be used only for one function or variable or Object

Developing a project.

Traditional website

Data comes from server
when we move from one page
to another i.e. hard reach

Note Flipcart is an isomorphic Application

But with Single Page Application → It is very good for small Projects but for big project we need another technology with it. i.e

Isomorphic Application are also Single Page Application but just their extended version. It's majorly used now.

Drawbacks with single Page Application

1st Performance issues.
2nd Lacks search engine Optimisation (SEO)
3rd Non-scalable.

Isomorphic Application

It is a combination of Traditional + Single Page Application.

i.e Server and client side both will have JSON file.

i.e Advantages → ① Better SEO
② Better performance
③ Better maintenance

NextJS :- Application framework.

used with react library for better UI.
so we get its environment too.

#Javascript Arrays :→ Something which holds multiple values of some type.

1st way

```
<script type='text/javascript'>
    var friends = ['Naman', 'Jeel', 'Nipun'];
    document.write(friends[2]);
</script>
```

Output

Nipun

String multiple values.

2nd way

```
<script type='text/javascript'>
    var friends = new Array('Naman', 'Jeel', 'Nipun');
    document.write(friends)
</script>
```

Output

Naman, Jeel, Nipun.

#

map() method :→ The map() method creates a new array with the results of calling a function for every array element.

or

It creates an array by calling an existing array.
without affecting the existing array.
So it is mostly used as an alternatives to loop.

Syntax:→

```
array.map(function(currentValue, index, arr), thisValue)
```

- array → existing array jisse loop lagana hai
- function → callback func.
- currentValue → existing array value
- index → Some arguments
- arr → Optional which array obj you are calling / mapping.
- thisValue → our value could be made and called

map() method continuation 3⇒

like if we are writing in javascript format

then

```
<script>
const oldarr = ['sheikh', 'Muhamm', 'Todeeb'];
console.log(oldarr[0]);
console.log(oldarr[1]);
console.log(oldarr[2]);
const newarr = oldarr.map(function(cvalue, i) { return i+":"+cvalue });
    console.log(newarr);
```

Output
'0 : sheikh'
'1 : Muhamm'
'2 : Todeeb'

now array of object.

```
<script
const studentdata = [
    { id: 1, name: 'Todeeb', degree: 'HighSchool' },
    { id: 2, name: 'Noman', degree: 'HighSchool' },
    { id: 3, name: 'Nijum', degree: 'Masters' },
];
console.log(studentdata[0].name);
console.log(studentdata[2].degree);
```

→ Output: 'Todeeb'
→ Output: 'Masters'

// the above method requires cumbersome process to print every data so to avoid that
// we use map() method using arrow function

```
const newdata = studentdata.map( (cvalue) => {
    return `My name is ${cvalue.name}`;
});
console.log(newdata);
```

(it is Backtick)

→ Output:
My name is Todeeb
My name is Noman
My name is Nijum.

Or

```
return `My name is ${cvalue.name}. My degree is ${cvalue.degree}`;
```

notice this

Output:
My name is Todeeb My degree is HighSchool
Noman "
Nijum " Masters

```
console.log(newdata);
</script>
```

For writing the Above thing in html

we can use

```
<p id='showdata'> </p>
document.getElementById('showdata').innerHTML = newdata;
```

key

A Unique value to every object

key help React identify
which items have changes (added/removed/re-ordered)
TO give a unique identity to every element inside the array, a key is required.

So
key is not a prop.

if - else in React

written inside function

eg:

```
const favSeries = 'amazon';

const favS = () => {
    if (favSeries === 'netflix') {
        return <Netflix />;
    else {
        return <Amazon />;
        }
    };
```

Ternary Operator.

Condition ? Col : Col2

Condition → if true → return this
else return this

Array Destructuring in ES6 in Javascript

So the destructuring assignment syntax is a javascript expression that makes it possible to unpack values from array, or properties from objects, into distinct variables.

Earlier How we used to do

```
<script>
const myproglang = ['js', 'php', 'c', 'python', 'java'];
//es5
  var top1 = myproglang[0];
  var top2 = myproglang[1];
  var top3 = myproglang[2];

  console.log(top1); ⟶ Output: js
```

} ⟶ So these many line got compressed in one line in ES6 i.e

```
//but in es6 now how we do
let [top1, top2, top3, top4, top5] = myproglang;
console.log(top1) ⟶ Output: js

//now to print 1st and last element of array using es6.
let [top1, , , , top5] = myproglang;
console.log(`My fav language is ${top1} and ${top5}`)
```

↙ backtick i.e. template literals where we can use strings and jsx together