

MERN Development Boot Camp

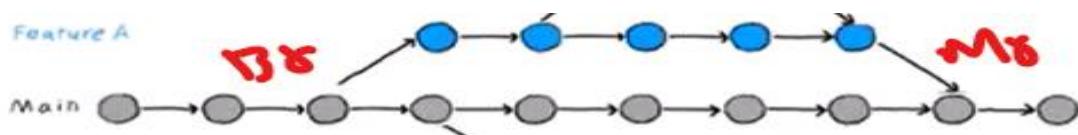
Module # 01 GIT

Git: Version Control System used to track changes to the file and facilitate collaborative development.

- 1) git init (to track the changes)
- 2) git add. (run it before commit)
- 3) git commit -m "Message"
- 4) git log (to see your commits)

Branches: series of commits.

we create branch if we are working in a team and both the members are working on different functionalities such as Singin/Signup so we create a branch signup and the signin will be implementing over the main then in the end we have to merge the branch into the main.



- 1) git branch feature/signup (create branch for signup)
- 2) git checkout feature/signup (to go to the branch)

Merging the Branch to main:

1st need to identify the destination of merge as we have created a branch for signup and now, we need to merge it with the main (master) so from master we need to run a command:

- 1) git merge feature/signup (so it'll merge the feature/signup code into the master)

The staging area is where we do stage so the file will be ready for the commit means we have a box in which we are collecting the changes which are ready for commit known as staging.

Let suppose we have to make a partial commit means we have two files and we need to commit just one file so how we can do that:

by staging the file we can achieve this as commit require staging that's why we use "git add ." it means move all the files (all files in which we have made changes) to the staging area so the we can make a commit but if we use "git add main.txt" so for now only main.txt will move to the staging and ready for commit so when we make a commit only main.txt will able to make a commit.

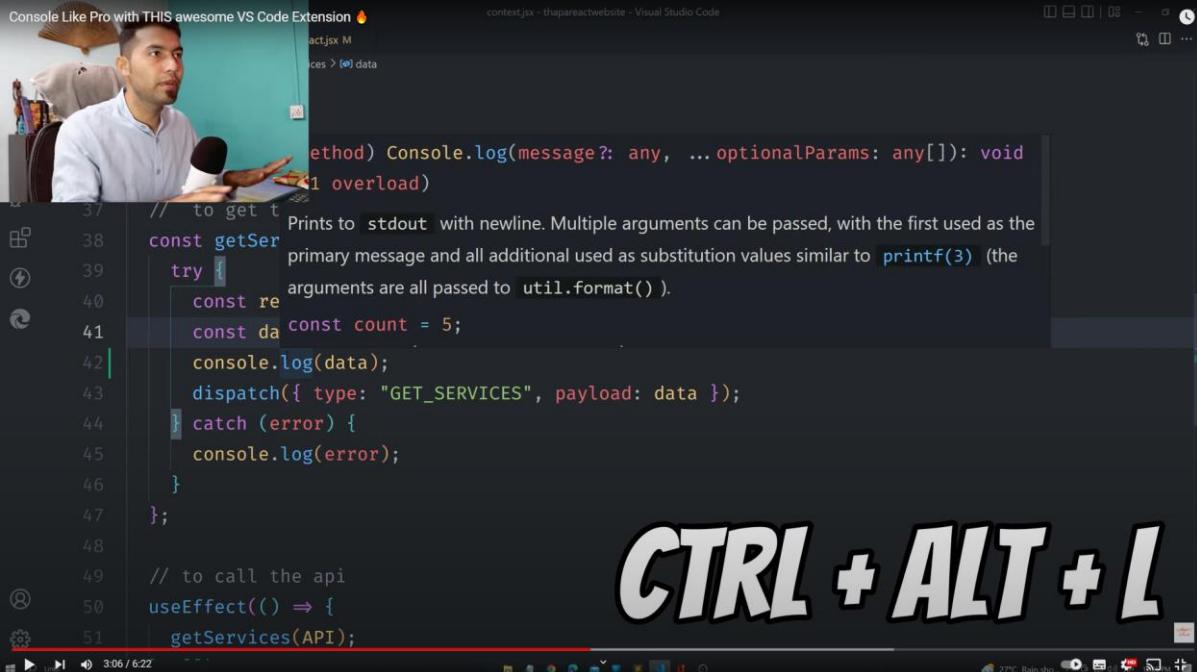
we cannot make changes to the file which is in the staging area; to make changes we need to remove the file from the staging area so that we can make changes.

Merge Conflict:

Occurs when we merge code and we find different code on same line of master and the specific branch, so there will be a merge conflict.

Tips

Console.log tips



Console Like Pro with THIS awesome VS Code Extension 🔥

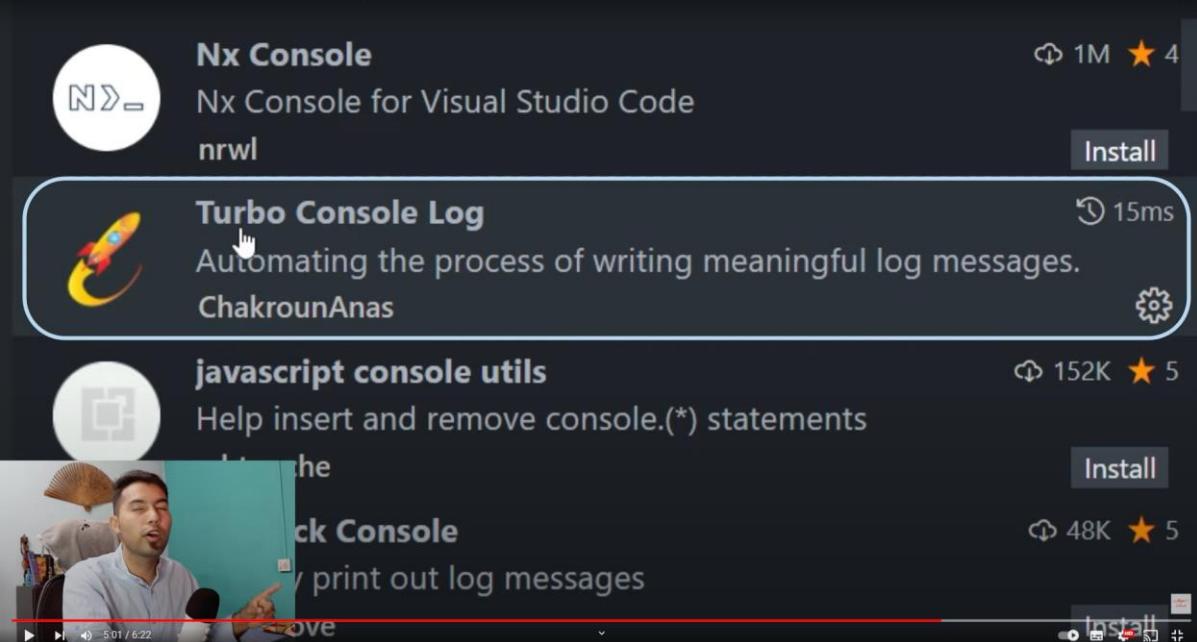
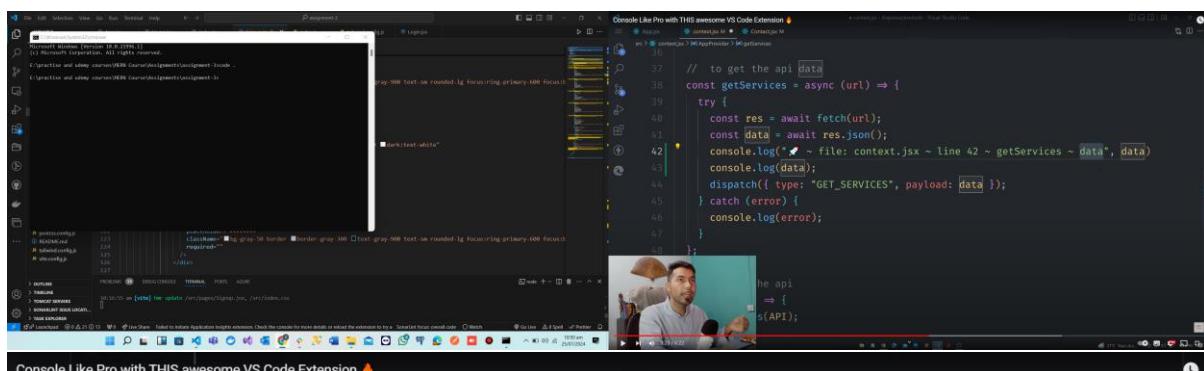
```
context.jsx - thupareactwebsite - Visual Studio Code
```

```
    method) Console.log(message?: any, ...optionalParams: any[]): void
      1 overload)

  37 // to get t
  38 const getSer
  39   try {
  40     const re
  41     const da const count = 5;
  42     console.log(data);
  43     dispatch({ type: "GET_SERVICES", payload: data });
  44   } catch (error) {
  45     console.log(error);
  46   }
  47 };
  48
  49 // to call the api
  50 useEffect(() => {
  51   getServices(API);
```

Prints to stdout with newline. Multiple arguments can be passed, with the first used as the primary message and all additional used as substitution values similar to `printf(3)` (the arguments are all passed to `util.format()`).

CTRL + ALT + L



Nx Console
Nx Console for Visual Studio Code
nrwl

Turbo Console Log
Automating the process of writing meaningful log messages.
ChakrounAnas

javascript console utils
Help insert and remove console.(*) statements

Stack Console
Easily print out log messages

HTML:

Html is a markup language not programming logic because it has no logic , its just a presentation , we can't deal with situations like if this then this etc.

Programming languages: connection to db , logic building , dynamic functionality , user login etc

The extra spaces or lines in editor will not affect the result in web browser in case of html

Req:

A browser

A text Editor

Creating an HTML file

- Does NOT need a server
- Files must end with the **.html** extension
- Runs in a web browser (Chrome, FireFox, etc)
- **index.html** is the root / home page of a website

http://www.something.com

Loads the **index.html** file

http://www.something.com/about.html

Loads the **about.html** file

Tag Syntax

`<tagname>content</tagname>`

- Element names surrounded by angle brackets

`<h1>About Us</h1>`

- Normally come in pairs (start tag and end tag)

`<p>This is a paragraph</p>`

- End tag is usually the same but with a forward slash

`
` (self closing)

- Some tags close themselves (*Remnant of XHTML*)

`
` (Fine in HTML5)

HTML Page Structure

```
<html>
  <head>
    <title>Page title</title>
  </head>
  <body>
    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>
    <p>This is another paragraph.</p>
  </body>
</html>
```

Page Structure (HTML)

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>
  <body>
    <h1>My First Heading</h1>
    <p>My first paragraph.</p>
  </body>
</html>
```

HTML Crash Course For Absolute Beginners

Doctype

- Explains what type of document the page is
- HTML4, HTML5, XHTML, etc

HTML5

```
<!DOCTYPE html>
```

HTML4.01 Strict

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
```

XHTML 1.0 Strict

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

The screenshot shows a dual-pane interface. On the left, a Sublime Text editor displays the code for an HTML file named 'index.html'. The code includes doctype declarations for HTML5 and HTML4.01 Strict, along with various heading tags (h1 through h6) and a paragraph tag. On the right, a Google Chrome browser window displays the rendered content of the HTML code, showing the six headings from 'Heading One' down to 'Heading Six'.

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>HTML Cheat Sheet</title>
5   </head>
6   <body>
7     <!-- Headings -->
8     <h1>Heading One</h1>
9     <h2>Heading Two</h2>
10    <h3>Heading Three</h3>
11    <h4>Heading Four</h4>
12    <h5>Heading Five</h5>
13    <h6>Heading Six</h6>
14
15    <!-- Paragraph -->
16    <p>
17
18    </p>
19  </body>
20 </html>
```



Inline .vs Block Level Elements

Inline Elements:

- Do not start on a new line
- Take only the necessary width

Block Elements:

- Start on a new line
- Take full width available

Block Level: <div>, <h1> - <h6>, <p>, <form>

Inline Level: , , <a>

The screenshot shows a code editor on the left and a browser window on the right. The code editor displays the following HTML code:

```
11 <h4>Heading Four</h4>
12 <h5>Heading Five</h5>
13 <h6>Heading Six</h6>
14
15 <!-- Paragraph -->
16 <p>
17     Lorem ipsum dolor sit amet, consectetur
18     adipisicing elit, sed do eiusmod
19     tempor <strong>incididunt ut labore</strong>
20     > et dolore magna aliqua. Ut enim ad minim
21     veniam,
22     quis nostrud <em>exercitation ullamco</em>
23     laboris nisi ut aliquip ex ea commodo
24     consequat. Duis aute irure dolor in
25     reprehenderit in voluptate velit esse
26     cillum dolore eu fugiat nulla pariatur.
     Excepteur sint occaecat cupidatat non
     proident, sunt in culpa qui officia deserunt mollit
     anim id est laborum.
```

The browser window shows the rendered HTML with the following structure:

- Heading One**
- Heading Two**
- Heading Three**
- Heading Four**
- Heading Five**
- Heading Six**
- Paragraph text: "Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."

difference b/w and

unlike other elements like , which indicates strong importance or emphasis, the tag is primarily used for stylistic purposes. It simply instructs the browser to display the enclosed text in bold, without implying any inherent significance or importance.

```
<p>
    Lorem ipsum dolor sit amet, consectetur
    adipisicing elit, <a href="">sed do eiusmod
    </a>
    tempor incididunt ut labore et dolore
    magna aliqua. Ut enim ad minim veniam,
    quis nostrud exercitation ullamco laboris
    nisi ut aliquip ex ea commodo
    consequat. Duis aute irure dolor in
    reprehenderit in voluptate velit esse
    cillum dolore eu fugiat nulla pariatur.
    Excepteur sint occaecat cupidatat non
    proident, sunt in culpa qui officia
    deserunt mollit anim id est laborum.
</p>
```

```
Lorem ipsum dolor sit amet, consectetur
adipisicing elit, <a href="http://
google.com">sed do eiusmod</a>
```

Heading Six

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

To open the link in new tab

```
Lorem ipsum dolor sit amet, consectetur
adipisicing elit, <a href="http://
google.com" target="_blank">sed do eiusmod
</a>
```



HTML Crash Course For Absolute Beginners



Watch later

Tag Attributes

```
<tagname
attributeName="attributevalue">content</tag
name>

<h1 title="My Company" >About Us</h1>
```

- All tags can have attributes
- Provide information about an element
- Placed within the start tag
- Key/value pairs (id="someId")

list:

The screenshot shows a browser window with the title "HTML Crash Course For Absolute Beginners". The left pane displays the HTML code for "index.html", which includes various heading levels (h1-h6), a paragraph with lorem ipsum text, and two lists (ul). The right pane shows the browser's developer tools, specifically the Elements tab of the DevTools. The DOM tree is expanded to show the structure of the headings and lists. The Styles tab is active, displaying CSS rules for the headings. One rule for h4 is highlighted, showing the following code:

```
menu, dir {  
    display: block;  
    list-style-type: disc;  
    -webkit-margin-before:  
        1em;  
    -webkit-margin-after:  
        1em;  
    -webkit-line-height-multiplier: 1.5x;}
```

This screenshot is similar to the one above, showing the developer tools in a different state. The DOM tree now shows a nested list structure under the first list item of the ul. The Styles tab still displays the same CSS rule for h4. The browser window title is now "HTML Cheat Sheet".

tables:

```

47      <!-- Table -->
48      <table>
49          <thead>
50              <tr>
51                  <th>Name</th>
52                  <th>Email</th>
53                  <th>Age</th>
54              </tr>
55          </thead>
56          <tbody>
57              <tr>
58                  <td>Brad Traversy</td>
59                  <td>brad@something.com</td>
60                  <td>35</td>
61              </tr>
62          </tbody>
63      </table>
64

```



```

<tbody>
    <tr>
        <td>Brad Traversy</td>
        <td>brad@something.com</td>
        <td>35</td>
    </tr>
    <tr>
        <td>John Doe</td>
        <td>jdoe@something.com</td>
        <td>45</td>
    </tr>
    <tr>
        <td>Sara Williams</td>
        <td>sara@something.com</td>
        <td>25</td>
    </tr>
</tbody>
</table>

```

using table for layout is wrong approach

 line break

<hr> : horizontal row ; this will draw a line

Forms

```
<!-- Forms -->
<form action="process.php" method="POST">
    <div>
        <label>First Name</label>
        <input type="text" name="firstName">
    </div>
    <div>
        <label>Last Name</label>
        <input type="text" name="lastName">
    </div>
</form>
```

```
<div>
    <label>Email</label>
    <input type="email" name="email">
</div>
<br>
<div>
    <label>Message</label>
    <textarea name="message"></textarea>
</div>
<br>
<div>
    <label>Message</label>
    <textarea name="message"></textarea>
</div>
</form>
```

```
<div>
    <label>Gender</label>
    <select name="gender">
        <option value="male">Male</option>
        <option value="female">Female</option>
        <option value="other">Other</option>
    </select>
</div>
```

```
<div>
    <label>Age:</label>
    <input type="number" name="age" value="30">
</div>
<br>
<div>
    <label>Birthday:</label>
    <input type="date" name="birthday" >
</div>
<br>
</form>
```

- List Item 3
- List Item 4

1. List Item 1
2. List Item 2
3. List Item 3
4. List Item 4

Name	Email	Age
Brad Traversy	brad@something.com	35
John Doe	jdoe@something.com	45
Sara Williams	sara@something.com	25

First Name
Last Name

- List item 1
- List Item 2
- List Item 3
- List Item 4

Name	Email	Age
Brad Traversy	brad@something.com	35
John Doe	jdoe@something.com	45
Sara Williams	sara@something.com	25

First Name
Last Name
Email
Message

John Doe jdoe@something.com 45
Sara Williams sara@something.com 25

First Name
Last Name
Email
Message
Gender

Name	Email	Age
Brad Traversy	brad@something.com	35
John Doe	jdoe@something.com	45
Sara Williams	sara@something.com	25

First Name
Last Name
Email
Message
Gender
Age:
Birthday:

```

<div>
    <label>First Name</label>
    <input type="text" name="firstName"
        placeholder="Enter first name">
</div>
<br>
<!-- Button -->
<button>Click Me</button>
<br>
<input type="submit" name="submit"
    value="Submit">
</form>

<!-- Button -->
<button>Click Me</button>
<br>
<input type="submit" name="submit"
    value="Submit">
</form>

<!-- Image -->


```

4. List Item 4

Name	Email	Age
Brad Traversy	brad@something.com	35
John Doe	jdoe@something.com	45
Sara Williams	sara@something.com	25

First Name

Message

Gender

Age:

Birthday:

Birthday:

Input me jo name attribute ha wo server side k lye hota ha k hum value kaise grap krenge backend pe

Image

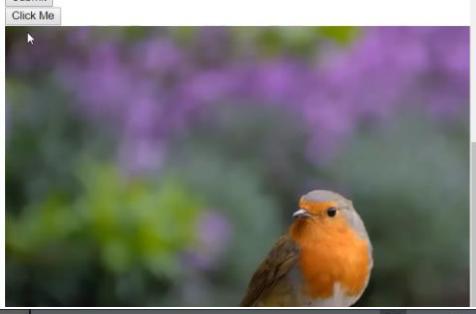
```

</div>
<br>
<input type="submit" name="submit"
    value="Submit">
</form>

<!-- Button -->
<button>Click Me</button>

<!-- Image -->

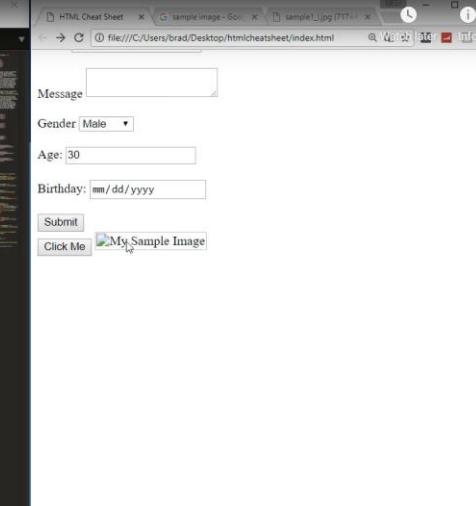

```



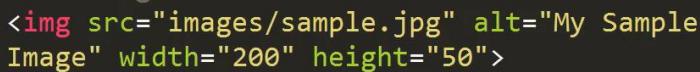

```

15      <br>
16      <div>
17          <label>Birthday:</label>
18          <input type="date" name="birthday"
19          >
20      </div>
21      <br>
22      <input type="submit" name="submit"
23          value="Submit">
24
25      <!-- Button -->
26      <button>Click Me</button>
27
28      <!-- Image -->
29      

```



```
<br>

<!-- Image -->

```

Gender:

Age:

Birthday:

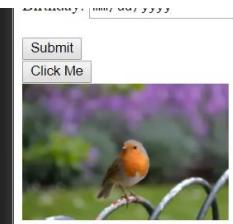


to open img in a new tab

```
<!-- Image -->

    <img alt="My Sample Image" width="200" data-bbox="201 275 601 305">

```



Quotations

```
<!-- Quotations -->
<blockquote cite="http://traversymed">

</blockquote>
```

```
<!-- Quotations -->
<blockquote cite="http://traversymedia.com">
    Lorem ipsum dolor sit amet, consectetur
    adipisicing elit, sed do eiusmod
    tempor incididunt ut labore et dolore
    magna aliqua. Ut enim ad minim veniam,
    quis nostrud exercitation ullamco laboris
    nisi ut aliquip ex ea commodo
    consequat. Duis aute irure dolor in
    reprehenderit in voluptate velit esse
    cillum dolore eu fugiat nulla pariatur.
```



Velit esse cillum dolore eu fugiat nulla
tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea
commodo consequat. Duis aute irure dolor in reprehenderit in voluptate
velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat
cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id
est laborum.

abbr

```
<p>The <abbr title="World Wide Web">WWW</abbr>  
is awesome</p>
```

Velit esse cillum dolore eu fugiat nulla
tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea
commodo consequat. Duis aute irure dolor in reprehenderit in voluptate
velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat
cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id
est laborum.

The WWW is awesome

```
<p>The <abbr title="World Wide Web">WWW</abbr>  
is awesome</p>
```

Velit esse cillum dolore eu fugiat nulla
tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea
commodo consequat. Duis aute irure dolor in reprehenderit in voluptate
velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat
cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id
est laborum.

The WWW is awesome

```
<div style="margin-top:500px"></div>
```

The WWW is awesome

cite tag

```
<p><cite>HTML crash course</cite> by Brad  
Traversy</p>
```

The WWW is awesome

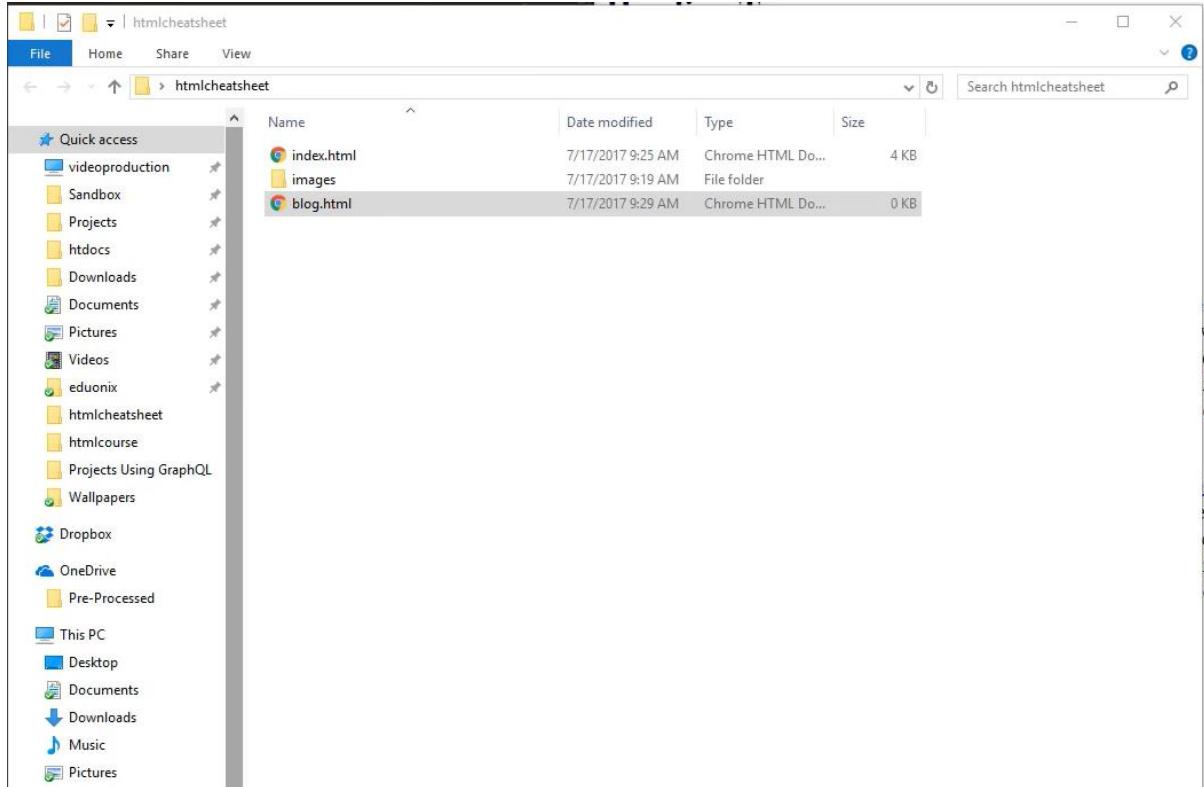
HTML crash course by Brad Traversy

Semantic tags

HTML5 *Semantic* *Tags*

A semantic element clearly describes its meaning to both the browser and the developer.

```
<header></header>
<footer></footer>
<aside></aside>
<main></main>
<article></article>
<nav></nav>
<section></section>
<details></details>
```



The screenshot shows a dual-pane interface. On the left, the Sublime Text editor displays the file `blog.html` with the following content:

```
<!DOCTYPE html>
<html>
<head>
    <title>My Blog</title>
</head>
<body>
    <header>
        <h1>My Website</h1>
    </header>

    <section>
        <article>
            <h3>Blog Post One</h3>
            <small>Posted by Brad on July 17</small>
        </article>
    </section>
</body>
</html>
```

On the right, a web browser window titled "My Website" shows the rendered HTML. The page title is "My Website". It features a header with the title "My Website". Below the header is a section containing an article. The article has a heading "Blog Post One" and a timestamp "Posted by Brad on July 17".

The screenshot shows a dual-pane interface. On the left, the Sublime Text editor displays the file `blog.html` with the following content:

```
<small>Posted by Brad on July 17</small>
<p>Lorem ipsum dolor sit amet, consectetur
adipisicing elit, sed do eiusmod
tempor incididunt ut labore et dolore
magna aliqua. Ut enim ad minim veniam,
quis nostrud exercitation ullamco laboris
nisi ut aliquip ex ea commodo
consequat. Duis aute irure dolor in
reprehenderit in voluptate velit esse
cillum dolore eu fugiat nulla pariatur.
Excepteur sint occaecat cupidatat non
proident, sunt in culpa qui officia
deserunt mollit anim id est laborum.</p>
<a href="post.html">Read More</a>
</article>

<article class="post">
    <h3>Blog Post Two</h3>
    <small>Posted by Brad on July 17</small>
    <p>Lorem ipsum dolor sit amet, consectetur
adipisicing elit, sed do eiusmod
tempor incididunt ut labore et dolore
magna aliqua. Ut enim ad minim veniam,
quis nostrud exercitation ullamco laboris
nisi ut aliquip ex ea commodo
consequat. Duis aute irure dolor in
reprehenderit in voluptate velit esse
cillum dolore eu fugiat nulla
pariatur. Excepteur sint occaecat cupidatat non
proident, sunt in culpa qui officia deserunt mollit anim id est
laborum.
```

On the right, a web browser window titled "My Website" shows the rendered HTML. It displays two blog posts. The first post is titled "Blog Post One" and contains the text: "Posted by Brad on July 17". The second post is titled "Blog Post Two" and contains the text: "Posted by Brad on July 17". Both posts include a "Read More" link.

```

<small>Posted by Brad on July 17</small>
<p>Lorem ipsum dolor sit amet, consectetur
adipisicing elit, sed do eiusmod
tempor incididunt ut labore et dolore
magna aliqua. Ut enim ad minim veniam,
quis nostrud exercitation ullamco laboris
nisi ut aliquip ex ea commodo
consequat. Duis aute irure dolor in
reprehenderit in voluptate velit esse
cillum dolore eu fugiat nulla pariatur.
Excepteur sint occaecat cupidatat non
proident, sunt in culpa qui officia
deserunt mollit anim id est laborum.</p>
<a href="post.html">Read More</a>
</article>
</section>

```

My Website

Blog Post One

Posted by Brad on July 17

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

[Read More](#)

Blog Post Two

Posted by Brad on July 17

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

[Read More](#)

Blog Post Three

Posted by Brad on July 17

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

[Read More](#)

Blog Post Two

Posted by Brad on July 17

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

[Read More](#)

Blog Post Three

Posted by Brad on July 17

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

[Read More](#)

Categories

- Category_1
- Category_2
- Category_3

Blog Post Two

Posted by Brad on July 17

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

[Read More](#)

Blog Post Three

Posted by Brad on July 17

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

[Read More](#)

Categories

- Category_1
- Category_2
- Category_3

Copyright © 2017, My Website

The screenshot shows a dual-pane view. On the left, the Sublime Text editor displays the HTML code for a blog website. On the right, a Google Chrome window shows the rendered website.

```

<!DOCTYPE html>
<html>
<head>
    <title>My Blog</title>
    <style type="text/css">
        #main-header{
            text-align:center;
            background-color:black;
            color:white;
            padding:10px;
        }
    </style>
</head>
<body>
    <header id="main-header">
        <h1>My Website</h1>
    </header>

```

The browser preview shows a black header with white text "My Blog" and "My Website". Below the header is a section titled "Blog Post One" with placeholder text. There are three "Read More" links and a "Blog Post Three" section with similar content. A "Categories" sidebar lists "Category 1", "Category 2", and "Category 3". At the bottom, it says "Copyright © 2017, My Website".

<meta> tags

When google indexes your website , its gonna look at description and keywords things like that

The screenshot shows the same dual-pane setup. The editor now includes meta tags in the head section:

```

<head>
    <title>My Blog</title>
    <meta name="description" content="Awesome blog by Traversy Media" />
    <meta name="keywords" content="web design blog, web dev blog, traversy media" />
    <style type="text/css">
        #main-header{
            text-align:center;
            background-color:black;
            color:white;
            padding:10px;
        }

```

The browser preview shows the updated page content reflecting the changes made in the code.

navigation

The screenshot displays three windows: a Sublime Text editor, a browser, and another Sublime Text editor.

Top Sublime Text Window (blog.html):

```
14
15     #main-footer{
16         text-align: center;
17         font-size:18px;
18     }
19 </style>
20 </head>
21 <body>
22     <header id="main-header">
23         <h1>My Website</h1>
24     </header>
25
26     <a href="index.html">Go to index</a>
27
28     <section>
29         <article class="post">
30             <h3>Blog Post One</h3>
31             <small>Posted by Brad on July 17</small>
32             <p>Lorem ipsum dolor sit amet, consectetur
33                 adipisicing elit, sed do eiusmod
34                 tempor incididunt ut labore et dolore
35                 magna aliqua. Ut enim ad minim veniam,
36                 quis nostrud exercitation ullamco laboris
37                 nisi ut aliquip ex ea commodo
```

Bottom Sublime Text Window (index.html):

```
1 <!DOCTYPE html>
2 <html>
3     <head>
4         <title>HTML Cheat Sheet</title>
5     </head>
6     <body>
7         <a href="blog.html"></a>
8         <!-- Headings -->
9         <h1>Heading One</h1>
10        <h2>Heading Two</h2>
11        <h3>Heading Three</h3>
12        <h4>Heading Four</h4>
13        <h5>Heading Five</h5>
14        <h6>Heading Six</h6>
15
16         <!-- Paragraph -->
17         <p>
18             Lorem ipsum dolor sit amet, consectetur
19                 adipisicing elit, sed do eiusmod
20                 tempor <strong>incididunt ut labore</strong>
21                 et dolore magna aliqua. Ut enim
22                 ad minim veniam,
23                 quis nostrud <em>exercitation ullamco</em>
24                 laboris nisi ut aliquip ex ea commodo
```

Browser Window:

The browser shows a website titled "My Website". It features a header with "My Website", a footer with "Go to index", and three blog posts:

- Blog Post One**: Posted by Brad on July 17. Content: "Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum." [Read More](#)
- Blog Post Two**: Posted by Brad on July 17. Content: "Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum." [Read More](#)
- Blog Post Three**: Posted by Brad on July 17. Content: "Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum." [Read More](#)

Footer: Copyright © 2017, My Website

CSS

The screenshot shows a video player interface. The title bar says "CSS Crash Course - Tutorial for Complete Beginners". The main content area contains the following text:

CSS stands for Cascading Style Sheets and is the language used to style an HTML document

It is a stylesheet language and not a programming language

With CSS you control the layout of the webpage, the spacing between elements, what color the text should be in and so on

Select HTML elements and apply styles

How do I select an HTML element?

How do I style that selected element?

The video player has a play button, a progress bar, and a timestamp of "1:05:47". The bottom right corner shows a "2x" zoom level.

How to add CSS?

Inline styles

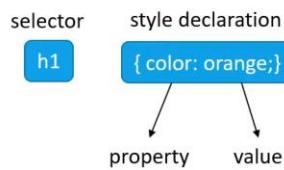
Internal stylesheet

External stylesheet

The screenshot shows the Visual Studio Code interface with the title bar "CSS Crash Course - Tutorial for Complete Beginners" and the file name "index.html". The code editor displays the following HTML code:

```
File Edit Selection View Go Run Terminal Help
index.html - CSS Crash Course - Visual Studio Code
index.html
1  <!DOCTYPE html>
2  <html lang="en">
3      <head>
4          <meta charset="UTF-8" />
5          <meta http-equiv="X-UA-Compatible" content="IE=edge" />
6          <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7          <title>Document</title>
8      </head>
9      <body>
10         <h1 style="color: orange">CSS Crash Course</h1>
11     </body>
12 </html>
13
```

CSS Rule Syntax



2nd method:

```
File Edit Selection View Go Run Terminal Help
index.html • index.html - CSS Crash Course - Visual Studio Code
index.html > body > h1
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8" />
5     <meta http-equiv="X-UA-Compatible" content="IE=edge" />
6     <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7     <title>Document</title>
8   <style>
9     h1 {
10       color: orange;
11     }
12   </style>
13 </head>
14 <body>
15   <h1>CSS Crash Course</h1>
16 </body>
17 </html>
18
```

3rd way:

The screenshot shows two tabs open in Visual Studio Code: "index.html" and "styles.css".

The "index.html" tab displays the following HTML code:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Document</title>
    <link rel="stylesheet" href="styles.css" />
  </head>
  <body>
    <h1>CSS Crash Course</h1>
    <h1>Another h1 element</h1>
  </body>
</html>
```

The "styles.css" tab displays the following CSS code:

```
h1 {
  color: orange; }
```

universal selector *

Used to reset properties that browser set by default

CSS Reset

ch

Character unit. The CSS `ch` unit is defined as the width of the character 0 (zero, or U+0030) of the font.

While the `ch` unit works as an exact measurement for monospaced / fixed width fonts like Courier, it can be unpredictable with proportional fonts like Arial.

For example, if your font is Courier and you set an element's width to `60ch`, that element will be exactly 60 characters wide.

But if your font is Arial and you set an element's width to `60ch` there's no telling how wide the element will be – characters may overflow the container, or fall short.



[Source](#)

CSS Unit

<https://www.freecodecamp.org/news/css-unit-guide/#:~:text=The%20CSS%20ch%20unit%20is,with%20proportional%20fonts%20like%20Arial.>

```
* {
  margin: 0;
  padding: 0;
  font: inherit;
}

/* dark mode user-agent-styles */

html {
  color-scheme: dark light;
}

/* min body height */

body {
  min-height: 100svh;
}
```

Under the radar CSS features for your CSS reset

```

display: block;
max-width: 100%;

}

h1, h2, h3, h4, h5, h6 {
  text-wrap: balance;
}

p {
  max-width: 62ch;
  margin-bottom: 2rem;
  text-wrap: pretty;
}

.title {
  font-size: 5rem;
}

```

Add these to your reset

Section 1: a shorter title

Jump to:

- [Section 2](#)
- [Section 3](#)

“[Lorem ipsum dolor sit amet consectetur adipisicing elit. Justo molestias natus repellat totam, soluta vero dignissimos! Quia consequuntur expedita paritur incident, ipsa recusandae! Laboriosam fugit quos iste nam numquam facere.

Fugit, ipsa? Quae consequuntur molestias, veritatis quibusdam voluptas totam! Quas nobis nam praesentium culpa necessitatibus unde non minima, totam provident mollitia magni doloremque optio laborum. Officiis nam officia repellendus deleniti?

Natus illum voluptatibus explicabo corrupti nemo tempore, non veniam minus dolorum sit perferendis, reprehenderit unde! Eius eveniet numquam quisquam, nihil accusamus facere placeat quia eum tenetur quibusdam. Ipsam, architecto commodi.

Repellendus tempore cumque insta aliquam perspiciatis beatae corporis, tempora eius? Maxime aliquid, dolores facilis quasi, ex esse eligendi cum eaque, voluptatibus itaque minus eos. Praesentium error deleniti necessitatibus modi enim.

Section 2: a section with a longer title that might wrap over multiple lines!

Jump to:



GRID GARDEN

Level 28 of 28

You win! By the power of CSS grid, you were able to grow enough carrots for Froggy to bake his world famous 20-carrot cake. What, were you expecting a different hoppy friend?

If you enjoyed Grid Garden, be sure to check out Flexbox Froggy to learn about another powerful new feature of CSS layout. You can also find new coding games over at [Codeip](#).

Want to support Grid Garden? Spread the word to your friends and family about Grid Garden!

FLEXBOX FROGGY

Level 24 of 24

Bring the frogs home one last time by using the CSS properties you've learned:

- `justify-content`
- `align-items`
- `flex-direction`
- `order`
- `align-self`
- `flex-wrap`
- `flex-flow`
- `align-content`

```

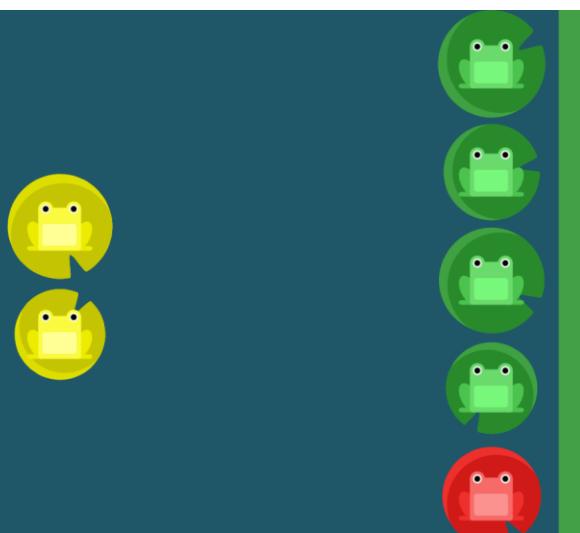
1 #pond {
2   display: flex;
3   flex-direction: column-reverse;
4   flex-wrap: wrap-reverse;
5   align-content: space-between;
6   justify-content: center;
7 }
8
9
10

```

[Next](#)

Flexbox Froggy is created by [Codeip](#) • [YouTube](#) • [Twitter](#) • [GitHub](#) • [Settings](#)

Want to learn CSS grid? [Play Grid Garden](#).



Variables

There are 3 types of variables in javascript

- 1) Let
- 2) Const
- 3) Var

Three major aspects to see differences:

- ❖ Scope
- ❖ Re-assignment / updating value
- ❖ Re-declaration

Dynamically typed language not static: therefore, we don't define data types. means jab ham variable declare krte hain tou data type nhi btate wo khud value se judge krlega aur runtime pe hi assign krdega whether its string num or boolean , obj etc

1) Variables:

Const: The scope of a const variable is **block scope**. And value of const **can't be re-assigned again. Can't be redeclared**

Let and const 2 different scopes me 2 different variables hain

E.g: {

```
  const v1 = 1;  
  console.log(v1);
```

}

let The scope of a let variable is **block scope** but can be **updateable or re-assigned**. Within a **single scope** var with same name **can't be redeclared**.

E.g: let v1 = 1;
 Console.log(v1);
 {
 v1 = 10;
 console.log(v1);
 }

-----XXXXXXXXXX-----

Let v1 =23;
v1= 24 --> ok
But
Let v1=23;
Let v1 = 24 -->error

```
15 let v4 = "12"
16 console.log("value of v4",v4);
17 {
18     let v4 = 23
19     console.log("value of v4 within scope: ")
20 }
```

TERMINAL ... powershell + ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂

```
2)
    at Module._load (node:internal/modules/cjs/loader:960:3)
2)
    at Function.executeUserEntryPoint [as runMain] (node:internal/modules/run_main:83:12)
        at node:internal/main/run_main_module:23:47

Node.js v20.3.0
● PS E:\practise and udemy courses\javascript> node .\variables.js
  number
  value of v4 12
  value of v4 within scope: 23
```

```
15 let v4 = "12"
16 console.log("value of v4",v4);
17
18 let v4 = 23
19 console.log("value of v4 within scope: ")
```

TERMINAL ... powershell + ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂

```
value of v4 12
value of v4 within scope: 23
✖ PS E:\practise and udemy courses\javascript> node .\variables.js
E:\practise and udemy courses\javascript\variables.js:18
  let v4 = 23
          ^
SyntaxError: Identifier 'v4' has already been declared
```

```
6  let v1 = "12";
7  console.log("v1 value ", v1);
8
9  {
10    v1 = 23;
11    console.log("v1 value within scope ", v1);
12  }
13  console.log("v1 value in outer scope ", v1);
14
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
at Function.executeUserEntryPoint [as runMain] (internal/main/run_main_module:11:1)
at node:internal/main/run_main_module:28:49

Node.js v20.11.1
PS D:\Desktop Data shifted here\Js project> node .\var.js
v1 value 12
v1 value within scope 23
PS D:\Desktop Data shifted here\Js project> node .\var.js
v1 value 12
v1 value within scope 23
PS D:\Desktop Data shifted here\Js project> node .\var.js
v1 value 12
v1 value within scope 23
v1 value in outer scope 23
```

Var: The scope of a var variable is functional or **global scope**. Can be **Re-assgined** and **re-declared**

Existing variable hi update hoga irrespective of any scope

```

5
6  var v1 = "12";
7  console.log("v1 value ", v1);
8
9  {
10    var v1 = 23;
11  }
12
13 console.log("v1 value in outer scope ", v1);
14

```

var	let	const
The scope of a <u>var</u> variable is functional or global scope.	The scope of a <u>let</u> variable is block scope.	The scope of a <u>const</u> variable is block scope.
It can be updated and re-declared in the same scope.	It can be updated but cannot be re-declared in the same scope.	It can neither be updated or re-declared in any scope.
It can be declared without initialization.	It can be declared without initialization.	It cannot be declared without initialization.
It can be accessed without initialization as its default value is “undefined”.	It cannot be accessed without initialization otherwise it will give ‘referenceError’.	It cannot be accessed without initialization, as it cannot be declared without initialization.
These variables are hoisted.	These variables are hoisted but stay in the temporal dead zone until the initialization.	These variables are hoisted but stays in the temporal dead zone until the initialization.

2) Data Types:

7 primitive data types in js:

1. String 2. Number 3. Boolean 4. Undefined 5. Null 6. Array 7. Object

3) Statement and Expression:

Every code line considered as Statement. Eg: const a = 1;

In a line if we use any arithmetic or logical operator so it becomes Expression.

Eg: const a = 1 +1 ;

Eg: if(a<b){}

if , else if , else

4) Loops:

Conditional Loops and Counter Loops:

- 1) for(let i=0; i<=20; i++){} >>> Counter Loop
- 2) while(i < 20){ >>> Condition Loop
 Console.log();
 i++;
}

5) Nested Loop:

```
for(int i=0; i<=10; i++){  
    for(int j=0; j<=5; j++){  
    }  
}  
3) Nested Loop:  
for(int i=0; i<=10; i++){  
    for(int j=0; j<=5; j++){  
        Console.log(i,j);  
    }  
}
```

6) Arrays:

const arr [];

Methods in Arrays:

Slice, splice, reverse, concat, sort, join, filter, reduce, find, flat, push, pop, shift, unshift, indexOf, includes

Functions and Methods:

> functions can be accessible without any object

> methods are for any class or can be linked with object

7) De-Structuring in Arrays:

Const arr = [1, 2, 3, 4];

Const [val1, val2] = arr; //in this step we are storing arr index 0 and 1 value in val1 and val2
this is destructuring.

Console.log(val1, val2);

}

Difference between Function and Method:

Wo functions jo without any object independently exist krte hain wo functions hotey hain
and method hamesha kisi class k hote hain means kisi object k sath link hote hain saarey methods jo
array k hain wo saare kisi object se related honge for instance: indexOf use krne k lye aik array ka
hona zaroori h.

Objects

-iterable

-key value pairs

-We can modify key names

-can store multiple data types

```
JS objects.js > ...
1
2 const person = {
3     name: "person 1",
4     gender : "M",
5     course: {
6         name: "MERN",
7         code : "MR-123",
8     },
9     array : [1,"one"]
10 }
11
12 console.log(person)
13 console.log("object.key: ",person.name)
14 console.log("object.key.key: ",person.course.name)
15 console.log("object.arr[index]: ",person.array[1])
16
17
18
19 // keys,values, entries
20 // De-structuring

PROBLEMS 20 DEBUG CONSOLE TERMINAL PORTS AZURE

PS E:\practise and udemy courses\javascript> node .\objects.js
{
  name: 'person 1',
  gender: 'M',
  course: { name: 'MERN', code: 'MR-123' },
  array: [ 1, 'one' ]
}
object.key: person 1
object.key.key: MERN
object.arr[index]: one
```

De-structuring in object

```
1 const person = {  
2   name: "person 1",  
3   gender: "M"  
4 };  
5  
6 person["array"] = [1, "one"];  
7 person.course = {  
8   name: "MERN",  
9   code: "MR-123",  
10};  
11  
12 // keys, values, entries  
13  
14 // De-structuring  
15  
16 const { name } = person;  
17
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
[ 'name', 'gender', 'array', 'course' ]  
PS D:\Desktop Data shifted here\Js project> node .\obj.js  
[ 'name', 'gender', 'array', 'course' ]
```

Destructuring krte waqt {} use krenge

{ } me exactly same key ka name denge jiski value chahye unlike in array hum variable ka koi bhi name rakh dete the

```
14 // De-structuring
15
16 const { name } = person;
17
18 console.log(name);
19
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
[ 'name', 'gender', 'array', 'course' ]
[ 'person 1', 'M', [ 1, 'one' ], { name:
PS D:\Desktop Data shifted here\Js project
[
  [ 'name', 'person 1' ],
  [ 'gender', 'M' ],
  [ 'array', [ 1, 'one' ] ],
  [ 'course', { name: 'MERN', code: 'MR-' }
]
PS D:\Desktop Data shifted here\Js project
person 1
PS D:\Desktop Data shifted here\Js project
```

Agr ma koi aisa variable name rakhdu jiske name ki koi key exist hi nhi krti tou mere pas undefined aayega

Function

// Function declaration vs function expression

Hoisting

```
3 factorial ("arg1","arg2")      //hoisting ; function calling before declaration can be done in function declaration but can't be done function expression
4 function factorial (param1, param2) {
5   console.log(param1);
6   console.log(param2);
7
8 }
```

Function expression me hoisting nhi hoti means we cannot call function before dc

```

JS functions.js X
JS functions.js > ...
1 // Parameter vs arguments
2
3 factorial ("arg1","arg2")           //hoisting ; function calling before declaration can be done in function declaration but can't be done function expression
4 function factorial (param1, param2) {
5   console.log(param1);
6   console.log(param2);
7 }
8
9
10 // Function declaration vs function expression
11
12 // function expression
13 palindrome("arg1","arg")
14 const palindrome = function(param1, param2) {
15   console.log(param1);
16   console.log(param2);
17 }
18
19
20 // Arrow function , anonymous function , IIFE
21
22 // Default parameters
PROBLEMS 3 DEBUG CONSOLE TERMINAL PORTS AZURE
● PS E:\practise and udemy courses\javascript> node .\functions.js
arg1
arg2
● PS E:\practise and udemy courses\javascript> node .\functions.js
arg1
arg2
E:\practise and udemy courses\javascript\functions.js:13
palindrome("arg1","arg")
^

ReferenceError: Cannot access 'palindrome' before initialization
  at Object.<anonymous> (E:\practise and udemy courses\javascript\functions.js:13:1)
  at Module._compile (node:internal/modules/cjs/loader:1255:14)
  at Module._extensions..js (node:internal/modules/cjs/loader:1309:10)
  at Module.load (node:internal/modules/cjs/loader:1113:32)
  at Module._load (node:internal/modules/cjs/loader:960:12)
  at Function.executeUserEntryPoint [as runMain] (node:internal/modules/run_main:83:12)
  at node:internal/main/run_main_module:23:47

Node.js v20.3.0
© PS E:\practise and udemy courses\javascript> []

```

Function Pass by Copy:

```

26
27 const copy = (param) => {
28   console.log(param);
29   param = 3; [
30 ];
31
32 let param = 1;
33 copy(param);
34
35 // Higher-order functions
36
PROBLEMS DEBUG CONSOLE TERMINAL PORTS AZURE

```

Function Pass by Reference:

```

const copy = (param) => {
  console.log(param);
  param[3];
};

let param = 1;

const updateParam = (newValue) => {
  param = newValue;
};

copy(updateParam);
console.log(param);

```

Difference between map and forEach loop:

the main difference between map and forEach is that map returns a new array with the results of the function, while forEach does not return anything and only modifies the original array

ForEach:

forEach is a method that executes a provided function once for each element in an array. It does not return a new array, but instead modifies the original array.

Let's understand the line It does not return a new array, but instead modifies the original array

```
const names = ["Rohit", "Aakash", "Vinay", "Ashish", "Vasu"];  
  
const newNames=name.forEach(function(name, index, array) {  
    array[index] = name + "DEV";  
});  
console.log(newName) // Undefined  
  
console.log(names); // Output: ['RohitDEV', 'AakashDEV',  
'VinayDEV', 'AshishDEV', 'VasuDEV']
```

Here we can see the original array names are changed and newNames is coming undefined means it is not returning anything.

REACT

Framework and library difference

Framework --> opinited (mujhe jo bhi task krna ha framework me hoga uska method as it is extensive set of tool)

Library: choti choti libraries ko milake hum aik task krte hain

DOM (Document Object Model): html k multiple tags like 1 div uske ander multiple h and p etc browser uska 1 tree like structure bananta ha apne pas, nodes ki heirarchhy.

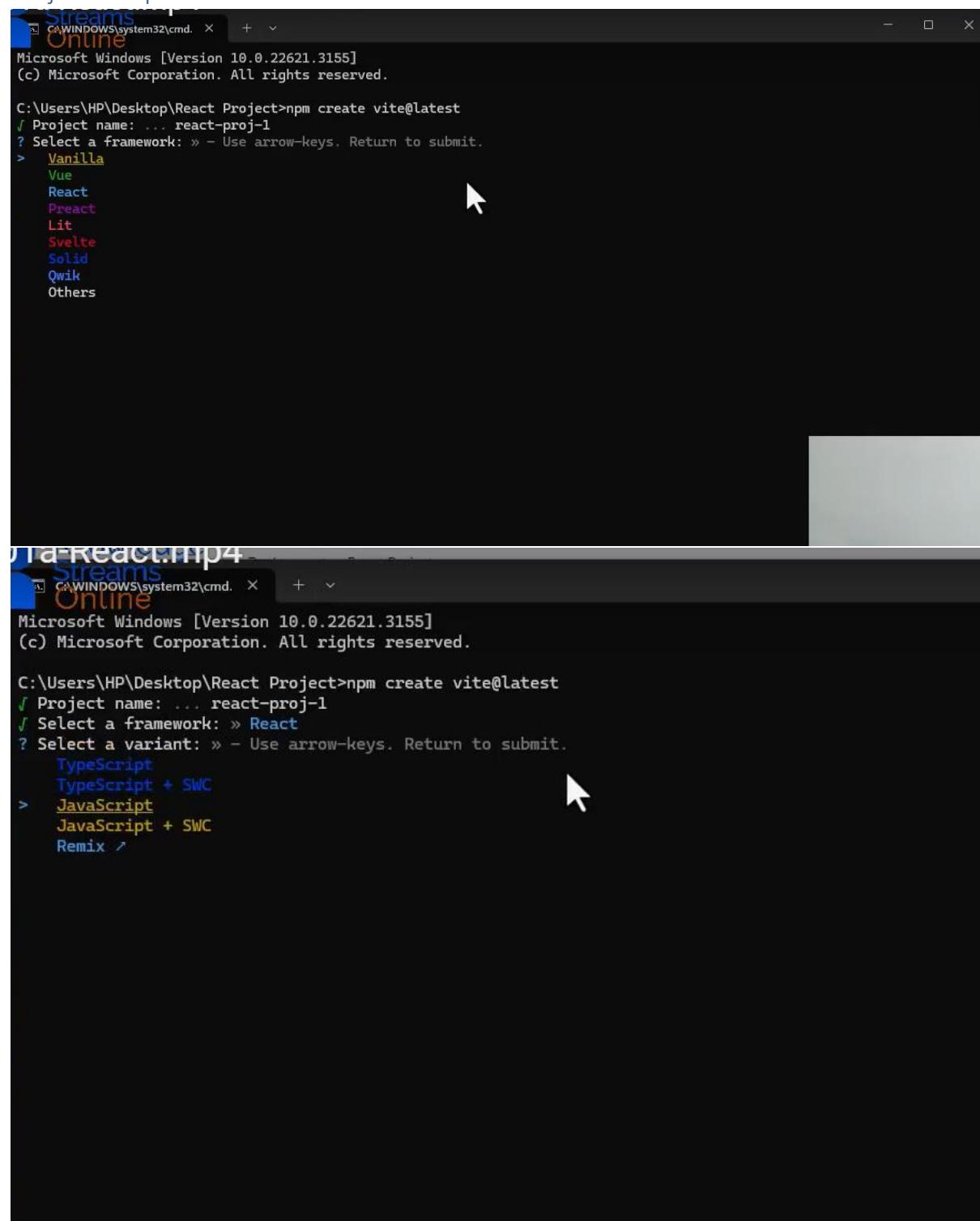
Jab bhi hum kisi pg ko refresh krte hain tou ye DOM re-build hota h, agr switch krte hain kisi pg pe tab bhi DOM Refresh hota h.

React apne pas actual DOM ki 1 copy banake rakhleta ha , jb bhi koi change aata ha kisi node me wo change virtual DOM me hota ha then actual DOM and **virtual DOM** compare hote hain aur jo chnage hota h sirf wo actual DOM me update hota ha

React ko chalane k lye aik software ki need h which is **node JS**

V8 Engine is name of software jo hamare browser me hota h aur javascript ko chalata h **node = V8Engine + C++**, Aim is k javascript code without browser bhi run hojaye

Project setup



The screenshot shows two instances of a Windows Command Prompt window. Both windows have the title bar 'Streams Online' and the path 'C:\Windows\system32\cmd.' at the top.

Top Window:

```
C:\Users\HP\Desktop\React Project>npm create vite@latest
? Project name: ... react-proj-1
? Select a framework: » - Use arrow-keys. Return to submit.
> Vanilla
  Vue
  React
  Preact
  Lit
  Svelte
  Solid
  Qwik
  Others
```

Bottom Window:

```
C:\Users\HP\Desktop\React Project>npm create vite@latest
? Project name: ... react-proj-1
? Select a framework: » React
? Select a variant: » - Use arrow-keys. Return to submit.
  TypeScript
  TypeScript + SWC
> JavaScript
  JavaScript + SWC
  Remix ↴
```

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows a project structure named "REACT-PROJ-1". The contents include a "public" folder, a "src" folder containing "assets", "# App.css", "# App.jsx", "# index.css", and "# main.jsx", and a ".eslintrc.cjs" file which is currently selected.
- Editor (Top Right):** Displays the content of the ".eslintrc.cjs" file. The code is as follows:

```
1 module.exports = {
2   root: true,
3   env: { browser: true, es2020: true },
4   extends: [
5     'eslint:recommended',
6     'plugin:react/recommended',
7     'plugin:react/jsx-runtime',
8     'plugin:react-hooks/recommended',
9   ],
10  ignorePatterns: ['dist', '.eslintrc.cjs'],
11  parserOptions: { ecmaVersion: 'latest', sourceType: 'module' },
12  settings: { react: { version: '18.2' } },
13  plugins: ['react-refresh'],
14  rules: {
15    'react/jsx-no-target-blank': 'off',
16    'react-refresh/only-export-components': [
17      'warn',
18      { allowConstantExport: true },
19    ],
20  },
21}
22
```

- Terminal (Bottom):** Shows a command-line session in PowerShell (PS) on Windows. The command "npm install" is being run, and the output shows the process of building the dependency tree for the project.

The screenshot shows the VS Code interface with the following details:

- File Explorer (Left):** Shows the project structure under "REACT-PROJ-1". The "node_modules" folder is expanded, displaying numerous packages like "@ampproject", "@babel", "@esbuild", "@eslint", etc.
- Editor (Top Right):** Displays the content of the ".eslintrc.cjs" file. The code defines a module exports object with rules for browser support, extends, ignore patterns, parser options, settings, and rules for react and react-refresh.
- Terminal (Bottom Right):** Shows the command "npm install" being run in the terminal. The output indicates 278 packages added and audited, with 103 packages looking for funding. It also lists several deprecated warnings from various packages.

JSX

JSX

Write HTML directly inside JavaScript code

HTML

```
class=""
```

JSX

```
className=""
```

MEANS we can write javascript and html combined.

Javascript ka code html return kr raha ha

JSX me **X** denote kr raha h k ye HTML nhi **XML** h means **extended markup language** kuch certain functionalities aisi hain jo HTML me nhi hain isme hain

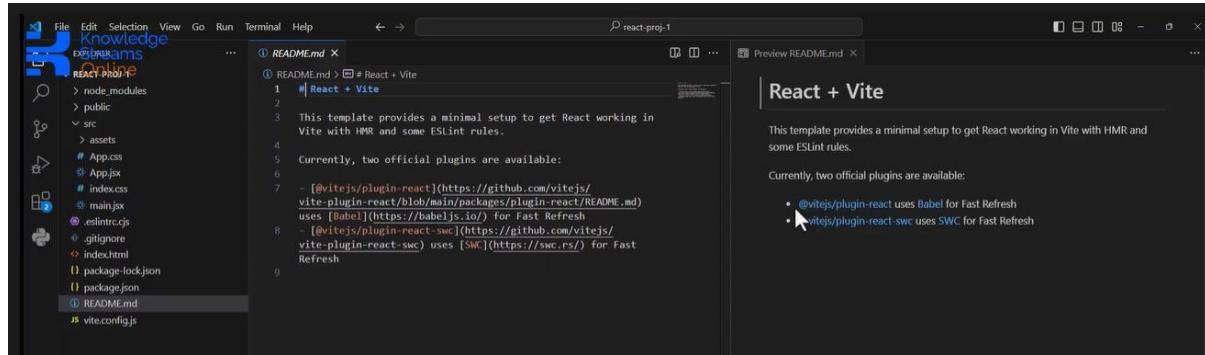
Jab project run hota ha tou code plain javascript me convert hoke run hota ha , tjs work is done by **BABEL**

Aisa translator jo apko aik language se doosri me le jaye without changing the complexity is known as **transpiler**

We are using **Vite** , there could be multiple transpilers , multiple techniques jo aik tool k proj me use hoti hain.

Default Project Structure

Readme.md : file contain project relevant info



Package.json: React proj configuration stored here , kon konsi dependencies is proj me use hui wi hain along with their versions. If another dev wants to continue my proj he can

Scripts --> cmd short commands

Package-lock.json : detailed information in this file

Mera proj jin packages pe dependent ha wo further kin packages me dependent ha wo info hoti ha is file me

.gitignore

Wo files jinko ham nhi chahte k git track kren unke names git ignore me mention krne hote hain

Eslinterc.cjs

Aik tool h esclinter uski configuration fie ha ye

Ye tool hamare upper achi coding practises enforce krta h for e.g kahin variable let se create kia wa ha aage kahin value change nhi hui code me

Index.html

Isme aik empty div ha jiski id root ha

Main.jsx

The screenshot shows a VS Code interface with a dark theme. On the left is the Explorer sidebar showing a file tree for a 'REACT-PROJECT'. The tree includes 'node_modules', 'public', 'src' (which contains 'assets', '# App.css', 'App.jsx', '# index.css', and 'main.jsx'), and several configuration files like '.eslintrc.cjs', '.gitignore', 'index.html', 'package-lock.json', 'package.json', 'README.md', and 'vite.config.js'. The 'main.jsx' file is currently selected and open in the main editor area. The code in 'main.jsx' is as follows:

```
src > main.jsx
1 import React from 'react'
2 import ReactDOM from 'react-dom/client'
3 import App from './App.jsx'
4 import './index.css'
5
6 ReactDOM.createRoot(document.getElementById('root')).render(
7   <React.StrictMode>
8     <App />
9   </React.StrictMode>,
10 )
11
```

Hum us div ki id pass kr rahe hain tou React DOM ko hum wo html wala div de rahe hain k yahan se apna DOM banana shuroo krdo tou React ka virtual DOM is tarah se html k real DOM se connect hota

Index.css deleted\, main.jsx se import of index.css remove

App.jsx code removed ; write from scratch

App.css removed

Assets folder removed

Remove svg from public folder

Props and children

Previously, you learned to pass props to and within a component. But there is also a special prop called props.children, which is automatically passed to every component. In this reading, you'll learn about props.children and its purpose.

To understand the concept of props.children, consider the following real-life situation: you have a couple of apples, and you have a couple of pears. You'd like to carry the apples some distance, so obviously, you'll use a bag.

It's not a "bag for apples", and it's not a "bag for pears." It's just a bag. Nothing about this bag makes it such that it needs to be referred to as a bag in which you'd only and always carry apples, nor a bag in which you'd only and always carry pears.

In a way, the bag "doesn't care" if it is used to carry apples or pears. Nothing about the bag changes. There are no changes in the bag's material, size, shape, or color - because it can handle apples or pears being carried inside, without issues.

Now, consider the following component:

13

```
export default Apples
```

There is also a Pears component:

5

```
}
```

Now, the question is this: Let's say you want a Bag component, which can be used to "carry" Apples or Pears. How would you do that?

This is where props.children comes in.

You can define a Bag component as follows:

1

2

3

4

5

6

7

8

9

10

11

12

13

14

```
function Bag(props) {  
  const bag = {  
    padding: "20px",  
    border: "1px solid gray",  
    background: "#fff",  
    margin: "20px 0"  
  }  
  return (  
    <div style={bag}>  
      {props.children}  
    </div>  
  )  
}  
export default Bag
```

So, what this does in the Bag component is: it adds a wrapping div with a specific styling, and then gives it props.children as its content.

But what is this props.children?

Consider a very simple example:

1

2

3

```
<Example>
```

```
Hello there  
</Example>
```

The Hello there text is a child of the Example JSX element. The Example JSX Element above is an "invocation" of the **Example.js** file, which, in modern React, is usually a function component.

This Hello there text can be passed as a **named prop** when rendering the Example component.

Here's what that would look like:

```
<Example children="Hello there" />
```

There are two ways to perform this action. But this is just the beginning.

What if you, say, wanted to surround the Hello there text in an h3 HTML element?

Obviously, in JSX, that is easily achievable:

```
<Example children={<h3>Hello there</h3>} />
```

What if the <h3>Hello there</h3> was a separate component, for example, named Hello?

In that case, you'd have to update the code like this:

```
<Example children={<Hello />} />
```

You could even make the Hello component more dynamic, by giving it its own prop:

```
<Example children={<Hello message="Hello there" />} />
```

So, how can you make the **Bag**, **Apples**, and **Pears** examples from the beginning of this reading work?

Here's how you'd render the Bag component with the Apples component as its props.children:

```
<Bag children={<Apples color="yellow" number="5" />} />
```

And here's how you'd render the Bag component, wrapping the Pears component:

```
<Bag children={<Pears friend="Peter" />} />
```

While the above syntax might look strange, it's important to understand what is happening. Effectively, the above syntax is the same as the two examples below.

```
</Bag>
```

You can even have multiple levels of nested JSX elements, or a single JSX element having multiple children, such as, for example:

```
<Trunk>  
  <Bag>  
    <Apples color="yellow" number="5" />  
    <Pears friend="Peter" />  
  </Bag>  
</Trunk>
```

So, in the above structure, there's a Trunk JSX element, inside of which is a single Bag JSX element, holding an Apples and a Pairs JSX element.

Before the end of this reading, consider this JSX element again:

1
2
3

```
<Bag>
  <Apples color="yellow" number="5" />
</Bag>
```

What is **Apples** to **Bag** in the above code?

In the above code, Apples is a prop of the Bag component. To explain further, the Bag component can wrap the Apples component, or **any other component**, because I used the {props.children} syntax in the Bag component function declaration. In other words, just like in the real world, when you take a bag to a grocery store, you can “wrap” a wide variety of groceries inside the bag, you can do the same thing in React: wrap a wide variety of components inside the Bag component, using the children prop to achieve this.

It's crucial to understand this when working with React.

Before the end of this reading, there's another important concept that you need to be aware of: *finding the right amount of modularization*. What does this mean? Imagine, for example, that you had a number of small bags, and that each bag could only carry a single apple or pear. You'd have to wrap each "apple" inside a "bag". That doesn't make much sense. You can think about components making your layouts modular in a similar way. You don't want to have an entire layout contained in a single component, because that would be very difficult to work with. On the flip side, if you made each HTML element in your layout a separate component, it would be very hard to work with, although such layout would be modular. So it's all about moderation. You need to organize your layouts by splitting them into meaningful areas of the page, and then code those meaningful areas as separate components. that would constitute the right amount of modularity. To reinforce this point, It might help to think of how a person would describe a website: a menu, a footer, a shopping cart, etc.

In conclusion, when you see a JSX element wrapping another JSX element, you can easily understand that it's all just props.children in the background.

Components

Structure , small piece of code ; different data put krsakte hain bar bar use krsakte hain ; take instagram post example

We can have multiple smal chunks of code on a page and/or we can use same component on multiple places

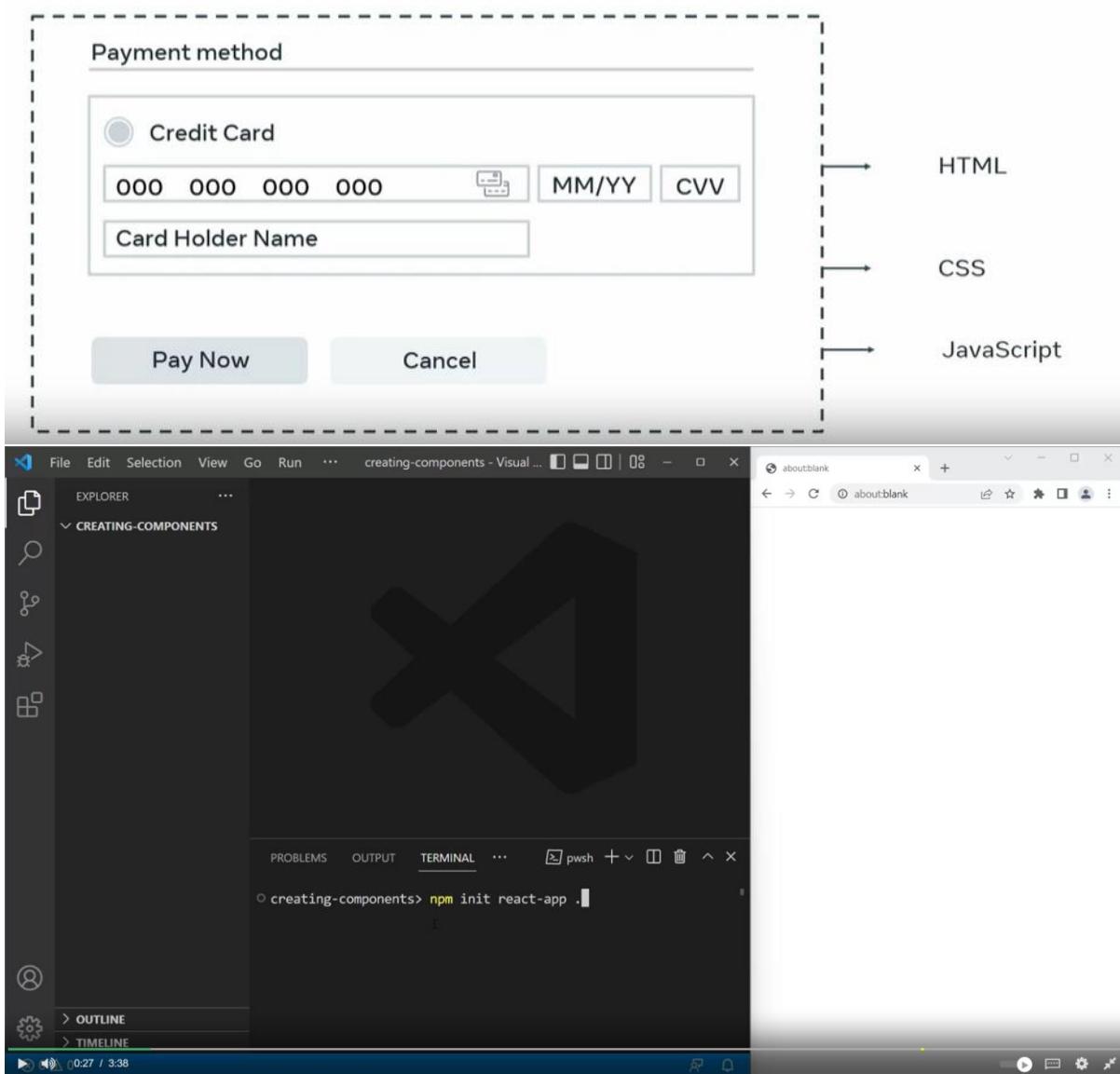
Reusable

Used multiple times and easily inserted anywhere where needed

Independent

Components can exist within the same space independently from each other

Component



Transpiler:

A browser cannot understand JSX syntax.

This means that making a browser understand React code requires a lot of supporting technologies.

An example of such a technology is a **transpiler**.

A **transpiler** takes a piece of code and transforms it into some other code.

Babel

What Babel does is this: it allows you to transpile JSX code (which cannot be understood by a browser) into plain JavaScript code (which can be understood by a browser).

Exports

Exports

App.js

```
//Default exports  
export default App;
```

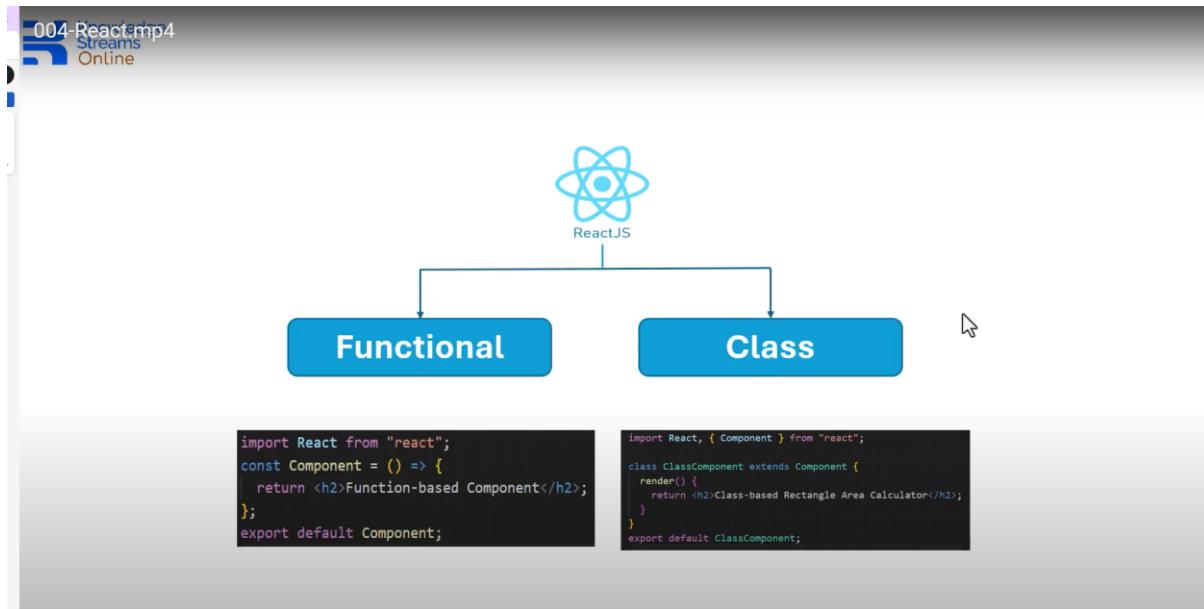
```
//Named exports  
export {App}
```

Component

Small piece of functionality

Module

Series of components



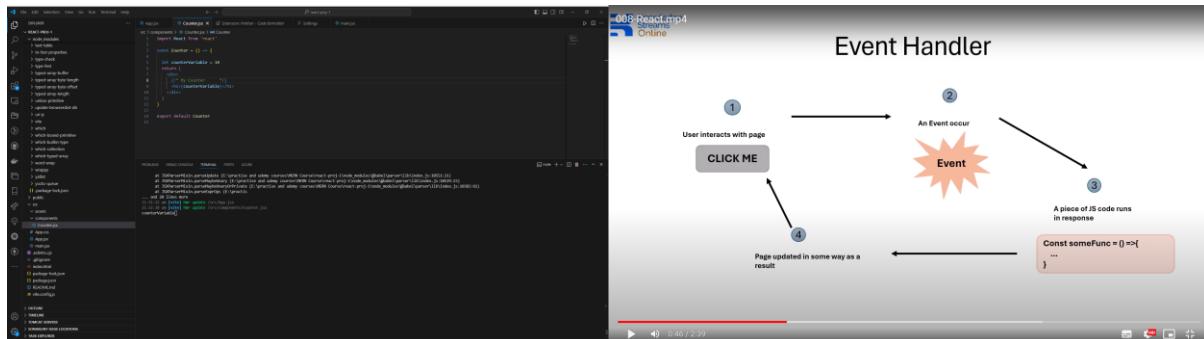
Aisa function or arrow function jisse kuch HTML return tkr rahe ho usey as a component use krsakte hain

Requirements

First letter must be capital

Return statement must have 1 outer most div

Event Handler



HTML me jab user kisi element se interact kta h tou aik event trigger hota h , event k trigger pe hum aik function pas karadete hai k iset run k

Issue: DOM is not updating

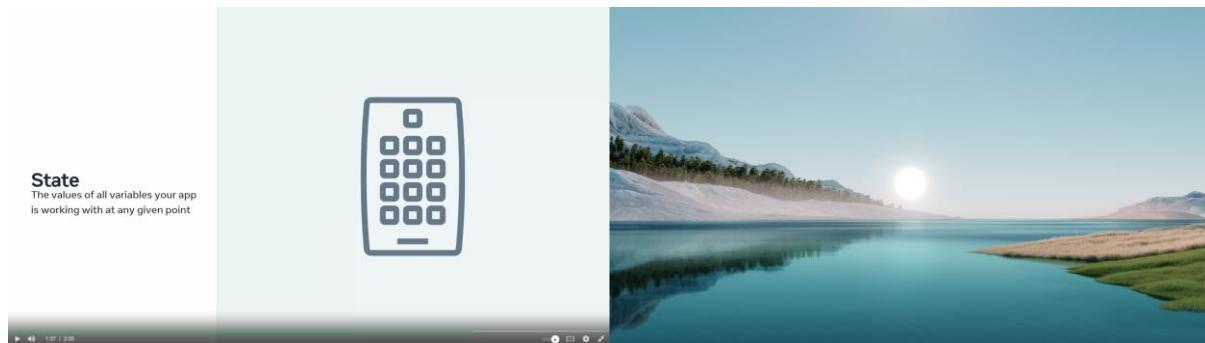
Variable ki value tou update horarhi ha event handler se ,console me value chnage horahi ha per DOM update nhi horha is waja se screen pe print nhi horha

reason

Aik dafa jab component render hoga tou browser kko component ka structure miljayega apne pas dikhane k lye uske baad beshak variable ko ma update krta rahun memory me, DOM ka structure tou chnage hi nhi horaha wo jo aik dafa ban gaya wohi h.

How to deal with it?

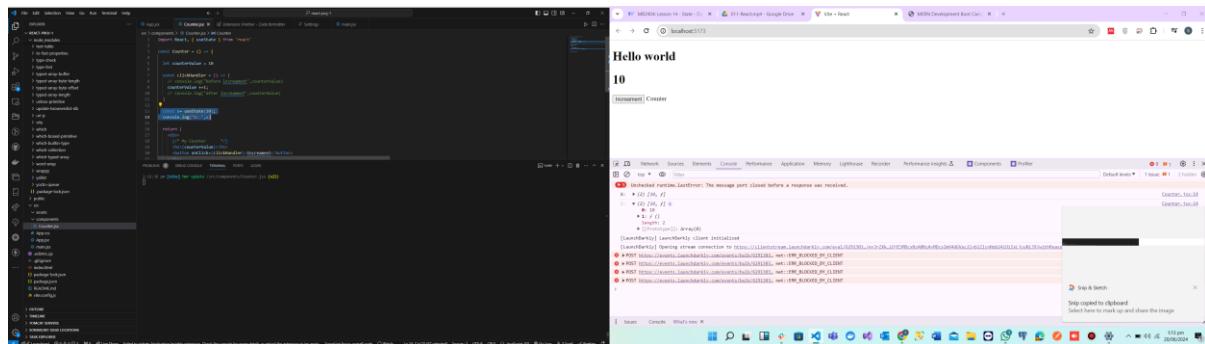
state



We need variables jinke chnage hone per hum react ko ye msg convey kren k apna DOM refresh kro and that are known as States

useState

State variable ko build krne k lye aik function hota ha usey useState kehte hain



Use state aik array ki form me data return krta ha ; look at the pic above
0th index pe value ha jo useState me pass ki which is 10

1st index me func return kia ha

Tou ham is data ko destructure krlete hain.

The screenshot shows the VS Code interface. The left sidebar displays the project structure: REACT-PROJ-1, node_modules, public, src, components, Counter.jsx (selected), App.jsx, main.jsx, and eslintrc.js. The right pane shows the code for Counter.jsx:

```
src > components > Counter.jsx > Counter
1 import { useState } from "react";
2
3 const Counter = () => [
4   const [counterValue, setCounterValue] = useState(10);
5
6   const clickHandler = () => {
7     setCounterValue(counterValue + 1);
8   };
9 ]
```

Few points to be noted:

- Yahan pe we dont use let instead we use const ; reason behind is variable ki value change nhi hogi jo function se set hogi wohi rahegi agr let use kr bhi len aur variable ko update krlen jaise pehle kr rahe the tou var tou update hojayega mem me lakin DOM update nhi hoga. Therefore we use const
- Jis name se statae ka variable banaya ha uska setter hamesha isi name se aayega, its a good practise not mandatory
- State variables are immutable**, we can not change the value of variables using let , only setter function iis used to change the value
- On a certain condition we can update the value of our state variables. Condition deni hogi hndler me

```
9
10  const decrementHandler = () => {
11    if (counterValue > 0) {
12      setCounterValue(counterValue - 1);
13    }
14  };
```

5.

ISSUE RESOLVED DOM IS UPDATING AS WE CHNAGE THE VALUE : THIS DONE BY STATE VAR USING USESTATE HOOK

Strict mode enable hone ki waja se console.log wali statement 2 times chalti hain

```
<button onClick={incrementHandler}>Increment</button>
<button onClick={decrementHandler}>Decrement</button>
```

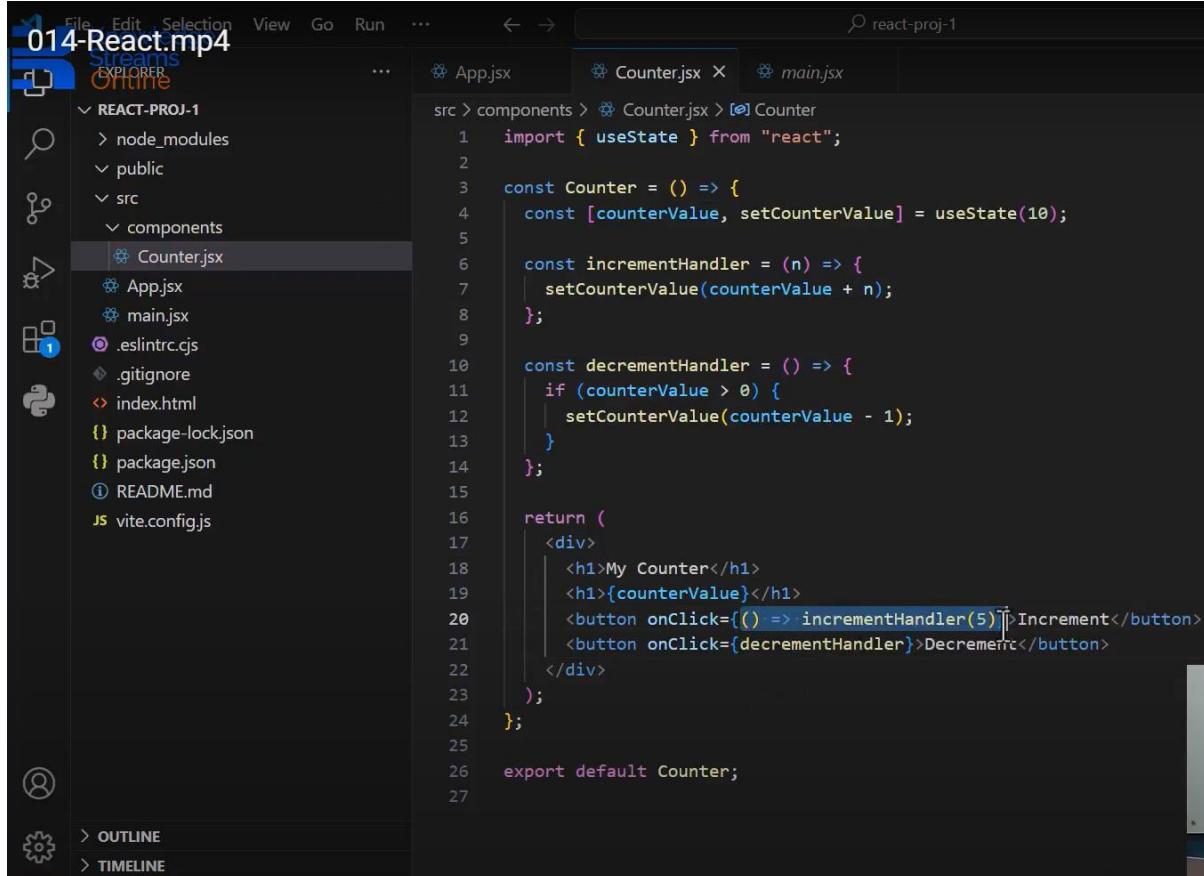
Event handler ko function ka reference pass kia wa ha. Hame function ka reference hi pass krna h k abhi is func ka reference rakhlo on click function call krna.

```
<h1>{counterValue}</h1>
<button onClick={incrementHandler()}>Increment</button>
```

Ye function call hua wa ha. Jaise hi program run hoga ye khud call hojayega.

Issue: What if I want to pass the parameter in the function?

Solution:

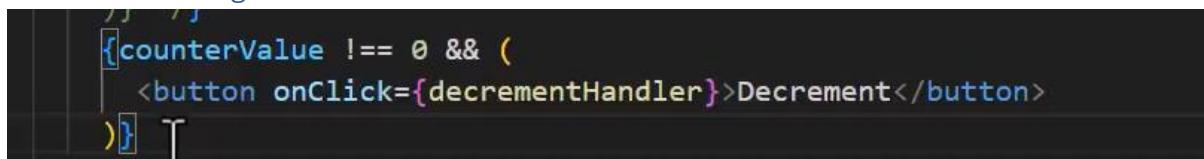


```
File Edit Selection View Go Run ... < > react-proj-1
014-React.mp4
REACT-PROJ-1
src > components > Counter.jsx > Counter
1 import { useState } from "react";
2
3 const Counter = () => {
4   const [counterValue, setCounterValue] = useState(10);
5
6   const incrementHandler = (n) => {
7     setCounterValue(counterValue + n);
8   };
9
10  const decrementHandler = () => {
11    if (counterValue > 0) {
12      setCounterValue(counterValue - 1);
13    }
14  };
15
16  return (
17    <div>
18      <h1>My Counter</h1>
19      <h1>{counterValue}</h1>
20      <button onClick={() => incrementHandler(5)}>Increment</button>
21      <button onClick={decrementHandler}>Decrement</button>
22    </div>
23  );
24
25  export default Counter;
26
27
```

Aik arrow function banallia jiske ander koi parametrr nhi h yani wo onclick pe hi call hoga aur is function k ander apna function call krlia along with parameter. Problem solved. This is short syntax for arrow function where we are not using curly brackets jahan {} nhi ha means ye function yehi return kr raha h.

1.Ternary operator conditional rendering

2.Short Circuiting



```
{counterValue !== 0 && (
  <button onClick={decrementHandler}>Decrement</button>
)}
```

AND operator me saare inputs true hun tou hi result true hota ha , agr aik condition bhi false hogain tou aage ki check hi nhi hotin, isilye agr pehli cond hi false hogai tou button render hi nhi hoga

3.Conditional Rendering with functions

State is separate for every occurrence of the same component

The screenshot shows the VS Code interface. On the left, the Explorer sidebar displays the project structure:

- REACT-PROJ-1
- node_modules
- public
- src
 - components
 - Counter.css
 - Counter.jsx
 - App.jsx
 - main.jsx
- .eslintrc.cjs
- .gitignore
- index.html
- package-lock.json
- package.json
- README.md
- vite.config.js

The current file is App.jsx, which contains the following code:

```
src > App.jsx > [App] App
1 import Counter from "./components/Counter";
2
3 const App = () => {
4   return (
5     <div>
6       |<Counter />
7       |<Counter />
8       |<Counter />
9     </div>
10  );
11};
12
13
14 export default App;
15
```

Hello world



Javascript objects:

JavaScript object

**Variable that can
contain many values**

**Groups related data of
different types**

**Contains object
properties**



JavaScript object

```
const fruit = {type:"Apple", quantity:500, color:"green"};
```



Properties

JavaScript object

```
const fruit = {type:"Apple", quantity:500, color:"green"};
```

```
console.log(fruit.type)
```

Props

React props

Pass data between components

Arguments are passed like HTML attributes

Uses the keyword props

Index.js

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

import App from './App.js'

ReactDOM.createRoot(
  document.querySelector('#root')
).render(<App title="Welcome" />)
```

App.js

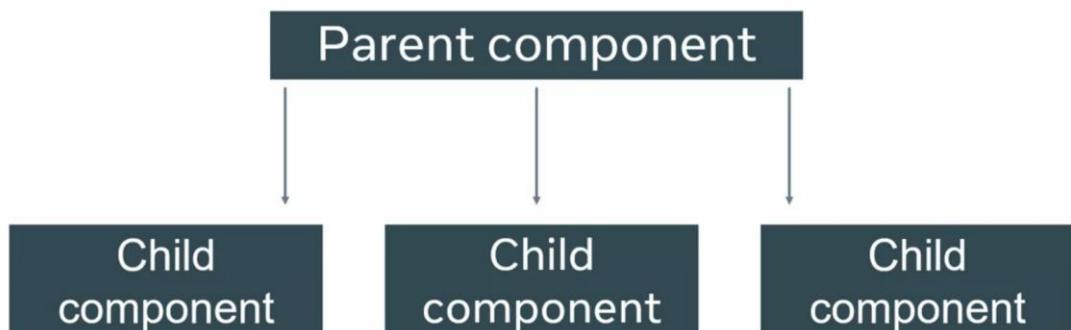
```
import React from 'react';

export function App(props) {
  return (
    <h1>{props.title}</h1>
  );
}
```

React props

Send multiple data types

Flexible dynamic content



Uni directional data flow , data can flow from parent to child using props but its not possible to send data from child to parent using props there are other approaches for it but not this

Pure functions:

Pure functions

Always returns the same output for the same argument values

Component with props

```
function Heading(props) {  
  props.title = "new value"; // not allowed → Not allowed  
  return (  
    <h1>{props.title}</h1>  
  );  
};
```

it must never modify
its own prompts.

Limitations

You also learned why developers use props in order to make their apps more dynamic and flexible. Finally, you examined some of the limitations about using props in that you cannot use them to send data back to a parent component, and the functions using props must never modify its own props.

```
<Counter itemName="Jeans"/>  
<Counter itemName="Shirts"/>  
<Counter itemName="Jacket"/>
```

Jeans	7	<button>Increment</button>	<button>Decrement</button>
Shirt	3	<button>Increment</button>	<button>Decrement</button>
Jacket	4	<button>Increment</button>	<button>Decrement</button>

State Vs Props

Wo variables jo component k ander defin krte hain uski internal working ko manipulate krne k liye usey state kehte hain jo hamare case me counterValue variable ha counter component me

Aise variables jo component k her occurrence k upper differ krne hain wo props ya wo variables jo ma pass krta hun component ko call krte waqt.

Map Function:

Javascript me arrays k liye aik func use hota ha map. Map ka method kia krta ha k aik function accept krta ha aur us array k her element k liye us function ko run krta ha aur koi result produce krta ha. Generated result ji bhi map return krta ha wo aik array me store hojata ha

```
| [items.map((()=>{}))]
```

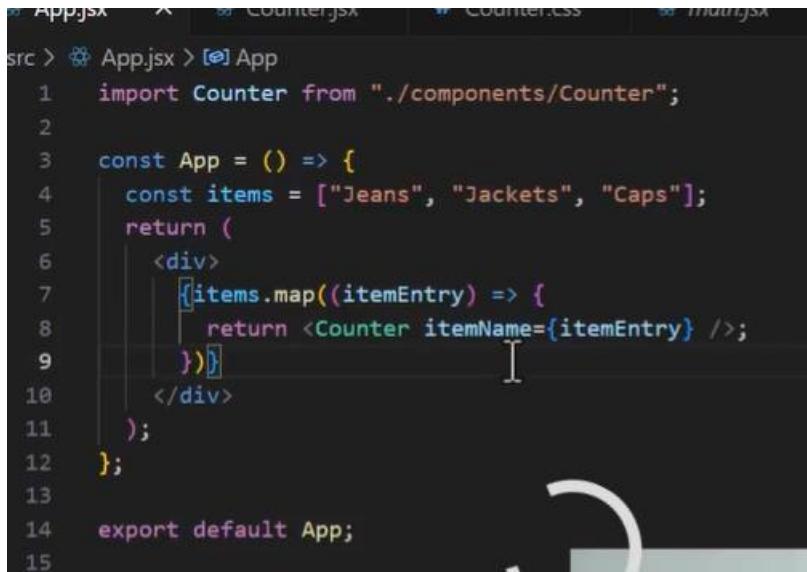
Map aik higher order function ha yani wo parameters me bhi aik function accept krta ha.

Ab is arrow function me bhi apne parameters jate hain

```
const App = () => {
  const items = ["Jeans", "Jackets", "Caps"];
  return <div>{items.map(itemEntry) => {}}</div>;
};

export default App;
```

Map function ka jo pehla parameter h usme jata ha array ka aik element at a time. Pehli def me itemEntry hoga jeans phir jacket etc etc..



```

src > App.jsx > App
1 import Counter from "./components/Counter";
2
3 const App = () => {
4   const items = ["Jeans", "Jackets", "Caps"];
5   return (
6     <div>
7       [items.map((itemEntry) => {
8         return <Counter itemEntry={itemEntry} />;
9       })]
10    </div>
11  );
12}
13
14 export default App;
15

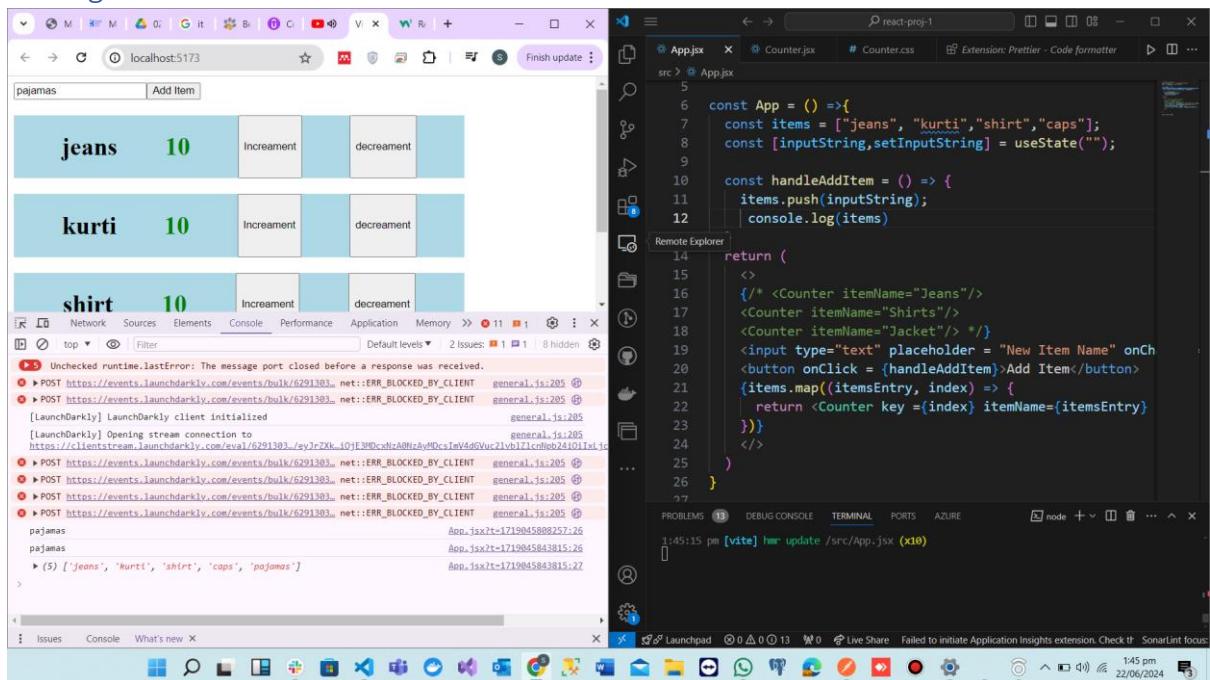
```

Map k function me return krwana zaroori h

Key prop

Array ka size kuch bhi hosakta h 10, 100 , 1000 etc her component seperately render horaha ha screen pe but react ko apne pas aik unique id rakhni h her aik component ki taake unko aapas me differentiate krsake aur us unique value ko key kehte hain yahan pe hum key index ko hi use krke bana rahe hain but at some places it can be problematic.

Adding New Items to the list



The screenshot shows a browser window on the left displaying a simple counter application with four items: jeans (10), kurti (10), shirt (10), and pajamas. Each item has an increment and decrement button. To the right is a Visual Studio Code (VS Code) interface showing the source code for App.jsx. The code defines a state 'items' containing four strings: 'jeans', 'kurti', 'shirt', and 'pajamas'. It uses a map function to render four Counter components, each with a unique key: the index of the item in the 'items' array. The VS Code interface also shows the browser's developer tools Network tab, which lists several POST requests to 'events.launchdarkly.com' with status codes like 'ERR_BLOCKED_BY_CLIENT' and '200 OK'.

List tou update horahi ha console me but DOM update nhi horaha so it means again we need a state for items list.

```
3
4  const App = () => {
5    const [inputString, setInputString] = useState("");
6    const [items, setItems] = useState([
7      "Jeans",
8      "Jackets",
9      "Caps",
10     "Shirt",
11     "Jeans",
12   ]);
13   const handleAddItem = () => {
14     const newItemsArr = [...items, inputString];
15     setItems(newItemsArr);
16     setInputString("");
17   };
18
19   return (
20     <div>
21       <input
22         type="text"
23         placeholder="New Item Name"
24         value={inputString}
25         onChange={(e) => {
26           setInputString(e.target.value);
27         }}
28       />
29       <button onClick={handleAddItem}>Add Item</button>
30     </div>
31   );
32 }
```

value ={} --> controlled input

To dynamically add items in the list and display:

```
File Edit Selection View Go Run Terminal Help react-proj-1
App.jsx Counter.jsx Counter.css Settings main.jsx
src > App.jsx
1 import Counter from "./components/Counter"
2 import { useState } from "react";
3
4 const App = () => {
5   const [inputString, setInputString] = useState("");
6   const [items, setItems] = useState(["jeans", "kurti", "shirt", "caps"])
7
8   const handleAddItem = () => {
9     //...items means pehle se array me jitne bhi items hain wo khol k rakhdo
10    const newItemArr = [...items, inputString]
11    setItems(newItemArr)
12    setInputString("") //dubara input ko empty krdo
13    // items.push(inputString);
14    // console.log(items)
15  }
16
17  return (
18    <>
19      <Counter itemName="Jeans"/>
20      <Counter itemName="Shirts"/>
21      <Counter itemName="Jacket"/> */
22    <input type="text" placeholder="New Item Name" onChange={(e) => {setInputString(e.target.value)}} />
23    <button onClick={handleAddItem}>Add Item</button>
24    {items.map((itemsEntry, index) => {
25      return <Counter key={index} itemName={itemsEntry} />
26    })}
27  )
28}
29 export default App
```

Props k ander koi string bhej sakte hain , obj , array , function and even poora component bhi bhej sakte hain

K.K.B

Props k ander koi string bhej sakte hain , obj , array , function and even poora component bhi bhej sakte hain

Delete Item from Array:

Delete ka jo button h wo mere counter component me ha aur jo state update krni ha item ko remove krne ki wo app.jsx me h how to acheive this?

```
src > App.jsx > [o] App
4  const App = () => {
7
8    >   const handleAddItem = () => { ...
12
13    };
14    const deleteItemClickHandler = () => {
15      console.log("DELETE FUNCTION FROM PARENT CALLED");
16    };
17
18  return (
19    <div>
20      <input
21        type="text"
22        placeholder="New Item Name"
23        value={inputString}
24        onChange={(e) => {
25          setInputString(e.target.value);
26        }}
27      />
28      <button onClick={handleAddItem}>Add Item</button>
29      {items.map((itemEntry, index) => {
30        return (
31          <Counter
32            key={index}
33            itemName={itemEntry}
34            deleteHandler={deleteItemClickHandler}
35          />
36        );
37      });
38    }
39  );
40}
```

Function banaya maine parent me aur usey props me pass krdia child ko.

Ab jaise humne dekha tha agr func me param pass krna ho tou usey arrow function banake uske ander call krdo taake arrow fuc ka reference store ho onclick me aur wo sirf onclick pe hi call ho and then ye aik higher order function ha jb ye chalega tou ander wala function call hojayege

Ab ye kia q?

Hame ye differentiate krna tha k konse counter ka delete button click hua ha aur isko hum index se differentiate krsakte hain

```

const deleteItemClickHandler = (index) => {
  console.log("DELETE FUNCTION FROM PARENT CALLED", index)
}

return (
  <>
  {/* <Counter itemName="Jeans"/>
  <Counter itemName="Shirts"/>
  <Counter itemName="Jacket"/> */}
  <input type="text" placeholder="New Item Name" onChange={(e) => setInputString(e.target.value)} />
  <button onClick={handleAddItem}>Add Item</button>
  {items.map((itemsEntry, index) => {
    return <Counter key={index} itemName={itemsEntry} deleteHandler={() => deleteItemClickHandler(index)} />
  })}
</>
)

```

New Item Name Add Item

jeans	10	Increment	decrement	Delete
kurti	10	Increment	decrement	Delete
shirt	10	Increment	decrement	Delete

Console tab in DevTools:

- top
- Filter: at renderRootSync (react-dom_client.js?v=3f5a7725;19142:15)
- ▶ POST https://events.launchdarkly.com/events/bulk/6291303 net::ERR_BLOCKED_BY_CLIENT
- ▶ POST https://events.launchdarkly.com/events/bulk/6291303 net::ERR_BLOCKED_BY_CLIENT
- ▶ GET http://localhost:5173/src/App.jsx?t=171905205303 net::ERR_ABORTED 500 (Internal Server Error)
- ▶ [hmr] Failed to reload /src/App.jsx. This could be due to syntax errors or importing non-existent modules. (see errors above)
- ▶ POST https://events.launchdarkly.com/events/diagnostic/6291303 net::ERR_BLOCKED_BY_CLIENT
- ▶ POST https://events.launchdarkly.com/events/diagnostic/6291303 net::ERR_BLOCKED_BY_CLIENT
- ▶ GET http://localhost:5173/src/App.jsx?t=1719052026176 net::ERR_ABORTED 500 (Internal Server Error)
- ▶ [hmr] Failed to reload /src/App.jsx. This could be due to syntax errors or importing non-existent modules. (see errors above)
- ▶ POST https://events.launchdarkly.com/events/bulk/6291303 net::ERR_BLOCKED_BY_CLIENT
- ▶ POST https://events.launchdarkly.com/events/bulk/6291303 net::ERR_BLOCKED_BY_CLIENT

Performance tab in DevTools:

- Default
- Performance insights
- Components
- Profiler

Logs in DevTools:

- DELETE FUNCTION FROM PARENT CALLED 0
- DELETE FUNCTION FROM PARENT CALLED 1
- DELETE FUNCTION FROM PARENT CALLED 2

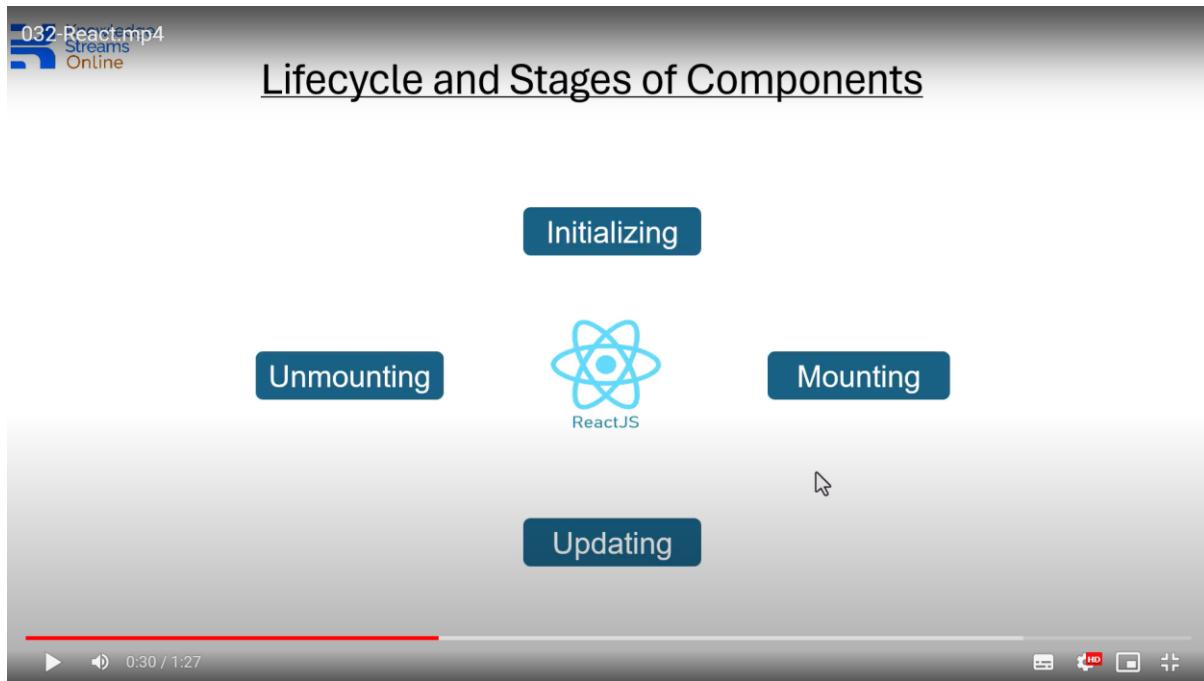
Filter method:

This is also higher order function like map , iske ander bhi aik function call hota ha.

How this func works?

Jab bhi aap kisi element k lye result True return krden wo element jake new resultant array me store hojata ah

Lifecycle of Components



Mounting: ye component build hogaya ab ye DOM k ander plugin hoga

Updating: means jaise hum state ko change krte hain tou updation hogi is phase me

Unmount: jab hum us component se move krjate hain tou wo comp un mount hojata ha

Initialize : dubara jab us component pe aayenge tou wo dubara initialize hojayega

UseEffect

UseEffect hook aik side effect run krne k kam aati ha k maine kuch operation perform kia aur uske sath me ye wala kam bhi hogaya.

So useeffect is dependent on some operation k agr ye operation perform hua tou side me ya kam bhi hojayega ya ye side effect bhi render hojayega

useEffect Hook



dependency array to contain
dependencies that when changed -
trigger the effect to run



useEffect(effect,deps);

↑
function that is to
run imperative and
likely effectful
code



▶ 1:06 / 1:50

Ab is dependency me hosakta ha ma aik variable rakhdoon x ya count aur jaise hi iski value change ho sath me effect jo k aik function h wo run hojaye

```
useEffect(()=>{},[])
```

UseEffect me 2 cheezen pass krni thi pehla agaya function , doosra dependency array

Jaise hi meri application run hoti h useEffect start me 1 dafa tou khud chalti h means:

shuroo me jab mera component mount hota ha tab useEffect khud chal jata ha 1 dafa

Case 2

The diagram illustrates the execution flow of the `useEffect` hook with a dependency array. It starts with a box labeled `[data]`, which points to a box labeled `Run at initial render`. This leads to another box labeled `Run after every re-render if data has changed since last render`. To the right of the boxes is a code snippet:

```
const [count, setCount] = useState(0);

useEffect(() => {
  // This code will run after the first render and every time 'count' changes
  console.log("Count changed:", count);
}, [count]);
```

Below the code is a video player interface showing a progress bar at 0:32 / 3:35 and some control icons.

Jaise hi dependency array me jo var ha jab wo change hoga useEffect chal jayega

```
const App = () => {
  const [inputString, setInputString] = useState("");
  const [items, setItems] = useState(["jeans", "kurti", "shirt"])

  const handleAddItem = () => { ... };
  const deleteItemClickHandler = (indexToDelete) => { ... };

  // useEffect ((() => {}, []) useEffect accepts two params 1st
  //           & 2nd is the dependency array
  useEffect [()=> {
    console.log("useEffect Ran");
  },[inputString]]
```

New Item Name | Add Item

jeans	10	Increment	decrement	Delete
kurti	10	Increment	decrement	Delete
shirt	10	Increment	decrement	Delete

Network Sources Elements **Console** Performance Application Memory Lighthouse Recorder Performance insights Components Profiler Default levels 7 1 2 Issues: 1 1 2 hidden

```
Unchecked runtime.lastError: The message port closed before a response was received.
useEffect Ran
[LaunchDarkly] LaunchDarkly client initialized
[LaunchDarkly] Opening stream connection to https://clientstream.launchdarkly.com/eval/6291303.../eyJrZXk...10jE3HDcxIzA0NzAvIDcsImV4dGVuc21vb21cnJob241011LjcuI1J9?withReasons=true
▶ POST https://events.launchdarkly.com/events/bulk/6291303... net::ERR_BLOCKED_BY_CLIENT
▶ POST https://events.launchdarkly.com/events/bulk/6291303... net::ERR_BLOCKED_BY_CLIENT
```

aik aik letter pe chal raha h | Add Item

jeans	10	Increment
kurti	10	Increment

Network Sources Elements **Console** Performance Application Memory Lighthouse Recorder

top | Filter

```
useEffect Ran
[LaunchDarkly] LaunchDarkly client initialized
[LaunchDarkly] Opening stream connection to https://clientstream.launchdarkly.com/eval/6291303.../eyJrZXk...10jE3HDcxIzA0NzAvIDcsImV4dGVuc21vb21cnJob241011LjcuI1J9?withReasons=true
▶ POST https://events.launchdarkly.com/events/bulk/6291303... net::ERR_BLOCKED_BY_CLIENT
▶ POST https://events.launchdarkly.com/events/bulk/6291303... net::ERR_BLOCKED_BY_CLIENT
▶ POST https://events.launchdarkly.com/events/bulk/6291303... net::ERR_BLOCKED_BY_CLIENT
▶ POST https://events.launchdarkly.com/events/bulk/6291303... net::ERR_BLOCKED_BY_CLIENT
② useEffect Ran
▶ POST https://events.launchdarkly.com/events/bulk/6291303... net::ERR_BLOCKED_BY_CLIENT
▶ POST https://events.launchdarkly.com/events/bulk/6291303... net::ERR_BLOCKED_BY_CLIENT
④4 useEffect Ran
▶ POST https://events.launchdarkly.com/events/diagnostic/6291303... net::ERR_BLOCKED_BY_CLIENT
▶ POST https://events.launchdarkly.com/events/diagnostic/6291303... net::ERR_BLOCKED_BY_CLIENT
③2 useEffect Ran
```

Case 1

```
25
26  useEffect(() => {
27    |   console.log("USE EFFECT RAN");
28  }, [ ]);
```

UseEffect is dependent on some array but that does not exist for now tou ye aik dafa suroo me chal jayeg bs uske baad dubara isko trigger nhi krsakte

Case 3:

```
25
26 |   useEffect(() => {
27 |     console.log("USE EFFECT RAN");
28 |   });
29 | 
```

Array ko hi khatam krda means ab ye kisi bhi cheez pe dependent nhi h jab jab component me jahan bhi change aayega useEffect render hojayega

Case 4

```
useEffect(() => [
  console.log("USE EFFECT RAN"),

  return () => {
    console.log("Clean Up Function Ran");
  },
], [items]);
```

Cleanup function

Pehli dafa jab component render hoga tou useEffect khud chal jayega but cleanup function call nhi hoga

jb useEffect 2nd time onwards chalega tou pehle cleanup function chalega then useEffect ka apna function render hoga

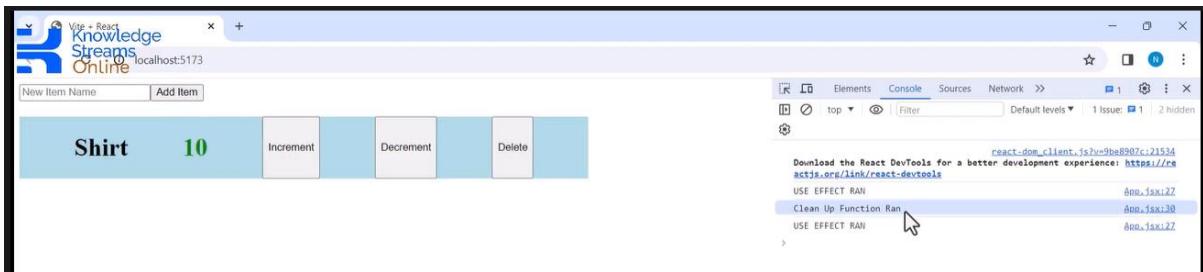
Example:

```
25
26 |   useEffect(() => [
27 |     console.log("USE EFFECT RAN");
28 |
29 |     return () => {
30 |       console.log("Clean Up Function Ran");
31 |     };
32 |   ],
33 | );
```

on first render:



then we added a shirt in item



Tou pehle cleanup func chala then useeffect

UseCase

Jab hum setTimeOut ya setInterval ka function use krte hain tou aik pg se doosre pg pe jate waqt hame wo interval clear krna parta ha k hum tou is pg se chale jayenge lakin wo interval ya time out behind the scenens browser me run hota rahe

UseCases of UseEffect Hook

1. Data fetching: Jo cheez hame general rule dekha k useEffect pehli dafa tou render hota ha ha tou isko hum whan use krsakte jahan hum ye chhaen k jaise hi page pe aaye koi data jo api se lana h wo ajey , aur ak hi dafa aaye jab tk hum us pg pe hain bar bbar api call na ho wahan use hosakta ha useeffect ka hook
2. Jb hum setTimeOut ya setInterval use krte hain tou wo browser me register hojate hain phir browser control krta ha usey , ab ager hum comp se hi shift krajayen aur p Peechay wo func chal raha ho tou kia faida isilyecleanup func k through usey stop krna bhi zaroori ha

Search Functionality

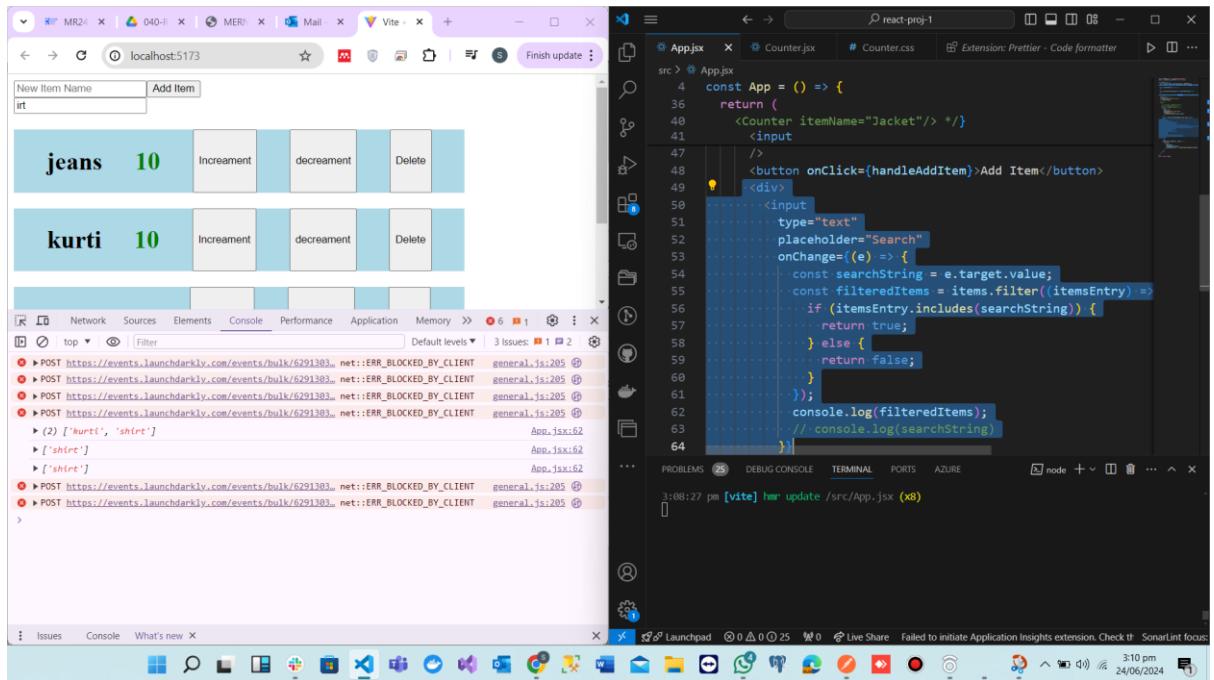
The screenshot shows a browser window on the left and a VS Code editor on the right. In the browser, there is a search input field with the placeholder "New Item Name" containing the text "kurti". Below the search bar, two items are listed: "jeans" and "kurti", each with a value of "10". Each item has three buttons: "Increment", "decrement", and "Delete". In the VS Code editor, the file "App.jsx" is open. The code includes a search input with the placeholder "Search" and an "onChange" event handler that filters an array of items based on the searchString. The code is as follows:

```
src > App.jsx
4  const App = () => {
36    return (
37      <>
39        <Counter itemName="Shirts"/>
40        <Counter itemName="Jacket"/> */
41        <input
42          type="text"
43          placeholder="New Item Name"
44          onChange={(e) => {
45            setInputString(e.target.value);
46          }}
47        />
48        <button onClick={handleAddItem}>Add Item</button>
49      </div>
50      <input type="text" placeholder="Search" onChange={(e)=> {
51        const searchString = e.target.value;
52        console.log(searchString);
53      }} />
54    </div>
55    {items.map((itemEntry, index) => {
56      return (
57        <Counter
```

.includes function

Substring search krne k kam aata h

for eg meri itemEntry h **shirt** agr searchString k ander ma **irt** type krun tou ye **true** return **krdega** b/c
irt as substring is shirt k ander aata ha



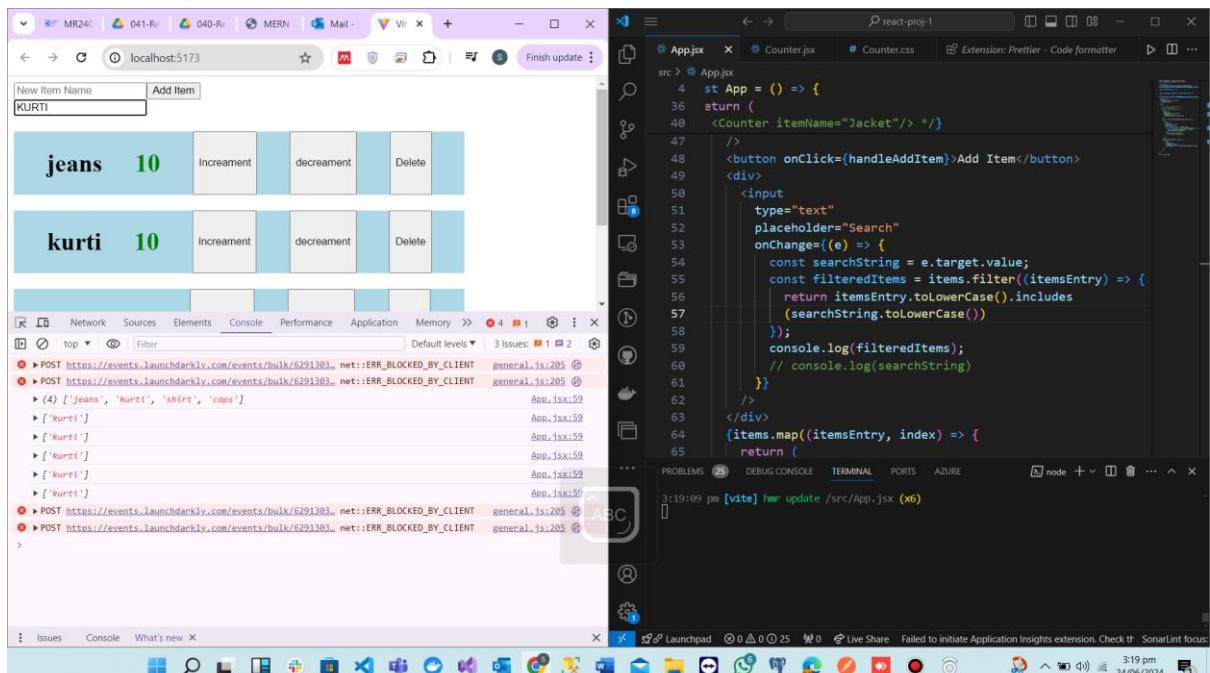
this function can be optimized

```

49   <div>
50     <input
51       type="text"
52       placeholder="Search"
53       onChange={(e) => {
54         const searchString = e.target.value;
55         const filteredItems = items.filter((itemsEntry) => {
56           return itemsEntry.includes(searchString)
57         });
58         console.log(filteredItems);
59         // console.log(searchString)
60       }}
61     />
62   </div>

```

Jahan true return hoga wo wala item array ka filtered items me store hojayega



Comparison k lye dono ko lowercase me convert krlia tou make search functionality case insensitive.

k.k.b

Agr ma apne items ki state ko hi update krwadun fiteredItems se jo user ne search kia ha tou meri items list wali state update hojayegi aur apna purana data lose krdegi and I don't want it

Aik seperate state banegi for searched items

```

2 import Counter from "./components/Counter";
3
4 const App = () => {
5   const [inputString, setInputString] = useState("");
6   const [items, setItems] = useState(["Jeans", "Shirts"]);
7   const [searchedItems, setSearchItems] = useState([]);
8
9   const handleAddItem = () => {
  
```

Returns a stateful value, and a
@version — 16.8.0
@see — <https://react.dev/referenc...>

Ab mapping searchedItems pe hogi

issue: pehle render k waqt to user ne kuch search hi nhi kia tou pg empty nazar aayega

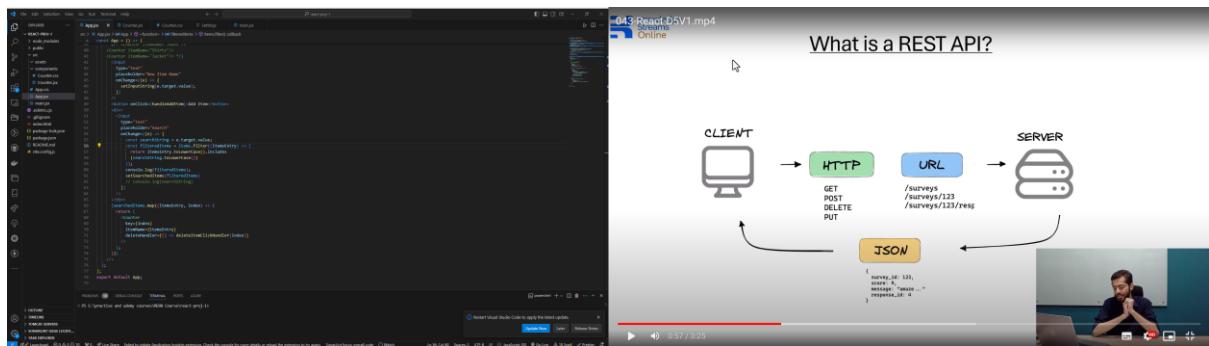
solution initial state usestate me searchedItems me items rakhden agr tou initially original data show hogta jb tk user kuch search na krle

Initial state wala masla tou hal hogaya

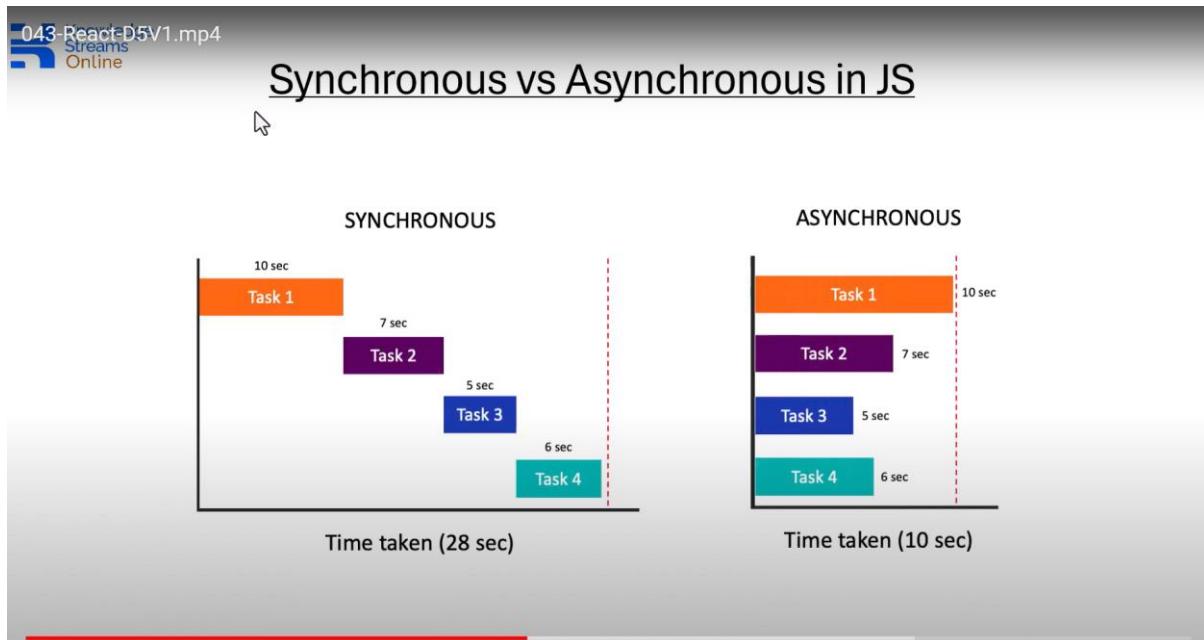
One more thing agr user koi empty string dede matlab kuch na likhe tou include ka function true return krta ha which means saare items k eleement true return honge aur original item ki array me jitne bhi elements the sab show hojayenge. This is how search functionality works

WEEK 5:

REST API:



ASYNCHRONOUS VS SYNCHRONOUS in JS



SYNCHRONOUS:

Normal javascript : pehle aik task hoga wo finish hoga phir doosra start hoga wo finish hoga

Asynchronous

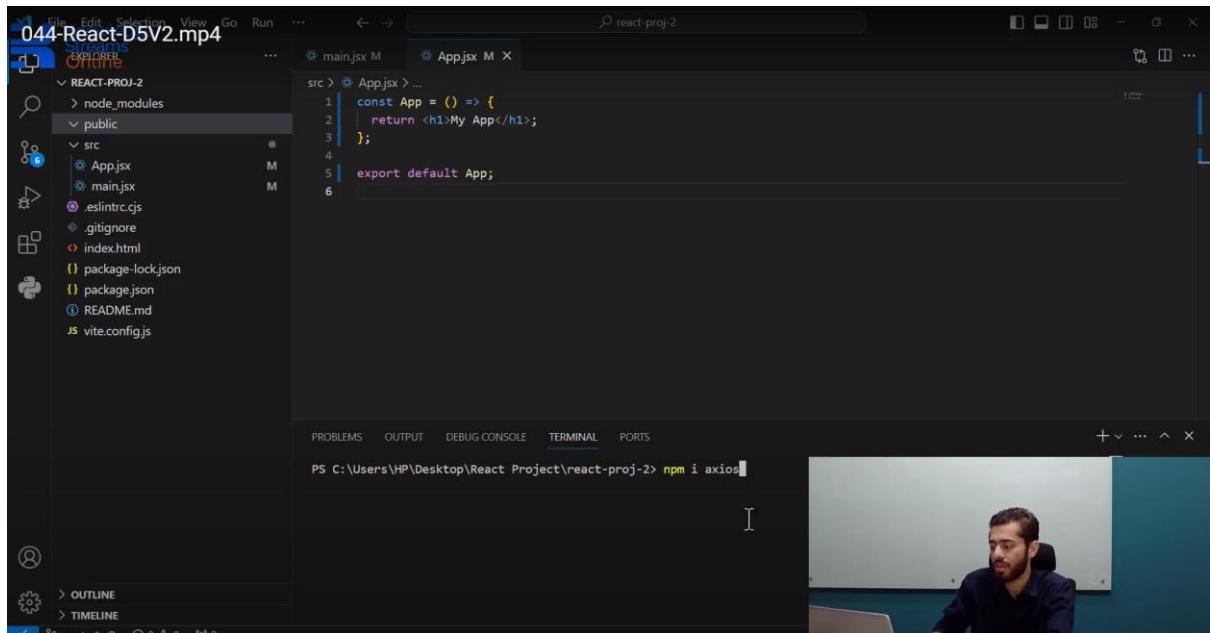
We can start multiple tasks/operations at the same time

Javascript is single threaded language : means 1 time pe aik hi operation perform krsakti h

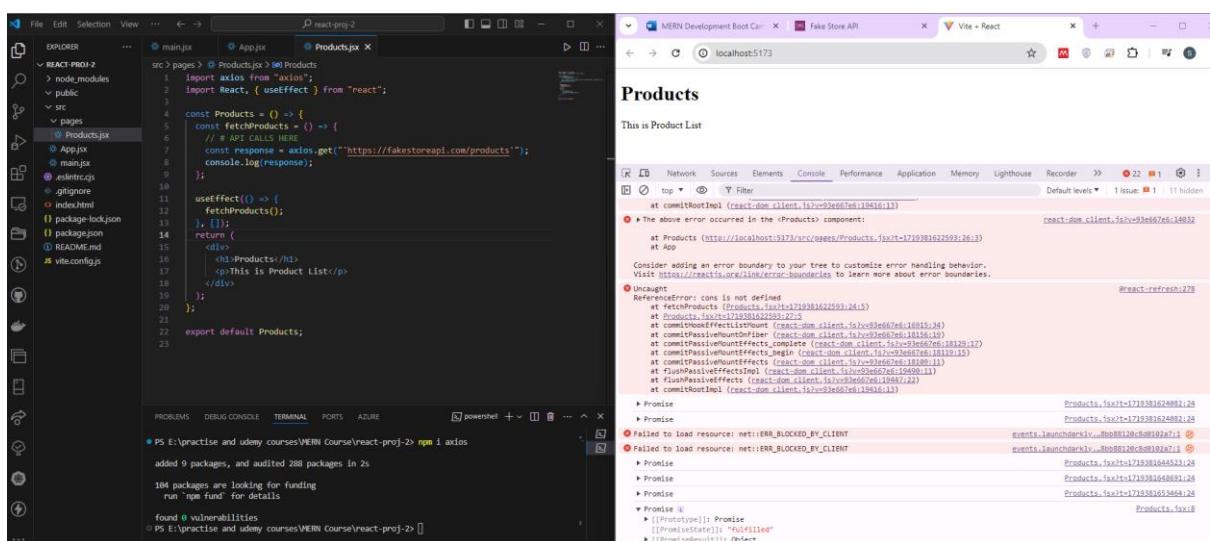
But javascript KUCH techniques ko use krti ha jisse wo ye illusion de sake k isne multiple operation at the same time perform kye hain

For example: insta page open krte hain tou front se request jati h server pe k data laake do jo latest post h wo user ko dikhani h , ab jb tk data nhi aarha aisa tou nhi h k app ruki wi ha chal nhi rahi , matlab request bhi gai wi ha user ko and simultaneously aur kam bhi horahre hain this is asynchronous behavior(reuest server ko send bhi ho aur sath sath frontend bhi chalta rahe)

Axios



The screenshot shows a video player interface with the title "044-React-D5V2.mp4". The video frame displays a person speaking. The video player has a toolbar at the top with icons for file operations, a search bar, and a progress bar. Below the video frame is a terminal window showing the command "PS C:\Users\HP\Desktop\React Project\react-proj-2> npm i axios".



The code editor shows a file named "Products.js" with the following content:

```
src > App.js > Products.js
1 import axios from 'axios';
2 import React, { useEffect } from "react";
3
4 const Products = () => {
5   const [products, setProducts] = useState([]);
6   // API CALLS HERE
7   const response = axios.get("https://fakestoreapi.com/products");
8   console.log(response);
9 }
10
11 useEffect(() => {
12   fetchProducts();
13 }, []);
14
15 render(
16   <div>
17     <h1>Products</h1>
18     <p>This is Product List:</p>
19   </div>
20 );
21
22 export default Products;
```

The terminal below shows the output of running "npm i axios":

```
PS E:\practise and udemy courses\MEIN Course\react-proj-2> npm i axios
added 9 packages, and audited 288 packages in 2s
104 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
```

axios aik promise return kr raha ha

Promise

Kuch is tarh k functions hote hain jinka response laane me kuch time lag sakte ha javascript ko jaise ye api call, ab iska reponse kitni der me aye ye pata depends on multiple factors including internet speed , backend issue etc

Tou javascript aik promise rerun krdta h

Problem is javascript does not know how much time api will take to bring results

Promise is baat ka response aajayega maybe positive or neg tou jab tk execution continue krlo ruko nhi

```

src/pages/Products/Products.jsx
  2 import React, { useEffect } from "react";
  3 
  4 const Products = () => {
  5   const [items, setItems] = useState([]);
  6   const fetchProducts = () => {
  7     // fakestore api is E-commerce website jo products ka data return krti h
  8     axios.get("https://fakestoreapi.com/products")
  9       .then((response) => {
10         // console.log(response);
11         console.log(response.data);
12       })
13       .catch((error) => {
14         console.log("Error: ", error);
15       })
16       .finally(() => {
17         console.log("Finally run this code");
18       })
19     ;
20   }
21 
22   useEffect(() => {
23     fetchProducts();
24   }, []);
25 
26   return (
27     <div>
28       <h1>Products</h1>
29       <p>This is Product List:</p>
30       {items.map((itemEntry) => {
31         return (
32           <div>
33             <h1 key={itemEntry.id}>{itemEntry.title}</h1>
34             <p>{itemEntry.description}</p>
35             <div>
36               <h2>Price: ${itemEntry.price}</h2>
37             </div>
38           </div>
39         );
40       })}
41     </div>
42   );
43 }
44 
45 export default Products;

```

PS C:\practise and udemy courses\MEAN Course\react-proj> npm i axios
added 9 packages, and audited 280 packages in 2s
104 packages are looking for funding
run npm fund for details
found 0 vulnerabilities

.then ya .catch me se koi aik chalega but finally tou her dafa chalega hi

Ab agr is data ko kisi state me ave karalen tou further manipulate krakte hain data kl

Implementation of an API Call:

```

src/pages/Products/Products.jsx
  1 import axios from "axios";
  2 import React, { useState, useEffect } from "react";
  3 
  4 const Products = () => {
  5   const [items, setItems] = useState([]);
  6   const fetchProducts = () => {
  7     // fakestore api is E-commerce website jo products ka data return krti h
  8     axios.get("https://fakestoreapi.com/products")
  9       .then((response) => {
10         // console.log(response);
11         console.log(response.data);
12         setItems(response.data);
13       })
14       .catch((error) => {
15         console.log("Error: ", error);
16       })
17       .finally(() => {
18         console.log("Finally run this code");
19       })
20     ;
21   }
22 
23   useEffect(() => {
24     fetchProducts();
25   }, []);
26 
27   return (
28     <div>
29       <h1>Products</h1>
30       <p>This is Product List:</p>
31       {items.map((itemEntry) => {
32         return (
33           <div>
34             <h1 key={itemEntry.id}>{itemEntry.title}</h1>
35             <p>{itemEntry.description}</p>
36             <div>
37               <h2>Price: ${itemEntry.price}</h2>
38             </div>
39           </div>
40         );
41       })}
42     </div>
43   );
44 }
45 
46 export default Products;

```

Jab bhi aisa data ho hamare pas jiski unique ids exist krti hun wo id use krna preferable hota ha for key

Async await

While using await our func must be async

```

const fetchProducts = async () => {
  // axios
  //   .get("https://fakestoreapi.com/products")
  //   .then((response) => {
  //     console.log(response.data);
  //     setItems(response.data);
  //   })
  //   .catch((error) => {
  //     console.log("Error: ", error);
  //   })
  //   .finally(() => {
  //     console.log("Finally run this code");
  //   });
}

const response = await axios.get("https://fakestoreapi.com/products");

```

Func k call se pehle async likhna ha iska matlab ye h k ma apne func ko ye batarahi hun k await use hua wa ha

Axios.get jo aik promise return krta ha us promise k resolve hone ka wait kro uske baad response k ander apna data pull krna

Yahan pe response me wo aayega jo .then k response me aana tha ya jo .catch k error me aana tha

Fragment

<> </>

Return ka func aik main-container return krta h islye hame multiple div ko aik main div me wrap krna parta ha fragment hamari ye req poori krta ha

(browser pe jaake dekho ye show nhi hota as div) Laikin jb component DOMme jayega wahan pe fragments render nhi hote

This is adv of using fragments and its a good practise

Why can useEffect not be async

UseEffect ka jo hook ha uske upper mere component ka render hona depend krta

Jab bhi component render hota ha pehli dafa useEffect chalta ha and then state me jab change aata ha tab chlta ha

Ab component jab pehli dafa render hua ha tou pehle useEffect ka chalna zaroori ha phir mera component DOM k ander display hona shuroo hoga ab isi component k ander maine aik api call krdi ha jiska response kab aayega mujhe pata nhi tou component proceed hi nhi hopayega generation ki tarf hi nhi jayayega q k useEffect ki execution tou abhi poori hi nhi hui

Tou ye limitation aati h useEffect ka jo function h jisey effect bolte hain wo async nhi hosakta

```
useEffect(async () => {
  try {
    const response = await axios.get("https://fakestoreapi.com/products");
    setItems(response.data);
  } catch (e) {
    console.log("Error: ", e);
  }
}, []);
```

Its not possible

Wayout: make a separate func and call in useEffect

```
const fetchProducts = async () => { ... };
useEffect(() => {
  fetchProducts();
}, []);
```

Question in mind : fetchProducts() bhi tou async h jo useEffect me call hua ha answer is useEffect chalega uska kam sirf func ko call krna ha wo func call krega and then khud shut hojayeega component aage chala jayega generation ki taraf behind the scenes wo fetchProduct chalta rahega

```
useEffect(() => {
  axios
    .get("https://fakestoreapi.com/products")
    .then((response) => {
      console.log(response.data);
      setItems(response.data);
    })
    .catch((error) => {
      console.log("Error: ", error);
    })
    .finally(() => {
      console.log("Finally run this code");
    });
}, []);
```

This can also be done .then and catch can be used in useEffect but async await k case mehamne separate func banake usko call krna hoga useeffect async nhi hosakta

Good practise: to use API Call in separate function

```

src > components > ProductCard.jsx > ProductCard
1 const ProductCard = (props) => [
2   const { item } = props;
3   return <></>;
4 ];
5
6 export default ProductCard;
7

```

Props k ander jo item name ka abject aayega usko destructure kr rahe hain aur yahn pe aik constant bana rahe hain item naam ka hi jiske ander value store hojayegi props k ander jo item naam ka object h.

The screenshot shows a development environment with multiple tabs and panes:

- Left Sidebar (File Explorer):** Shows the project structure with files like main.jsx, App.jsx, Products.jsx, and ProductCard.jsx.
- Middle Column (Code Editor):** Displays the code for `ProductCard.jsx`. The code is simple, using a functional component structure with destructuring and an export statement.
- Right Column (Browser Preview):** Shows a browser window titled "Products" with a heading "This is Product List". Below it, there are four product cards:
 - Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops** (109.95)
 - Mens Casual Premium Slim Fit T-Shirts** (22.3)
 - Mens Cotton Jacket** (55.99)
 - Mens Casual Slim Fit** (15.99)
- Bottom Right (Sidebar):** A sidebar titled "Knowledge Share Center" listing lessons from "Lesson 50 - Intro" to "Lesson 64 - Using".
- Bottom Bottom (Console):** A developer tools console showing network requests and errors, including several "ERR_BLOCKED_BY_CLIENT" errors.

The image shows a code editor (VS Code) and a web browser side-by-side. The code editor displays a file named `ProductCard.js` with the following content:

```

1 import './ProductCard.css'
2
3 const ProductCard = ({props}) => {
4   const {item} = props;
5   return (
6     <div className='cardContainer'>
7       <h3>{item.title}</h3>
8       <p>{item.price}</p>
9     </div>
10   )
11 }
12
13 export default ProductCard

```

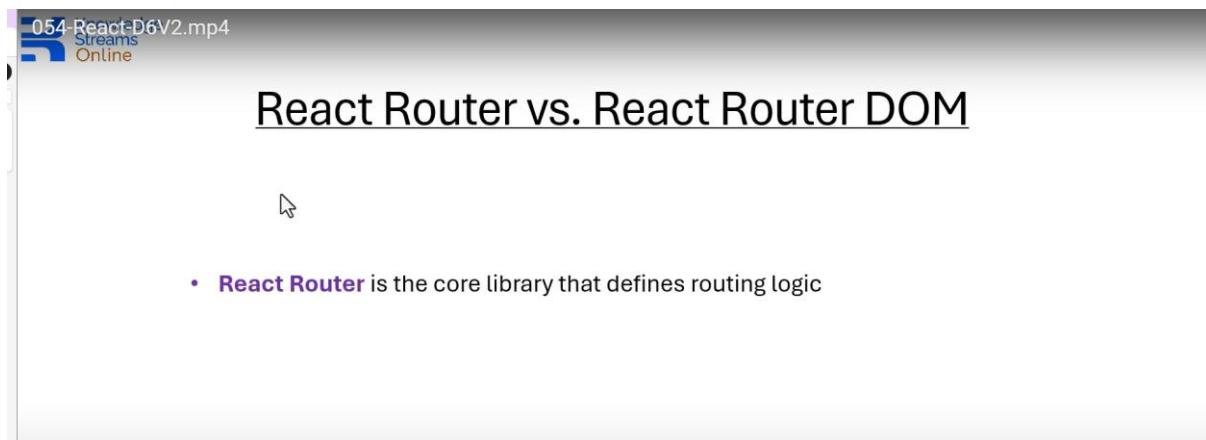
The browser window shows a list of products from a fake store API, with one item highlighted:

- Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops
- Mens Casual Premium Slim Fit T-Shirts
- Mens Cotton Jacket
- Mens Casual Slim Fit

The browser's developer tools console shows several errors related to failed reloads and network requests.

Routing

Switch from one pg to another is called routing



React Router DOM

Its extention of react-router

React Router vs. React Router DOM

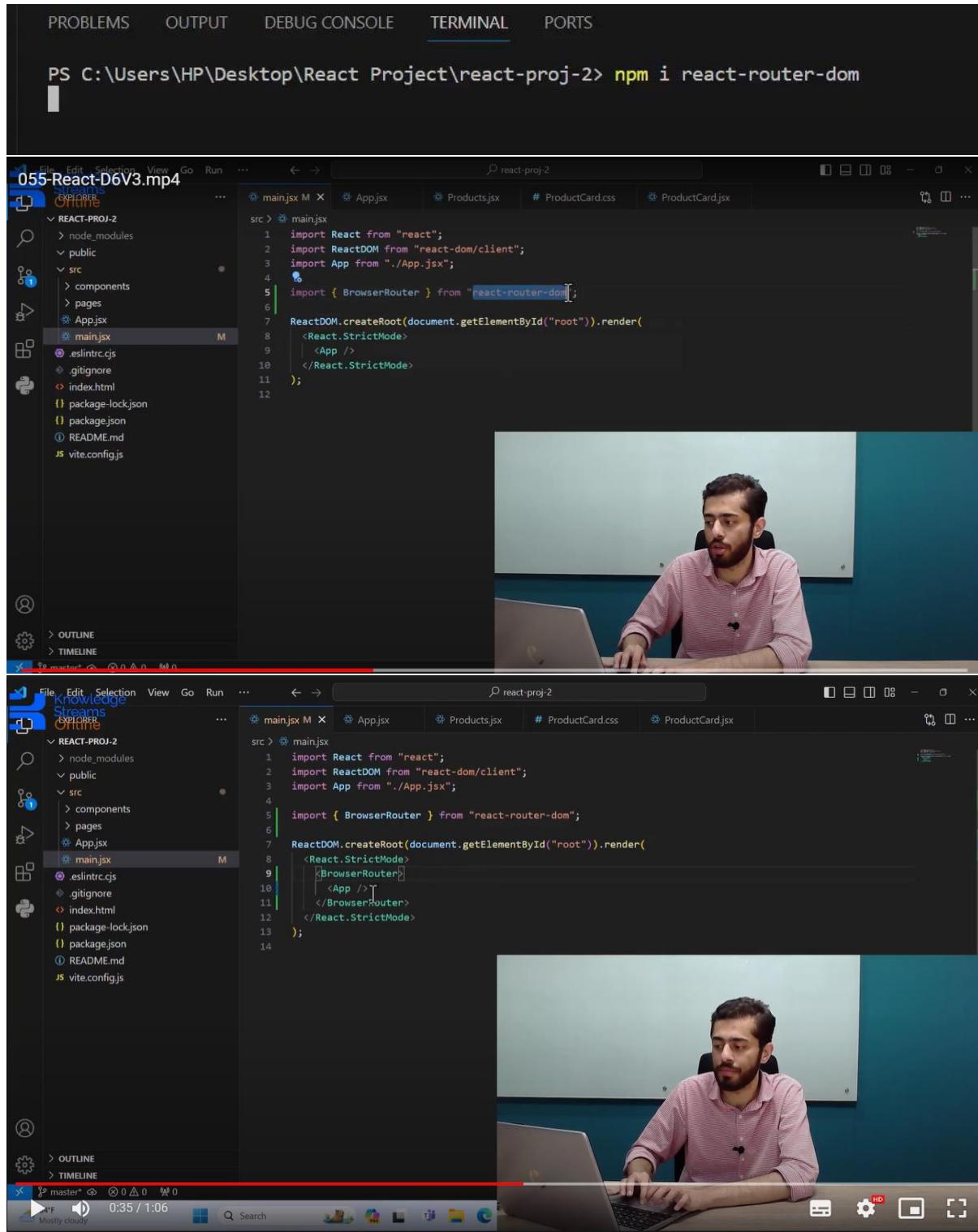


- **React Router** is the core library that defines routing logic
- **React Router DOM** is a package that provides bindings for React Router specifically tailored for web applications using React in the browser

▶ 1:35 / 2:37

HD

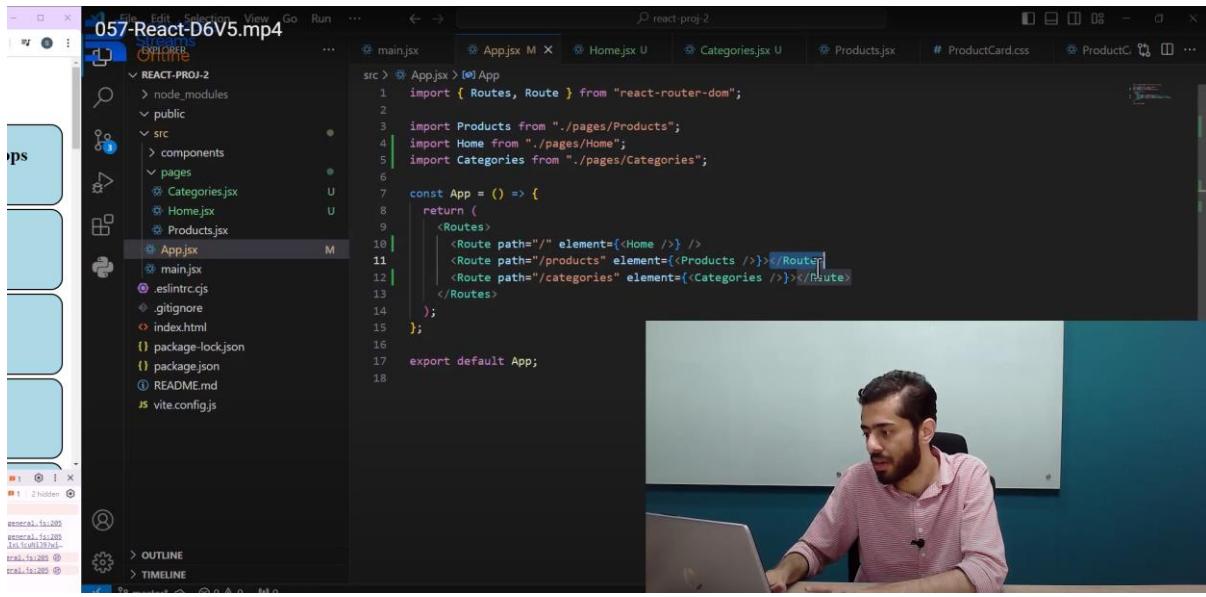
Installation



App ko `BrowserRouter` k ander wrap krdia isse adv ye hoga k mujhe apni poori app k ander browser router ki functionality ka access miljayega

Good practise

Agr hame opening and closing tag k beech me kuch nhi likhna tou usey self closing tag bana dete hain



The screenshot shows a code editor with a React project structure. The file `App.js` is open, displaying the following code:

```
src > App.js > (0) App
1 import { Routes, Route } from "react-router-dom";
2
3 import Products from "./pages/Products";
4 import Home from "./pages/Home";
5 import Categories from "./pages/Categories";
6
7 const App = () => {
8   return (
9     <Routes>
10       <Route path="/" element={<Home />} />
11       <Route path="/products" element={<Products />} />
12       <Route path="/categories" element={<Categories />} />
13     </Routes>
14   );
15 }
16
17 export default App;
```

Jahan pe koi path na ho usey index=true krdena ha

Further optimization:



The image shows two screenshots of a video call interface. In the top screenshot, a developer is shown from the chest up, wearing a pink shirt, gesturing with their hands while speaking. They are sitting at a desk with a laptop. The video call window has a title bar "react-proj-2" and several tabs for different files like main.jsx, App.jsx, Home.jsx, Categories.jsx, Products.jsx, ProductCard.css, and ProductC.css. The left sidebar shows a file tree for "REACT-PROJ-2" with folders like node_modules, public, src, components, and pages, and files like .eslintrc.js, .gitignore, index.html, package-lock.json, package.json, README.md, and vite.config.js. The bottom screenshot shows the same developer in a similar pose, continuing the explanation. The video call window title is "react-proj-2" and the tabs include main.jsx, App.jsx M X, Home.jsx U, Categories.jsx U, Products.jsx, # ProductCard.css, and ProductC.css. The file tree on the left is identical to the top screenshot.

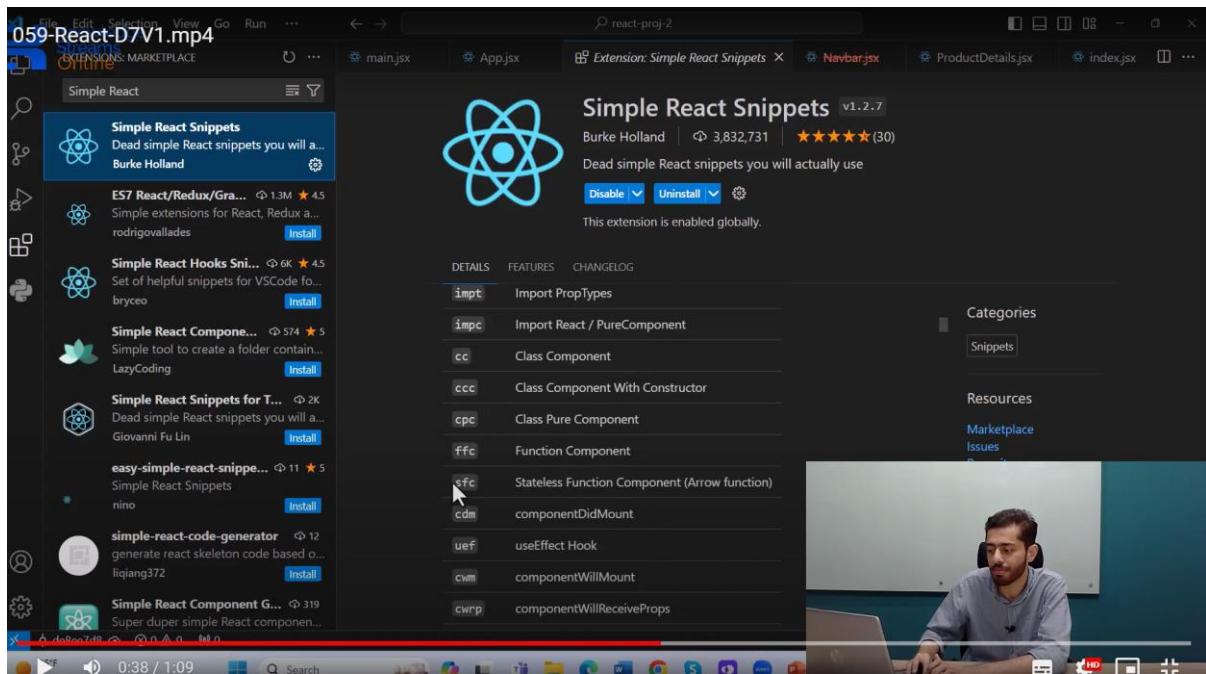
```
src > App.jsx > App
1 import { Routes, Route } from "react-router-dom";
2
3 import Products from "./pages/Products";
4 import Home from "./pages/Home";
5 import Categories from "./pages/Categories";
6
7 const App = () => {
8   return (
9     <Routes>
10       <Route index={true} element={<Home />} />
11       <Route path="/products" element={<Products />} />
12       <Route path="/categories" element={<Categories />} />
13     </Routes>
14   );
15
16   export default App;
17
```

Implementing Route for Error Page

The screenshot shows a developer's environment with three main windows:

- VS Code:** The left window displays the project structure for "react-proj-2". It includes files like App.js, main.js, and various components and pages such as Categories.jsx, HomePage.jsx, Products.jsx, and ErrorPage.jsx.
- Terminal:** The bottom-left terminal window shows npm audit results, indicating 291 packages audited with 3 vulnerabilities found.
- Browser:** The right window shows a browser tab titled "MERN Development Bo..." with the URL "localhost:5173/abc". The page title is "Error Page". The browser's developer tools Network tab shows several requests to "events.launchdarkly.com" and "events.launchdarkly.v1.com", with some errors related to "BLOCKED_BY_CLIENT".

Simple React Snippets VS Code Extension



Navbar

The screenshot shows the VS Code interface with two tabs open:

- index.jsx**: Contains the following code:

```
src > components > Navbar > index.jsx
1 import './styles.css';
2 const Navbar = () => {
3   return (
4     <div className="NavbarContainer">
5       <a className="NavLink" href="/">Home</a>
6       <a className="NavLink" href="/products">Products</a>
7       <a className="NavLink" href="/categories">Categories</a>
8     </div>
9   );
10 }
11 
12 
13 export default Navbar;
```

- App.jsx**: Contains the following code:

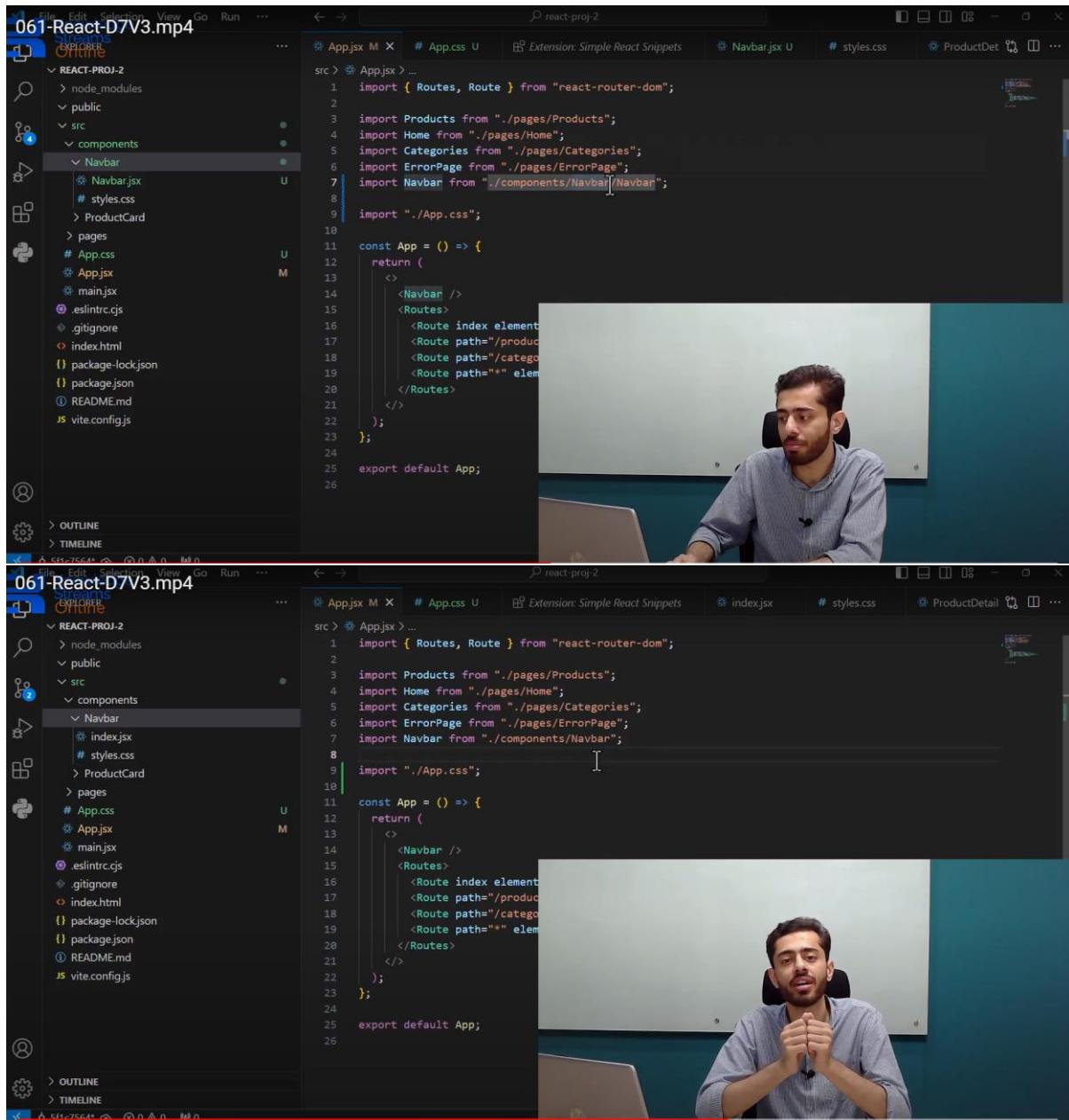
```
src > App.jsx
1 import { Route, Routes } from "react-router-dom";
2 import Products from "./pages/Products";
3 import Home from "./pages/Home";
4 import Categories from "./pages/Categories";
5 import ErrorPage from "./pages/ErrorPage";
6 import Navbar from "./components/Navbar";
7 
8 const App = () => {
9   return(
10   <>
11     <Navbar />
12     <Routes>
13       <Route path="/" element={<Home />} />
14       <Route path="/products" element={<Products />} />
15       <Route index element={<Home />} />
16       <Route path="/products" element={<Products />} />
17       <Route path="/categories" element={<Categories />} />
18       <Route path="*" element={<ErrorPage />} />
19     </Routes>
20   </>
21 }
22 
23 export default App;
```

The terminal below shows npm audit results:

```
PS E:\practise and udemy courses\VERN Course\react-proj-> npm i react-router-dom
added 3 packages, and audited 291 packages in 2s
104 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
PS E:\practise and udemy courses\VERN Course\react-proj-> []
```

The bottom status bar indicates SonarQube focus: overall code.

K.K.B



The screenshot shows a video call interface with two panes. The left pane displays the file structure of a React project named 'REACT-PROJ-2' in VS Code. The right pane shows a developer sitting at a desk, looking towards the camera. The developer is wearing a grey shirt and has a beard. The video player controls are visible at the bottom of the right pane.

File Structure (VS Code Explorer):

- REACT-PROJ-2
- node_modules
- public
- src
 - components
 - Navbar
 - pages
 - App.css
 - App.jsx
 - main.jsx
 - .eslintrc.js
 - .gitignore
 - index.html
 - package-lock.json
 - package.json
 - README.md
 - vite.config.js

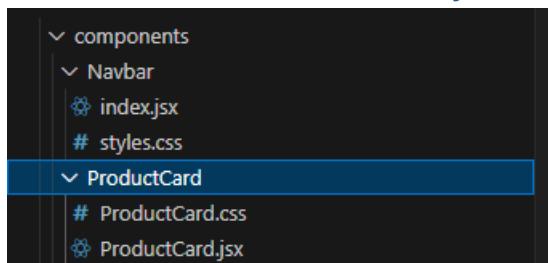
Code Editor (App.jsx):

```
App.jsx M X # App.css U Extension: Simple React Snippets Navbarjsx U # styles.css ProductDetail U ...
```

```
src > App.jsx > ...
1 import { Routes, Route } from "react-router-dom";
2
3 import Products from "./pages/Products";
4 import Home from "./pages/Home";
5 import Categories from "./pages/Categories";
6 import ErrorPage from "./pages/ErrorPage";
7 import Navbar from "./components/Navbar";
8
9 import "./App.css";
10
11 const App = () => {
12   return (
13     <>
14       <Navbar />
15       <Routes>
16         <Route index element=></Route>
17         <Route path="/products" element=><Products /></Route>
18         <Route path="/categories" element=><Categories /></Route>
19         <Route path="*" element=><ErrorPage /></Route>
20       </Routes>
21     </>
22   );
23 }
24
25 export default App;
```

Index aik aisa naam h jo by default hamara vscode pick krtा ha folder me se, agr kisi folder k ander maine index naam ki file banai wi ha tou import krte waqt sirf us folder ka naam likhna ha file wo khud pick krlega

Different Structures of Project



Both are correct.

Using NavLink instead of Anchor Tags-

We want Aik pg se doosre pe move krte waqt DOM rebuild na ho, browser pe jaake dekha jab navbar me pages pe switch kia tou DOM rebuild horaha tha pg refresh horaha tha <a> use kye we the us time

The screenshot shows the VS Code interface with an open file named 'index.jsx' containing React code. The code defines a 'Navbar' component that returns a div with three NavLink links: one to '/Home', one to '/products', and one to '/categories'. The browser window shows the 'Categories page' with a red header bar containing 'Home', 'Products', and 'Categories' links. The 'Categories' link is highlighted in red, indicating it is the active page. The browser's developer tools are open, showing the DOM structure and styles applied to the 'Categories' link and its container.

```
const Navbar = () => {
  return (
    <>
      <div className="NavbarContainer">
        <NavLink className="NavLink" to="/">Home</NavLink>
        <NavLink className="NavLink" to="/products">Products</NavLink>
        <NavLink className="NavLink" to="/categories">Categories</NavLink>
      </div>
    </>
  );
}

export default Navbar;
```

DOM Rebuild nhi horHA browser me dekha sirf jo highlighted ha sirf wo update hua pg switch krne pe

Navlink and Link difference figure out challenge

Logo pe click krne pe home pg pe aana ha

2 ways:

1st: NavLink

063-React-D7V5.mp4

```
index.jsx M # styles.css M ProductDetails.jsx ErrorPage.jsx Home.jsx Categories.jsx
src > components > Navbar > index.jsx > Navbar
  1 import { NavLink } from "react-router-dom";
  2
  3 import "./styles.css";
  4
  5 const Navbar = () => {
  6   return (
  7     <div className="navbarContainer">
  8       <div>
  9         <NavLink className="navLink" to="/">
 10           <h1>My Store</h1>
 11         </NavLink>
 12       </div>
 13       <div>
 14         <NavLink className="navLink" to="/">
 15           Home
 16         </NavLink>
 17         <NavLink className="navLink" to="/products">
 18           Products
 19         </NavLink>
 20         <NavLink className="navLink" to="/categories">
 21           Categories
 22         </NavLink>
 23       </div>
 24     );
 25   );
 26 }
 27
 28 export default Navbar;
```

2nd: useNavigate

063-React-D7V5.mp4

```
index.jsx M # styles.css M ProductDetails.jsx ErrorPage.jsx Home.jsx Categories.jsx
src > components > Navbar > index.jsx > Navbar
  1 import { Link, useNavigate } from "react-router-dom";
  2
  3 import "./styles.css";
  4
  5 const Navbar = () => {
  6   const navigate = useNavigate();
  7
  8   return (
  9     <div className="navbarContainer">
 10       <h1 onClick={() => {
 11         navigate("/");
 12       }}>
 13         My Store
 14       </h1>
 15       <div>
 16         <Link className="navLink" to="/">
 17           Home
 18         </Link>
 19         <Link className="navLink" to="/products">
 20           Products
 21         </Link>
 22         <Link className="navLink" to="/categories">
 23           Categories
 24         </Link>
 25       </div>
 26     );
 27   );
 28 }
 29
 30 export default Navbar;
```

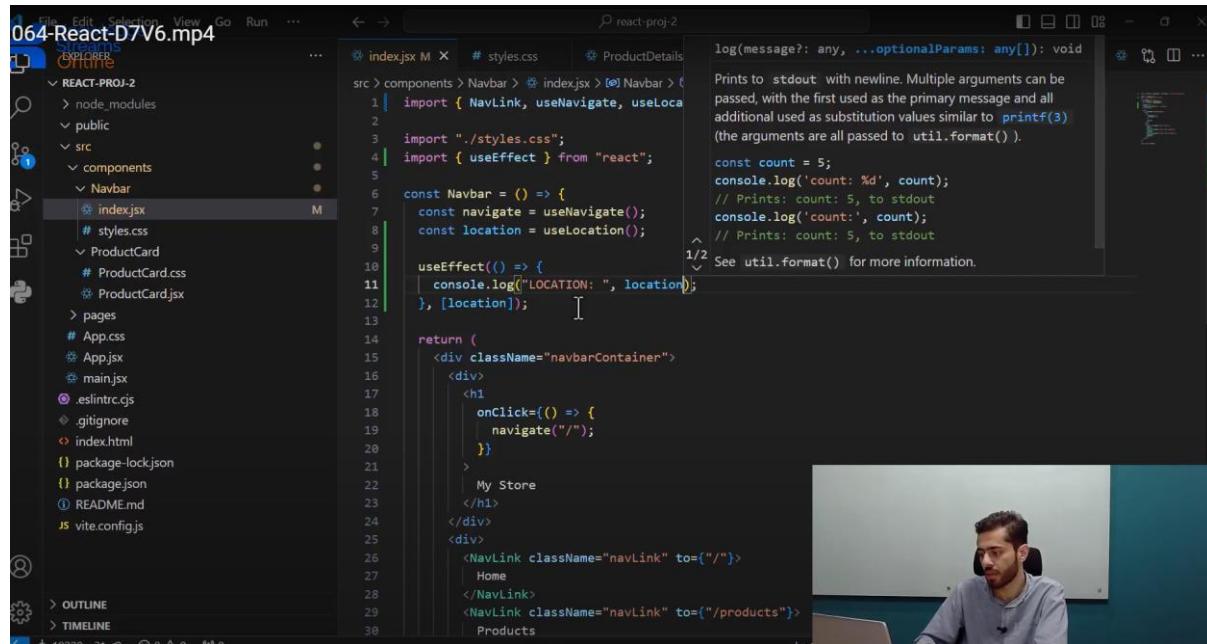
UseNavigate ki hook aik function return krti ha uska name usually hum navigate rakh dete hain.
Then navigate ko onclick pe call krado aur url pass krdo jahan navigate krna h.

Aik pg se doore pg pe jane k lye 2 methods hain:

1 is navlink agr ma kabhi hum aisa chahen k aik function k ander javascript ki execution se hum aik pg se doosre page k upper transition krjayen tou uske lye navigate ka function use krte hain jo milta ha react-router-dom k hook useNavigate se

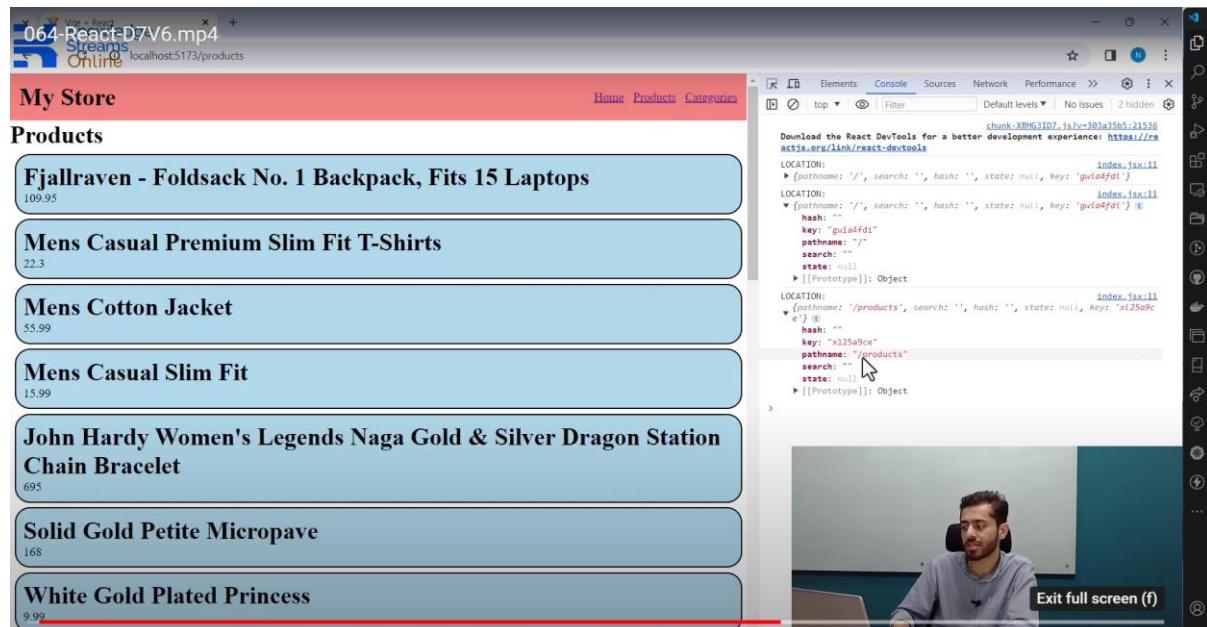
UseLocation

UseLocation hame aik object return krti ha jiska name kuch bhi hosakta ha but hum by convention location rakhte hain



```
index.jsx M # styles.css ProductDetails
src > components > Navbar > index.jsx > [e] Navbar > 0
  1 import { NavLink, useNavigate, useLoca
  2
  3 import "./styles.css";
  4 import { useEffect } from "react";
  5
  6 const Navbar = () => {
  7   const navigate = useNavigate();
  8   const location = useLocation();
  9
 10   useEffect(() => {
 11     console.log("LOCATION: ", location);
 12   }, [location]);
 13
 14   return (
 15     <div className="navbarContainer">
 16       <div>
 17         <h1>
 18           onClick={() => {
 19             navigate("/");
 20           }}
 21         >
 22           My Store
 23         </h1>
 24       </div>
 25       <div>
 26         <NavLink className="navLink" to="/">
 27           Home
 28         </NavLink>
 29         <NavLink className="navLink" to="/products">
 30           Products
 31       </div>
 32     </div>
 33   );
 34 }
 35
 36 export default Navbar;
```

UseLocation tell us currently ham kis page hain application k



My Store

Products

Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops
109.95

Mens Casual Premium Slim Fit T-Shirts
22.33

Mens Cotton Jacket
35.99

Mens Casual Slim Fit
15.99

John Hardy Women's Legends Naga Gold & Silver Dragon Station Chain Bracelet
695

Solid Gold Petite Micropave
168

White Gold Plated Princess
9.99

LOCATION: > [pathname: '/', search: '', hash: '', state: null, key: 'gviadfd1']
LOCATION: > [pathname: '/', search: '', hash: '', state: null, key: 'gviadfd1']
hash: ""
key: "gviadfd1"
 pathname: "/"
search: ""
state: null
> [[Prototype]]: Object

LOCATION: > [pathname: '/products', search: '', hash: '', state: null, key: 'sl259rc']
hash: ""
key: "sl259rc"
 pathname: "/products"
search: ""
state: null
> [[Prototype]]: Object

URL PARAMETERS

LINKEDIN pe linked.com tak url same then linkedin.com/Samara and linkedIn.com/Amna this is a dynamic url and here comes the concept of URL Parameters

```
File Edit Selection View Go ... < > react-proj-2
EXPLORER ... index.jsx 2 ProductDetails.jsx # App.css # styles.css App.jsx ...
REACT-PROJ-2
node_modules
public
src
components
  Navbar
    index.jsx
    styles.css
ProductCard
  ProductCard.css
  ProductCard.jsx
pages
  Categories.jsx
  ErrorPage.jsx
  Home.jsx
  ProductDetails.jsx
  Products.jsx
# App.css
App.jsx
main.jsx
.eslintrc.js
.gitignore
index.html
package-lock.json
package.json
README.md
vite.config.js
PROBLEMS 2 DEBUG CONSOLE TERMINAL PORTS AZURE
11:59:29 am [vite] hmr update /src/App.jsx (x5)
```

UseParams

UseParam ki jo hook h wo url me se url parameter ko access krna allow krdeti ha

```
<Route path="/categories" element={<Categories />} />
<Route
  path="/product-details/:productId"
  element={<ProductDetails />}
/>
```

:/ k aage jo bhi likha ho wo aik object me daalke productID naam ki key banake us string ko value rakho aur useParams ko pass krdo

File Edit Selection View Go Run ... ← → react-proj-2

REACT-PROJ-2

src > App.jsx # App.css Extension: Simple React Snippets index.jsx # styles.css ProductDetail

```

4 import Home from "./pages/Home";
5 import Categories from "./pages/Categories";
6 import ErrorPage from "./pages/ErrorPage";
7 import Navbar from "./components/Navbar";
8
9 import "./App.css";
10 import ProductDetails from "./pages/ProductDetails";
11
12 const App = () => {
13   return (
14     <>
15       <Navbar />
16       <Routes>
17         <Route index element={<Home />} />
18         <Route path="/products" element={<Products />} />
19         <Route path="/categories" element={<Categories />} />
20         <Route
21           path="/product-details/:productNum"
22           element={<ProductDetails />}
23         />
24         <Route path="*" element={<ErrorPage />} />
25       </Routes>
26     </>
27   );
28 }
29
30 export default App;
31

```



065-React-D7V7.mp4

localhost:5173/product-details/89

My Store

Product Details

Home Products Categories

Console

index.jsx:11

LOCATION: > {pathname: '/product-details/89', search: '', hash: '', state: null, key: 'default'}

PARAMS: > {productNum: '89'} ProductDetails.jsx:8

LOCATION: > {pathname: '/product-details/89', search: '', hash: '', state: null, key: 'default'}

PARAMS: > {productNum: '89'} ProductDetails.jsx:8

productNum: "89" [[Prototype]: Object]

File Edit Selection View Go ... ← → react-proj-2

REACT-PROJ-2

src > pages > ProductDetails.jsx # App.css # styles.css App.jsx

```

1 import React, { useEffect } from 'react';
2 import { useParams } from 'react-router-dom';
3
4 const ProductDetails = () => {
5   const params = useParams();
6
7   useEffect(() => {
8     console.log('PARAMS:', params);
9   }, [params]);
10
11   return (
12     <div>
13       <h1>Product Details</h1>
14     </div>
15   );
16 }
17
18 export default ProductDetails;
19

```

PROBLEMS DEBUG CONSOLE TERMINAL PORTS AZURE

12:06:55 pm [vite] hr update /src/pages/ProductDetails.jsx (x17)

Issues Console What's new

Fake Store API MERN Development Tools Vite + React

localhost:5173/product-details/21

My Store

Product Details

Home Products Categories

Console

index.jsx:10

LOCATION: > {pathname: '/product-details/21', search: '', hash: '', state: null, key: 'default'}

PARAMS: > {productId: '21'} ProductDetails.jsx:8

index.jsx:10

LOCATION: > {pathname: '/product-details/21', search: '', hash: '', state: null, key: 'default'}

index.jsx:10

LOCATION: > {pathname: '/product-details/21', search: '', hash: '', state: null, key: 'default'}

general:1:1x205

[LaunchDarkly] launchDarkly client initialized

general:1:1x205

[LaunchDarkly] OpenSST stream connection to https://clientstream.launchdarkly.com/eval/6291383...ev/r7XK...[0]E39CdxuABHzA/mDcsTeV4d0Qv21u3T1c...nbg24101x51u139w1t... general:1:1x205

general:1:1x205

general:1:1x205

general:1:1x205

general:1:1x205

general:1:1x205

general:1:1x205

The screenshot shows the VS Code interface with the ProductDetails.jsx file open in the editor. The code uses the useParams hook to get parameters from the URL. The browser window shows the 'Product Details' page with a red header bar.

```

src > pages > ProductDetails.jsx
1 import React, { useEffect } from 'react'
2 import { useParams } from 'react-router-dom'
3
4 const ProductDetails = () => {
5   const params = useParams();
6
7   useEffect(() => {
8     console.log("PARAMS: ", params)
9   }, [params])
10
11   return (
12     <div>
13       <h1>Product Details</h1>
14     </div>
15   )
16 }
17
18 export default ProductDetails
19
20
21
22
23
24
25
26
27
28
29
30
31

```

OPTIONAL PARAMETER

The screenshot shows the VS Code interface with the App.jsx file open in the editor. The code defines a Route for 'product-details/:productNum?' which maps to the ProductDetails component. A video call overlay is visible on the right.

```

src > App.jsx
1 import ErrorPage from "./pages/ErrorPage";
2 import Navbar from "./components/Navbar";
3
4 import "./App.css";
5 import ProductDetails from "./pages/ProductDetails";
6
7 const App = () => {
8   return (
9     <>
10       <Navbar />
11       <Routes>
12         <Route index element={<Home />} />
13         <Route path="/products" element={<Products />} />
14         <Route path="/categories" element={<Categories />} />
15         <Route
16           path="/product-details/:productNum?"
17           element={<ProductDetails />}
18         />
19         <Route path="*" element={<ErrorPage />} />
20       </Routes>
21     </>
22   );
23 }
24
25
26
27
28
29
30
31

```

Agr product details k aage kuch nhi likha wa tou bhi productdetail ka hi pg khulega aur agr koi specific product hua tou uski detail in this case

The screenshot shows a browser window displaying the product details page. The URL is 'localhost:5173/product-details'. The browser's developer tools show the params object is empty. A video call overlay is visible on the right.

```

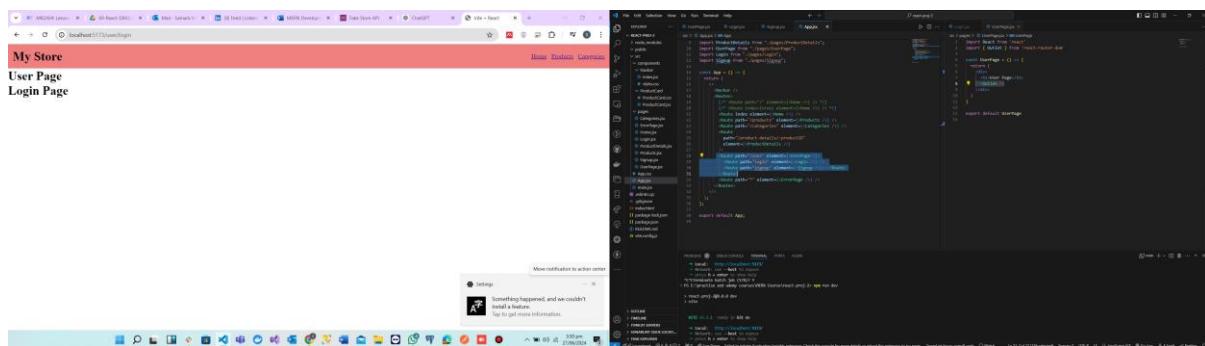
LOCATION: > (pathname: '/product-details', search: '', hash: '', state: null, key: 'default') index.js:11
PARAMS: > (productNum: '') ProductDetails.js:8
LOCATION: > (pathname: '/product-details/xyz', search: '', hash: '', state: null, key: 'default') index.js:11
PARAMS: > (productNum: 'xyz') ProductDetails.js:8
[LaunchDarkly] LaunchDarkly client initialized external.js:205
[LaunchDarkly] Opening connection to https://clients.stream.launchdarkly.com/events/0uik/5291303 net:ERR_BLOCKED_BY_CLIENT general.js:205
① POST https://events.launchdarkly.com/events/0uik/5291303 net:ERR_BLOCKED_BY_CLIENT general.js:205
② POST https://events.launchdarkly.com/events/0uik/5291303 net:ERR_BLOCKED_BY_CLIENT general.js:205

```

Laikin params ka obj empty h usme kuch bhi nhi h

Nested Routing

Outlet



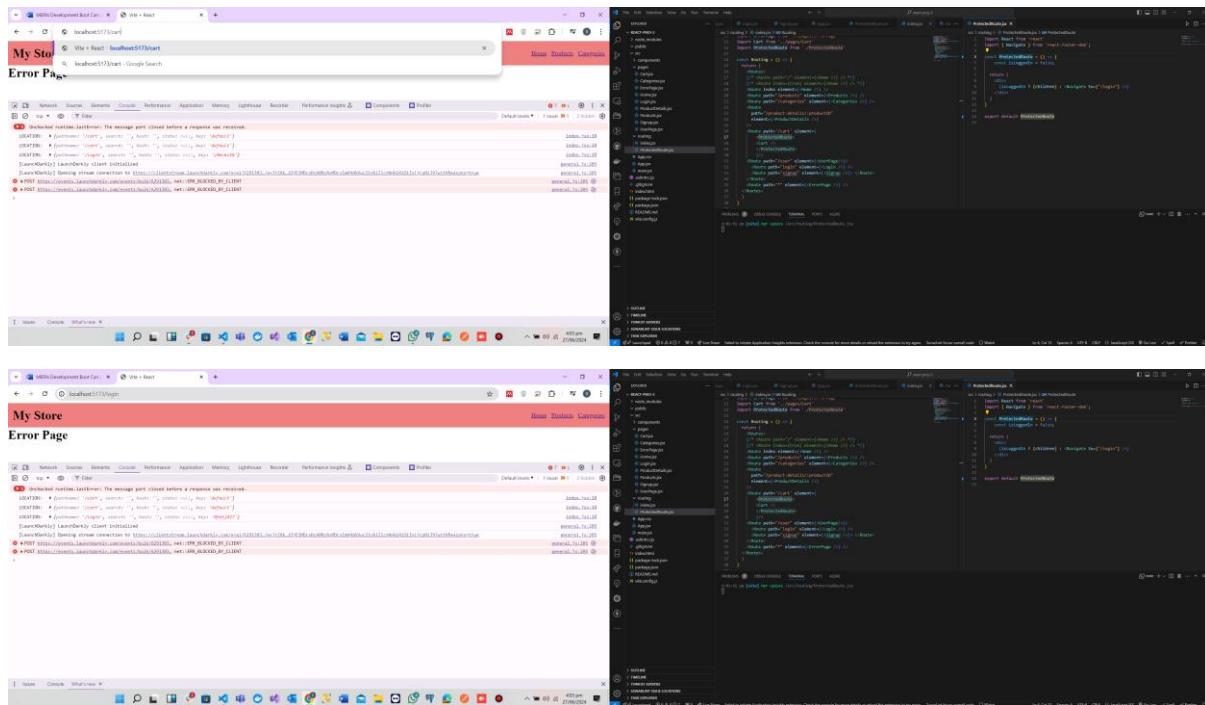
K.K.B: Nested routing me child route k path likhte waqt / nhi lagate

/outlet ki jaga nested route implement hojayega

Protected Route

</Navigate> is replace of useNavigate hook

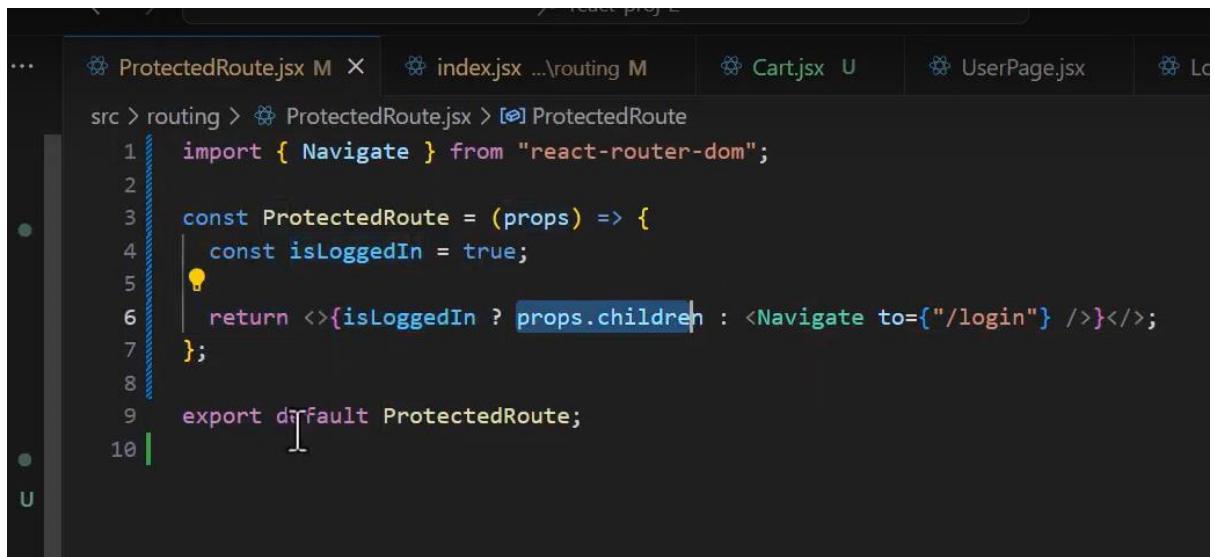
Ye aik component provide krtा ha to me jo path hoga ye uspe chala jayega



When I tried to access /cart page it redirected me to login

{children}

By default prop is children (kisi aik tag k ander itne bhi children hote hain wo sare automatically props me aate hain) , jo bhi child ha wo automatically render karado



```
src > routing > ProtectedRoute.js > [ProtectedRoute]
1 import { Navigate } from "react-router-dom";
2
3 const ProtectedRoute = (props) => {
4     const isLoggedIn = true;
5
6     return <>{isLoggedIn ? props.children : <Navigate to="/login" />}</>;
7 };
8
9 export default ProtectedRoute;
10
```

Agr tou user loged in ha tou protected route ka jo bhi child ha wo dikhado hamare case me

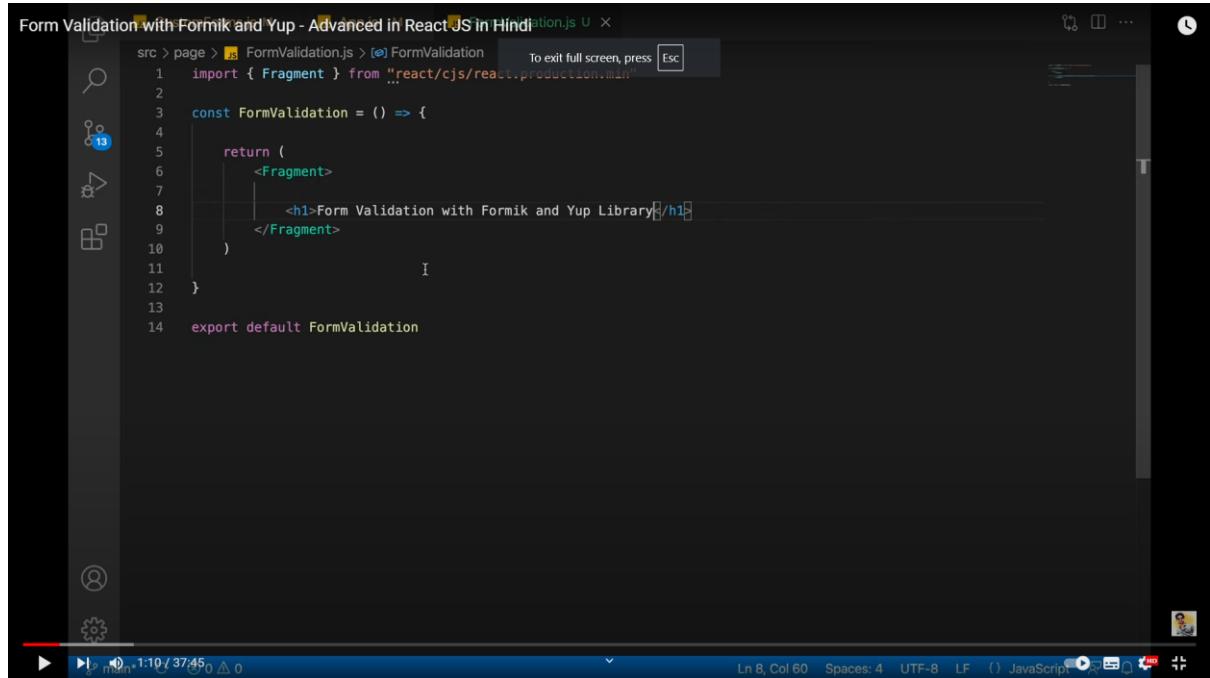


```
/>
<Route
  path="/cart"
  element={
    <ProtectedRoute>
      | <Cart />
    </ProtectedRoute>
  }
/>
```

Cart is child of protected route tou cart ka pg dikhado otherwise login ka pg dikhado

Form Validation

Formik , Yup



```
Form Validation with Formik and Yup - Advanced in ReactJS in Hindi.js U X
src > page > FormValidation.js > FormValidation
To exit full screen, press Esc
1 import { Fragment } from "react/cjs/react.production.min"
2
3 const FormValidation = () => {
4
5   return (
6     <Fragment>
7       |
8       <h1>Form Validation with Formik and Yup Library</h1>
9     </Fragment>
10  )
11
12}
13
14 export default FormValidation
```

The screenshot shows a code editor window with a dark theme. The title bar says "Form Validation with Formik and Yup - Advanced in ReactJS in Hindi.js U X". The code editor displays a single file named "FormValidation.js". The content of the file is a simple React component named "FormValidation" that returns a single

element with the text "Form Validation with Formik and Yup Library". The code is written in a modern style using the Fragment component from the React production build. The editor has a sidebar on the left with various icons and a status bar at the bottom showing file details like "Ln 8, Col 60" and "Spaces: 4".

The screenshot shows the npmjs.com package page for 'formik'. The page header includes tabs for 'React App', 'npm formik - Google Search', and 'formik - npm'. The main content area features the 'formik' logo (a blue 3D cube icon) and the text 'Build forms in React, without the tears.' Below this is a section for 'Organizations and projects using Formik' and a 'Authors' section listing Jared Palmer (@jaredpalmer) and Ian White (@eonwhite). On the right side, there's an 'Install' section with the command 'npm i formik', a 'Repository' link to github.com/formium/formik, a 'Homepage' link to formik.org, and a 'Fund this package' button. A 'Weekly Downloads' chart shows 1,630,294 downloads. Below that, details like Version 2.2.9, License Apache-2.0, Unpacked Size 580 kB, Total Files 22, Issues 551, and Pull Requests 11 are listed.

The screenshot shows the npmjs.com package page for 'yup'. The page header includes tabs for 'React App', 'npm yup - Google Search', and 'yup - npm'. The main content area features the 'yup' logo and the text 'Yup is a JavaScript schema builder for value parsing and validation. Define a schema, transform a value to match, validate the shape of an existing value, or both. Yup schema are extremely expressive and allow modeling complex, interdependent validations, or value transformations.' Below this is a 'Docs' section with links to 'API', 'Extending yup', 'TypeScript support', and 'Playground'. On the right side, there's an 'Install' section with the command 'npm i yup', a 'Repository' link to github.com/jquense/yup, a 'Homepage' link to github.com/jquense/yup, and a 'Weekly Downloads' chart showing 4,142,237 downloads. Below that, details like Version 0.32.11, License MIT, Unpacked Size 320 kB, Total Files 118, Issues 273, and Pull Requests 53 are listed.

Formik (3 things req)

1. Initial schema
2. Validation schema
3. .

```
src > page > FormValidation.js [FormValidation]
1 import { Formik, Form, Field } from "formik"
2 import { Fragment } from "react/cjs/react.production.min"
3
4 const FormValidation = () => {
5
6   const defaultValue = {
7     name: "",
8     emaill: "",
9     password: ""
10  }
11
12  return (
13    <Fragment>
14      <h1>Form Validation with Formik and Yup Library</h1>
15
16      <Formik>
17        <Form>
18          <Field type="text" name="name" placeholder="Enter Your name" />
19          <Field type="text" name="email" placeholder="Enter Your name" />
20          <Field type="text" name="password" placeholder="Enter Your name" />
21
22        </Form>
23      </Formik>
24    </Fragment>
25  )
26
27 }
28
29 export default FormValidation
```

Note : jo value key ki ha wohi name= "key" aayegi

Another video tutorial for Formik

```
React Formik Tutorial with Yup (React Form Validation)
EXPLORER FORMIK-TUTORIAL
OPEN EDITORS src > App.js > App
src > App.js src > App
src > BasicForm.js src/compo...
src > AdvancedForm.js src/c...
node_modules
public
  favicon.ico
  index.html
src
  components
    AdvancedForm.js
    BasicForm.js
    App.css
  App.js
  index.css
  index.js
  .gitignore
  package-lock.json
  package.json
  README.md

function App() {
  const [view, setView] = useState("basic");
  return (
    <div className="App">
      <nav>
        <h3 onClick={() => setView("basic")}>
          style={{ color: view === "basic" ? "#ffff" : "" }}>
          Basic
        </h3>
        <h3 onClick={() => setView("advanced")}>
          style={{ color: view === "advanced" ? "#ffff" : "" }}>
          Advanced
        </h3>
      </nav>
      {view === "basic" ? <BasicForm /> : <AdvancedForm />}
    </div>
  );
}

export default App;
```

webpack compiled successfully

```

const BasicForm = () => {
  const [email, setEmail] = useState("");
  return (
    <form autoComplete="off">
      <label htmlFor="email">Email</label>
      <input value={email} onChange={e => setEmail(e.target.value)} id="email" type="email" placeholder="Enter your email" />
    </form>
  );
}
export default BasicForm;

```

React Formik Tutorial with Yup (React Form Validation)

We don't have to worry for state management in formik case . Formik handles it at its own

```

import { useFormik } from "formik";
const BasicForm = () => {
  const formik = useFormik({
    validateOnChange: true,
    validateOnBlur: true,
    initialValues: {
      email: "",
    },
    validationSchema: Yup.object().shape({
      email: Yup.string()
        .required("Email is required")
        .email("Email is invalid"),
    }),
    onSubmit: (values) => {
      console.log(values);
    },
  });
  return (
    <form autoComplete="off">
      <label htmlFor="email">Email</label>
      <input id="email" type="email" {...formik.getFieldProps("email")}>
    </form>
  );
}
export default BasicForm;

```

use formic hook is just a custom react hook that returns all formic state and helpers and the helpers are basically just functions that help us to manage the form state

```

3
4 const SignupForm = () => {
5   const formik = useFormik({
6     initialValues: {
7       firstName: '',
8       lastName: '',
9       email: '',
10    },
11   onSubmitted: values => {
12     alert(JSON.stringify(values, null, 2));
13   },
14 });

```

they call use formic and then pass in an object to configure it and one of those properties is initial values and this is what is going to handle our state and then the onsubmit handler and a bunch of other fields

The screenshot shows the VS Code interface with the following details:

- Project Structure:** The Explorer view shows a project named "FORMIK-TUTORIAL" containing "node_modules", "public" (with "favicon.ico"), and "src" (containing "components", "App.css", "App.js", "index.css", "index.js", ".gitignore", "package-lock.json", "package.json", and "README.md").
- Code Editor:** The main editor window displays the file "BasicForm.js" with the following code:

```

1 import { useFormik } from "formik";
2
3 const BasicForm = () => {
4   const formik = useFormik({
5     initialValues: {
6       email: "",
7     },
8   });
9   return (
10     <form autoComplete="off">
11       <label htmlFor="email">Email</label>
12       <input
13         value={formik.values.email}
14         id="email" type="email" placeholder="Enter your email" />
15     </form>
16   );
17 };
18 export default BasicForm;

```

- Terminal:** The bottom terminal shows the output of the webpack compilation process.

The terminal output is as follows:

```

webpack compiled with 1 warning
webpack compiled successfully

```

React Formik Tutorial with Yup (React Form Validation)

```

src > components > BasicForm.js > BasicForm
1 | import { useFormik } from "formik";
2 |
3 | const BasicForm = () => {
4 |   const formik = useFormik({
5 |     initialValues: {
6 |       email: "",
7 |     },
8 |   });
9 |
10|   console.log(formik);
11|   return (
12|     <form autoComplete="off">
13|       <label htmlFor="email">Email</label>
14|       <input
15|         value={formik.values.email}
16|         onChange={formik.handleChange}
17|         id="email"
18|         type="email"
19|         placeholder="Enter your email"
20|       />
21|     </form>
22|   );
23| }
24| export default BasicForm;

```

to the console so you guys could see
the form state

Compiling...

Basic Advanced

Email

Enter your email

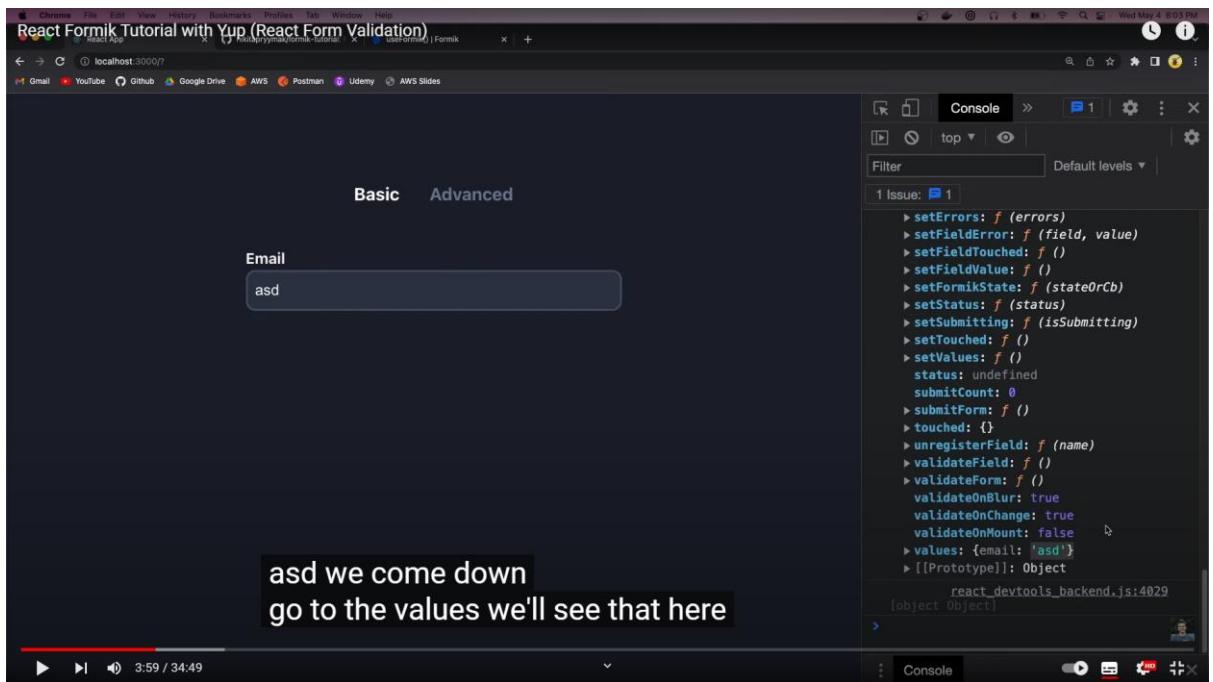
and open up the console here
okay so here's our object with the form

BasicForm.js:18

```

+ {values: {}, errors: {}, touched: {}}, stat
  us: undefined, isSubmitting: false, ...
  dirty: false
  > errors: {}
  > getFieldHelpers: f (name)
  > getFieldMeta: f (name)
  > getFieldProps: f (nameOrOptions)
  > handleBlur: f ()
  > handleChange: f ()
  > handleReset: f ()
  > handleSubmit: f ()
  > initialErrors: {}
  > initialStatus: undefined
  > initialTouched: {}
  > initialValues: {email: ''}
  > isSubmitting: false
  > isValid: true
  > isValidating: false
  > registerField: f (name, _ref3)
  > resetForm: f (nextState)
  > setErrors: f (errors)
  > setFieldError: f (field, value)
  > setFieldTouched: f ()
  > setFieldValue: f ()
  > setFormikState: f (stateOrCb)
  > setStatus: f (status)
  > setSubmitting: f (isSubmitting)
  > setTouched: f ()
  > setValues: f (O)
  status: undefined
  submitCount: 0
  > submitForm: f ()

```



React Formik Tutorial with Yup (React Form Validation)

```
onBlur={formik.handleBlur}
/>
<label htmlFor="password">Password</label>
<input
  id="password"
  type="password"
  placeholder="Enter your password"
  value={formik.values.password}
  onChange={formik.handleChange}
  onBlur={formik.handleBlur}
/>
<label htmlFor="confirmPassword">Confirm Password</label>
<input
  id="confirmPassword"
  type="password"
  placeholder="Confirm password"
  value={formik.values.confirmPassword}
  onChange={formik.handleChange}
  onBlur={formik.handleBlur}
/>
<button type="submit">Submit</button>
);
};
```

add a submit button down here
and

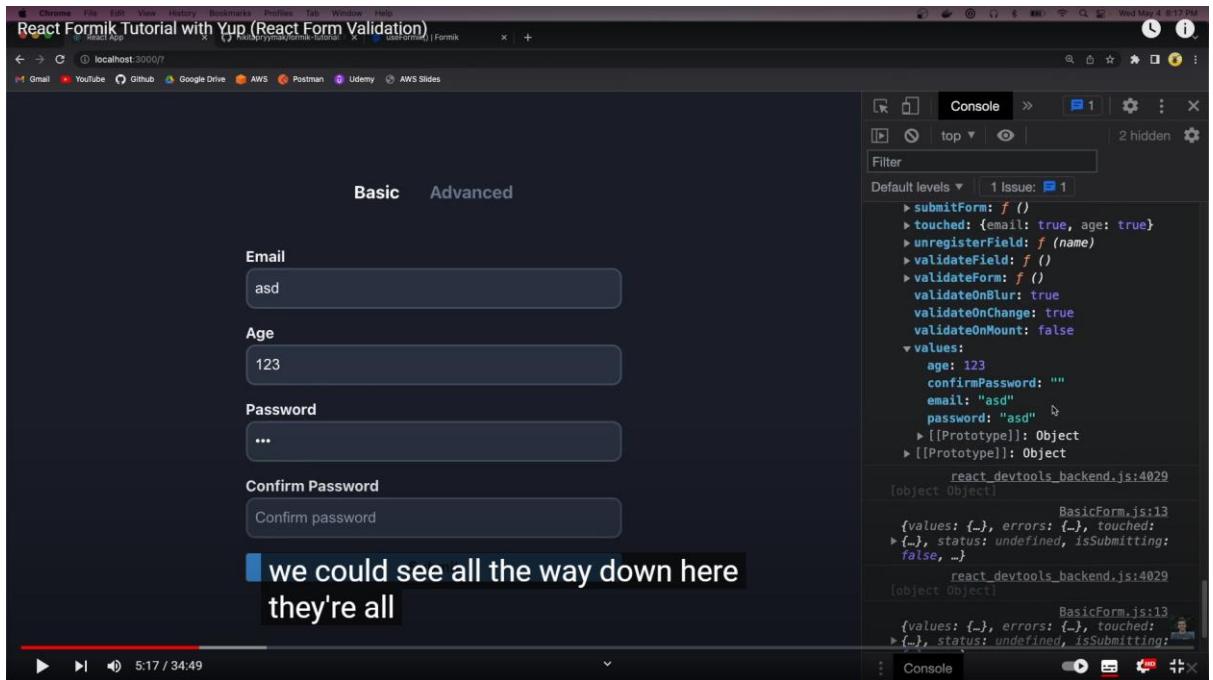
webpack compiled successfully

```
import { useFormik } from "formik";
const BasicForm = () => {
  const formik = useFormik({
    initialValues: {
      email: "",
      age: "",
      password: "",
      confirmPassword: ""
    }
  });

  console.log(formik);
  return (
    <form autoComplete="off">
      <label htmlFor="email">Email</label>
      <input
        value={formik.values.email}
        onChange={formik.handleChange}
        id="email"
        type="email"
        placeholder="Enter your email"
        onBlur={formik.handleBlur}
      />
      <label htmlFor="age">Age</label>
      <input
```

okay so now we

webpack compiled successfully



Destructure

it's kind of annoying to constantly put formic dot and then whatever method or

5:27

or property that we need so instead of just capturing it here like this we're going to destructure

5:32

and we'll just destructure all the properties and methods we need

Remove fromik. From everywhere

```

import { useFormik } from "formik";
const BasicForm = () => {
  const { values, handleBlur, handleChange } = useFormik({
    initialValues: {
      email: "",
      age: "",
      password: "",
      confirmPassword: ""
    },
    onSubmit: values
  });
  return (
    <form autoComplete="off">
      <label htmlFor="email">Email</label>
      <input
        value={values.email}
        onChange={handleChange}
        id="email"
        type="email"
        placeholder="Enter your email"
        onBlur={handleBlur}
      />
      <label htmlFor="age">Age</label>
      <input
        id="age"
        type="number"
      />
    </form>
  );
};

export default BasicForm;

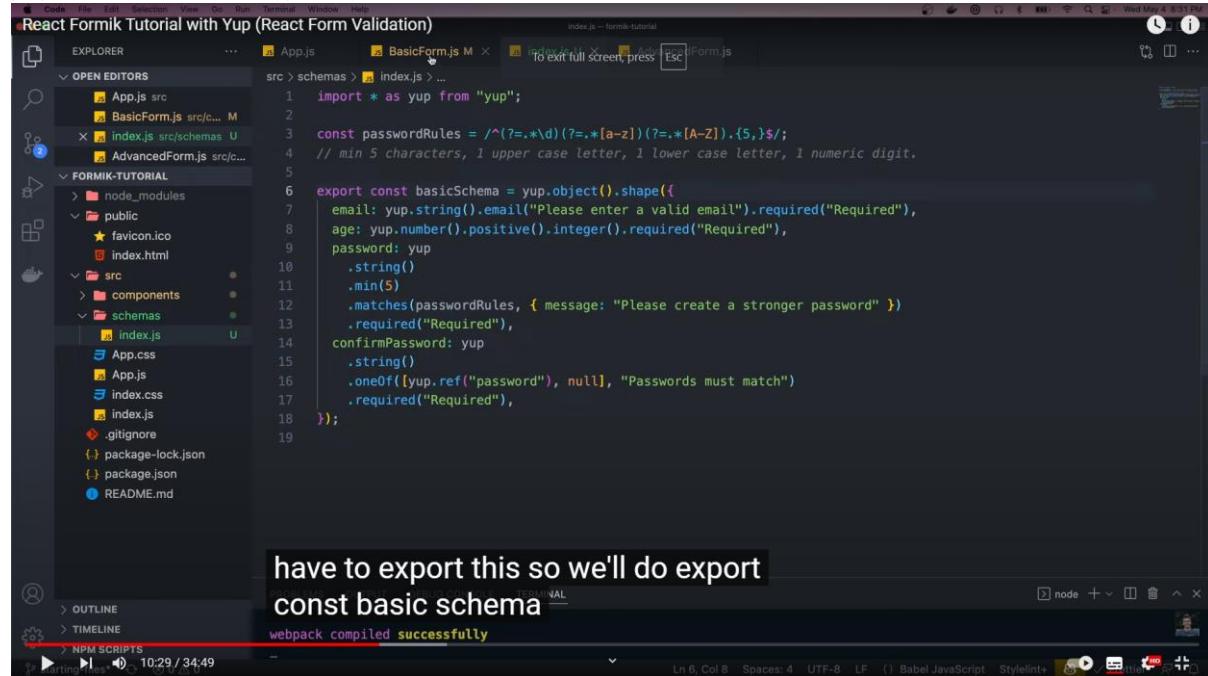
```

and if we hit save this should all

Schema

it just submits our form and it doesn't validate that we even set a password or our age or that it was a number or any of that so let's add some form validation for that we're going to be using the yup library and that works really well with formic it's actually built in to formic to use that library so in order to use it we first need to create a schema for our form and our schema is basically gonna define these different properties and it's gonna basically say what those

properties should be or what types they should be so let's go into our source folder



```
index.js (formik-tutorial) To exit full screen, press Esc

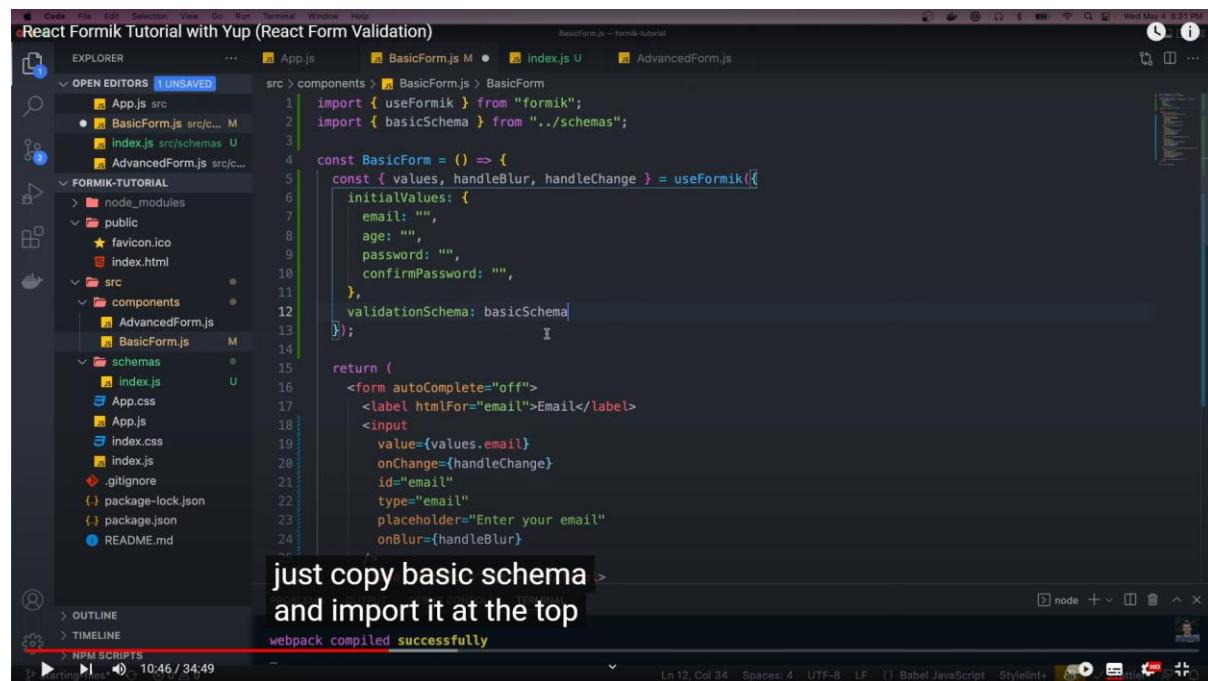
import * as yup from "yup";
const passwordRules = /^(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{5,}$/;
// min 5 characters, 1 upper case letter, 1 lower case letter, 1 numeric digit.

export const basicSchema = yup.object().shape({
  email: yup.string().email("Please enter a valid email").required("Required"),
  age: yup.number().positive().integer().required("Required"),
  password: yup
    .string()
    .min(5)
    .matches(passwordRules, { message: "Please create a stronger password" })
    .required("Required"),
  confirmPassword: yup
    .string()
    .oneOf([yup.ref("password"), null], "Passwords must match")
    .required("Required"),
});

Ln 19 / 34:49
```

have to export this so we'll do export
const basic schema

```
webpack compiled successfully
```



```
BasicForm.js (formik-tutorial)

import { useFormik } from "formik";
import { basicSchema } from "../schemas";

const BasicForm = () => {
  const { values, handleBlur, handleChange } = useFormik({
    initialValues: {
      email: "",
      age: "",
      password: "",
      confirmPassword: ""
    },
    validationSchema: basicSchema
  });

  return (
    <form autoComplete="off">
      <label htmlFor="email">Email</label>
      <input
        value={values.email}
        onChange={handleChange}
        id="email"
        type="email"
        placeholder="Enter your email"
        onBlur={handleBlur}
      >
    
```

just copy basic schema
and import it at the top

```
webpack compiled successfully
```

HandleChange

HandleBlur

Touched

Error

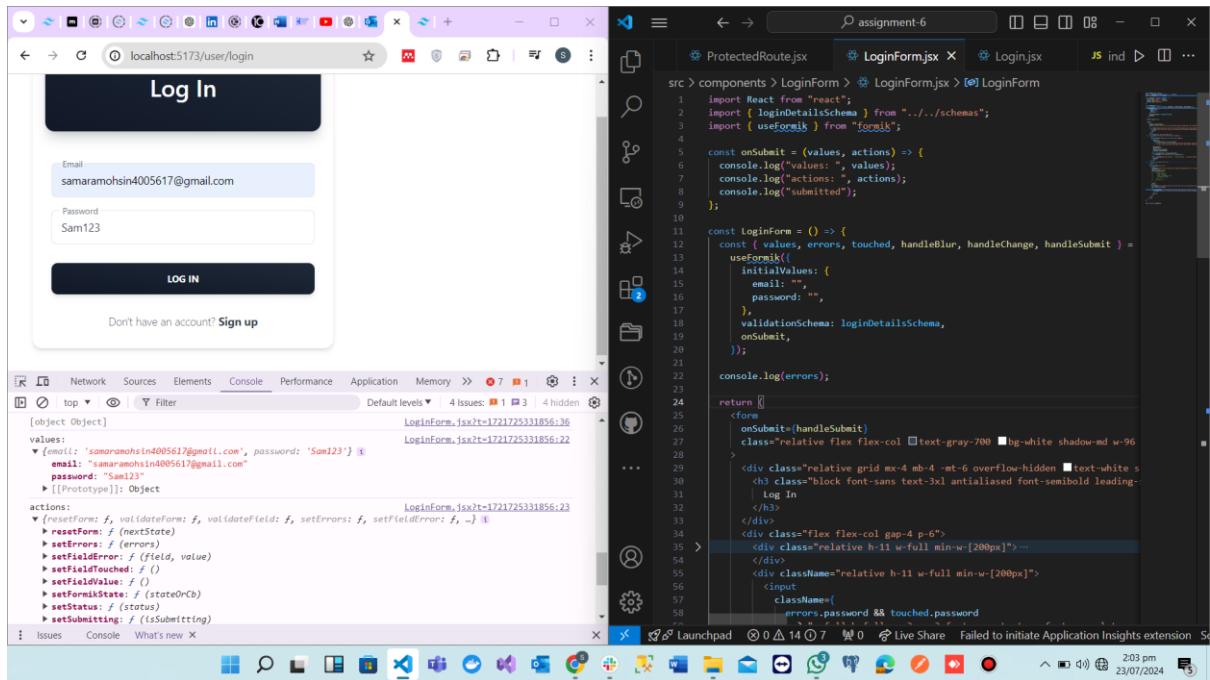
handleSubmit

The screenshot shows the VS Code interface with the title "React Formik Tutorial with Yup (React Form Validation)". The Explorer sidebar on the left lists files and folders, including App.js, BasicForm.js, App.css, index.js, and AdvancedForm.js. The BasicForm.js file is open in the editor, showing code for a form component. The code includes handling for email and age inputs, validation logic using Yup, and state management. The bottom status bar indicates "webpack compiled successfully".

```
BasicForm.js - formik-tutorial
src > components > BasicForm.js > BasicForm
  confirmPassword: '',
  validationSchema: basicSchema,
  onSubmit,
};

console.log(errors);

return (
  <form onSubmit={handleSubmit} autoComplete="off">
    <label htmlFor="email">Email</label>
    <input
      value={values.email}
      onChange={handleChange}
      id="email"
      type="email"
      placeholder="Enter your email"
      onBlur={handleBlur}
      className={errors.email && touched.email ? "input-error" : ""}>
    </label>
    <label htmlFor="age">Age</label>
    <input
      id="age"
      type="number"
      placeholder="Enter your age"
      value={values.age}
    </input>
  </form>
)
```



IsSubmitting

```
REACT FORMIK TUTORIAL WITH YUP (React Form Validation)
```

```
console.log(actions);
await new Promise((resolve) => setTimeout(resolve, 1000));
actions.resetForm();

};

const BasicForm = () => {
  const { values, errors, touched, isSubmitting, handleBlur, handleChange, handleSubmit } =
    useFormik({
      initialValues: {
        email: '',
        age: '',
        password: '',
        confirmPassword: '',
      },
      validationSchema: basicSchema,
      onSubmit,
    });

  console.log(errors);

  return (
    <form onSubmit={handleSubmit} autoComplete="off">
      <label htmlFor="email">Email</label>
      <input
        value={values.email}
        onChange={handleChange}
      >
    </form>
  );
}

BasicForm
```

webpack compiled successfully

ChartJS-React Lib for graphs

<https://www.youtube.com/watch?v=ZpfseYy5Hxg>



JS App.js JS Line.js JS Bar.js JS Pie.js

```
src > components > JS Line.js > ...
1 import { Line } from "react-chartjs-2"; 1.9k (gzipped: 874)
2 import {
3   Chart as ChartJS,
4   CategoryScale,
5   LinearScale,
6   PointElement,
7   LineElement,
8   Title,
9   Tooltip,
10  Legend,
11 } from "chart.js"; 143.2k (gzipped: 49.1k)
12
13
14
15 export const LineGraph = () => {
16   return <> Line Graph</>;
17 };
18
```

JS App.js JS Line.js X JS Bar.js JS Pie.js

```
src > components > JS Line.js > ...
3   Chart as ChartJS,
4   CategoryScale,
5   LinearScale,
6   PointElement,
7   LineElement,
8   Title,
9   Tooltip,
10  Legend,
11 } from "chart.js"; 143.2k (gzipped: 49.1k)
12
13 ChartJS.register([
14   CategoryScale,
15   LinearScale,
16   PointElement,
17   LineElement,
18   Title,
19   Tooltip,
20   Legend
21 ]);
22
23 export const LineGraph = () => {
24   return <> Line Graph</>;
25 };
```

Chart JS Tutorial - ReactJS Charts Beginner Crash Course → chart-js

JS App.js JS Line.js JS Bar.js JS Pie.js

```
src > components > JS Line.js > ...
1 import { Line } from "react-chartjs-2"; 1.9k (gzipped: 874)
2 import {
3   Chart as ChartJS,
4   CategoryScale,
5   LinearScale,
6   PointElement,
7   LineElement,
8   Title,
9   Tooltip,
10  Legend,
11 } from "chart.js"; 143.2k (gzipped: 49.1k)
12
13 ChartJS.register([
14   CategoryScale,
15   LinearScale,
16   PointElement,
17   LineElement,
18   Title,
19   Tooltip,
20   Legend
21 ]);
22
23 export const LineGraph = () =>
24   return <Line>;
```



Chart JS Tutorial - ReactJS Charts Beginner Crash Course → chart-js

JS App.js JS Line.js JS Bar.js JS Pie.js

```
src > components > JS Line.js > [e] LineGraph
1 LineElement,
2 Title,
3 Tooltip,
4 Legend,
5 } from "chart.js"; 143.2k (gzipped: 49.1k)
6
7 ChartJS.register(
8   CategoryScale,
9   LinearScale,
10  PointElement,
11  LineElement,
12  Title,
13  Tooltip,
14  Legend
15 );
16
17 export const LineGraph = () => {
18   return <Line options={} data={} />;
19 };
20
21
22
23
24
```



Chart JS Tutorial - ReactJS Charts Beginner Crash Course

```
src > JS FAKE_DATA.JS > [o] lineChartData > datasets
  1 const lineChartData = {
  2   labels: [
  3     "Monday",
  4     "Tuesday",
  5     "Wednesday",
  6     "Thursday",
  7     "Friday",
  8     "Saturday",
  9     "Sunday",
 10   ],
 11   datasets: [
 12     {
 13       label: "Steps",
 14       data: [3000, 5000, 4500, 6000, 8000, 7000, 9000],
 15       borderColor: "rgb(75, 192, 192)",
 16     },
 17   ],
 18 };
 19
```

Ln 16, Col 7 Spaces: 4 UTF-8 LF {} JavaScript ✓ Prettier



Chart JS Tutorial - ReactJS Charts Beginner Crash Course

```
src > components > JS Line.js > [o] LineGraph
  1 import React from 'react';
  2 import { Line } from 'chart.js';
  3
  4 const LineGraph = () => {
  5   const options = {};
  6
  7   return <Line options={options} data={lineChartData}>/>;
  8 }
  9
```

Ln 27, Col 53 (13 selected) Spaces: 4 UTF-8 LF {} JavaScript ✓ Prettier



Chart JS Tutorial - ReactJS Charts Beginner Crash Course

To exit full screen, press Esc

localhost:3000

Steps

9,000
8,000
7,000
6,000
5,000
4,000
3,000

Monday Tuesday Wednesday Thursday Friday Saturday Sunday

11:56 / 24:10 • Example >

Chart JS Tutorial - ReactJS Charts Beginner Crash Course

chart.js

JS App.js JS Line.js 1 JS FAKE_DATA.js X JS Bar.js JS Pie.js

src > JS FAKE_DATA.js > [o] lineChartData > [o] datasets > [o] data

```
1 export const lineChartData = {
2   labels: [
3     "Monday",
4     "Tuesday",
5     "Wednesday",
6     "Thursday",
7     "Friday",
8     "Saturday",
9     "Sunday",
10   ],
11   datasets: [
12     {
13       label: "Steps By Pedro",
14       data: [3000, 5000, 4500, 6000, 8000, 7000, 9000],
15       borderColor: "rgb(75, 192, 192)",
16     },
17     {
18       label: "Steps By Pedro's Girlfriend",
19       data: [3000, 5000, 5500, 8000, 12000, 11000, 15000],
20       borderColor: "rgb(75, 192, 192)",
21     },
22   ],
23 };
24
```

19:05 / 24:10 • Example >

Ln 19, Col 56 Spaces: 4 UTF-8 LF {} JavaScript Prettier

Up arrow icon

React with TypeScript:

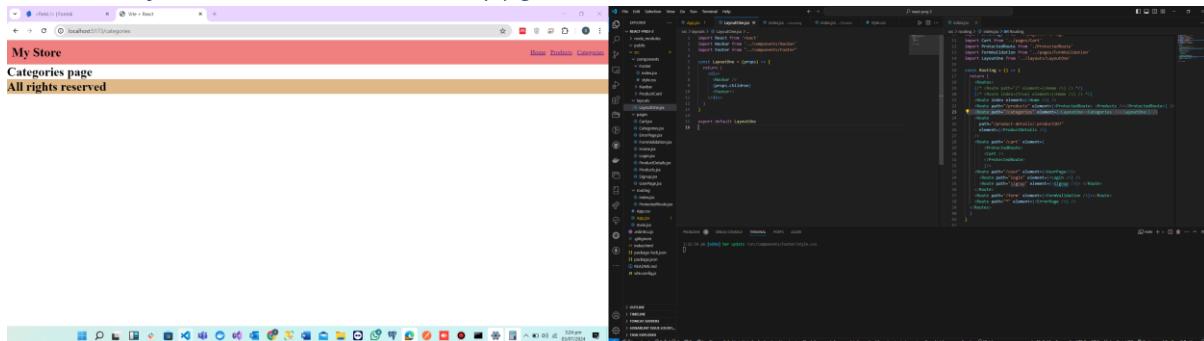
The screenshot shows a Visual Studio Code interface with a dark theme. A tooltip is displayed over the code, highlighting the `ReactNode` type used in the `TodosProviderProps` interface. The tooltip text reads: "Note that ReactNode is a generic type that covers a wide range of possible children types, including JSX elements, strings, and other React components." The code editor shows several files: `App.tsx`, `main.tsx`, `addtodo.tsx 1`, `todos.tsx 2`, and `TodosProviderProps`. The `TodosProviderProps` file contains the following code:

```
import { ReactNode, createContext } from "react";
export type TodosProviderProps = {
  children: ReactNode;
}
export const todosContext = createContext(null);
export const TodosProvidedeer = ({children}) => {
```

A video player window in the background shows a man speaking, likely the instructor.

Implementation of Layout

Navbar and footer remains static on every pg where we want like this



Tailwind vite



Tailwind CSS

```
Terminal
```

```
> npm install -D tailwindcss postcss autoprefixer
> npx tailwindcss init -p
```

3 Configure your template paths

Add the paths to all of your template files in your `tailwind.config.js` file.

```
tailwind.config.js
```

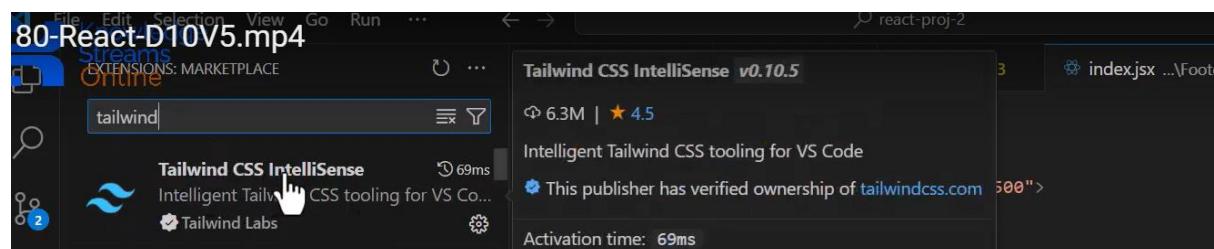
```
/** @type {import('tailwindcss').Config} */
export default {
  content: [
    "./index.html",
    "./src/**/*.{js,ts,jsx,tsx}",
  ],
  theme: {
    extend: {},
  },
  plugins: [],
}
```

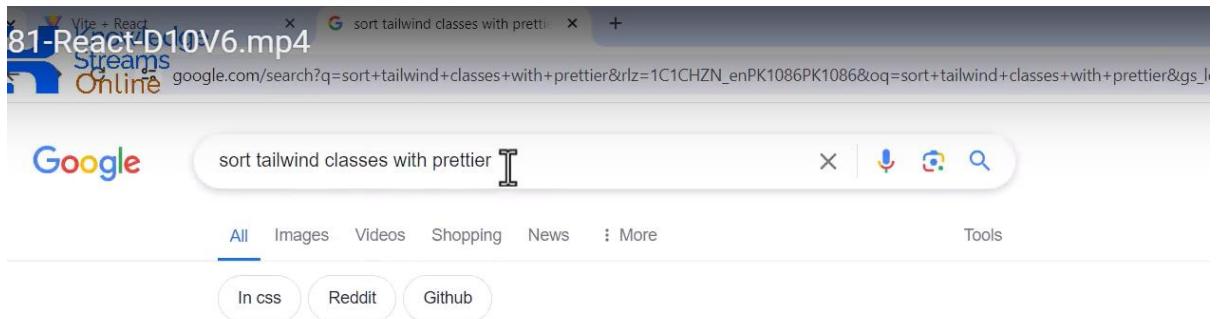
4 Add the Tailwind directives to your CSS

Add the `@tailwind` directives for each of Tailwind's layers to your `./src/index.css` file.

```
index.css
```

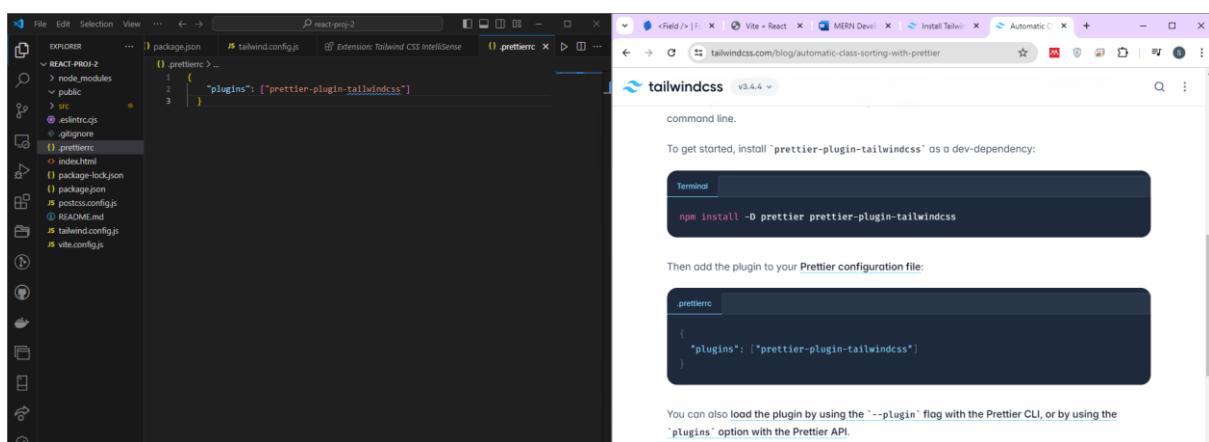
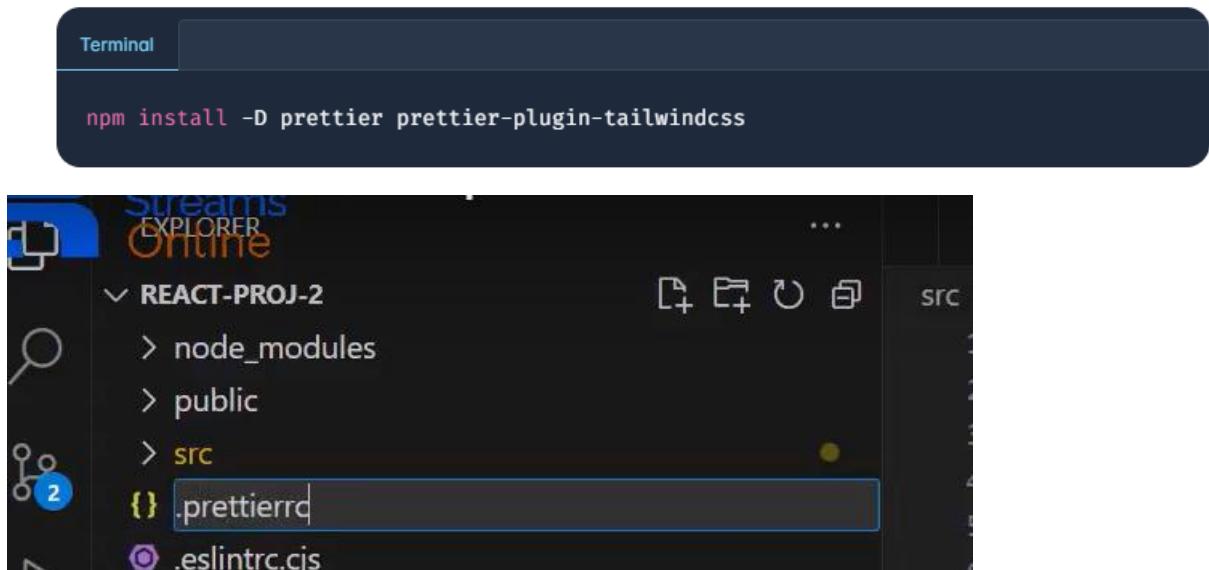
```
@tailwind base;
@tailwind components;
@tailwind utilities;
```





<https://tailwindcss.com/blog/automatic-class-sorting-with-prettier>

To get started, install `prettier-plugin-tailwindcss` as a dev-dependency:



Classes ko khudi sort krdega

Context

Aik state multiple components me needed ha tou wahan context kam aata ha.

State k lye hamne ye prha tha k state aik component ki her existance k lye memory management krti ha.

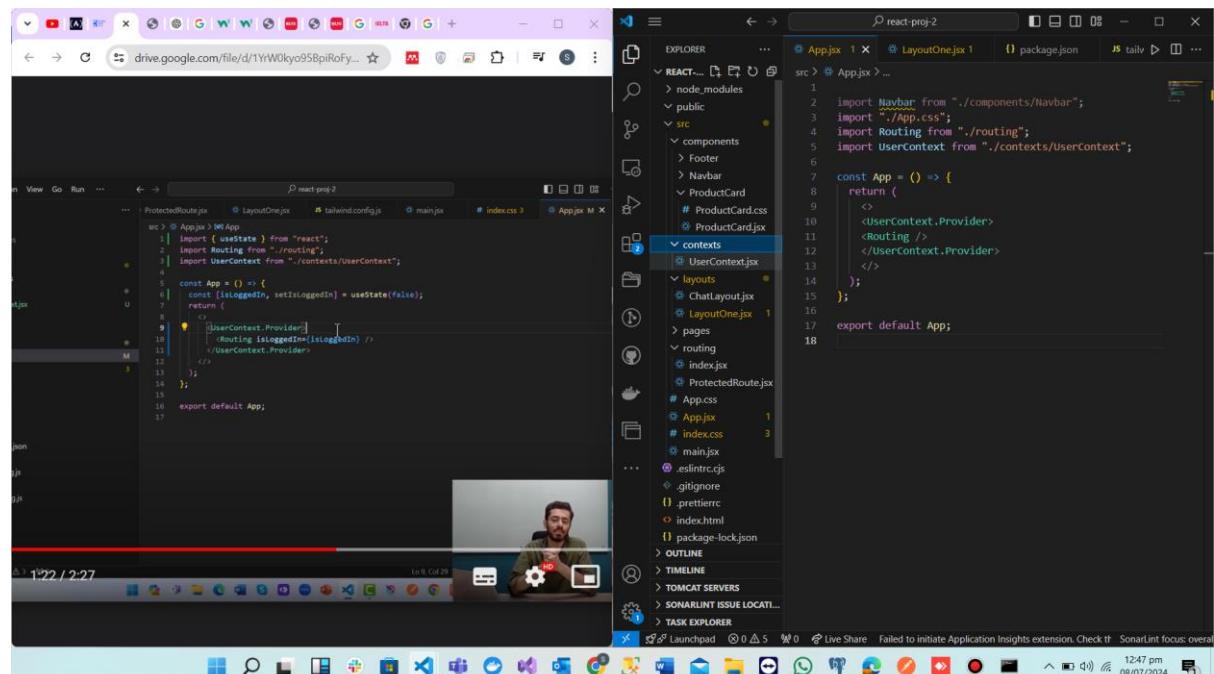
productCard ka component aik hi dafa banaya us component ko agr 10 times call kia tou 10 times

uski state alg alg banegi aur alg memory store horahi hogi
ham ye krsakte the k app.jsx me banalete aik variable aur ussko jahana access krna ha us component
me lelete but age hamare nested components hain lets say 10 aik k ander aik phir uske ander 2nd ,
2nd k ander 3rd upto so on.... tou her jaga props k through us var ko pass krna parega bhale uski
need na bhi ho component me q yhan se aage child component ko send krna ha this is called prop
drilling

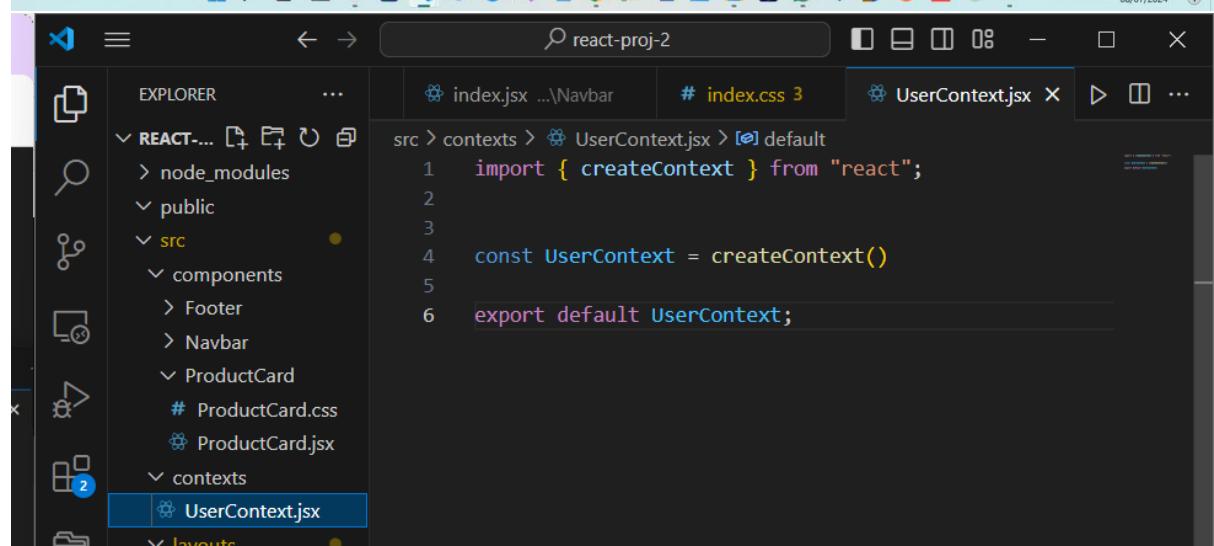
prop drilling se hamare components heavy hojate hain

we need a global state jahan se ham ksii bhi component me wo variable use krsaken

Lets see contetxt api ko use krte we global state kaise banai jati ha



```
src > App.jsx 1 x LayoutOne.jsx 1 package.json JS tailv D ...
src > App.jsx ...
1 import Navbar from "./components/Navbar";
2 import "./App.css";
3 import Routing from "./routing";
4 import UserContext from "./contexts/UserContext";
5
6 const App = () => {
7   const [isloggedIn, setIsloggedIn] = useState(false);
8   return (
9     <UserContext.Provider>
10       <Routing isloggedIn={isloggedIn} />
11     </UserContext.Provider>
12   );
13 }
14
15 export default App;
16
17
```

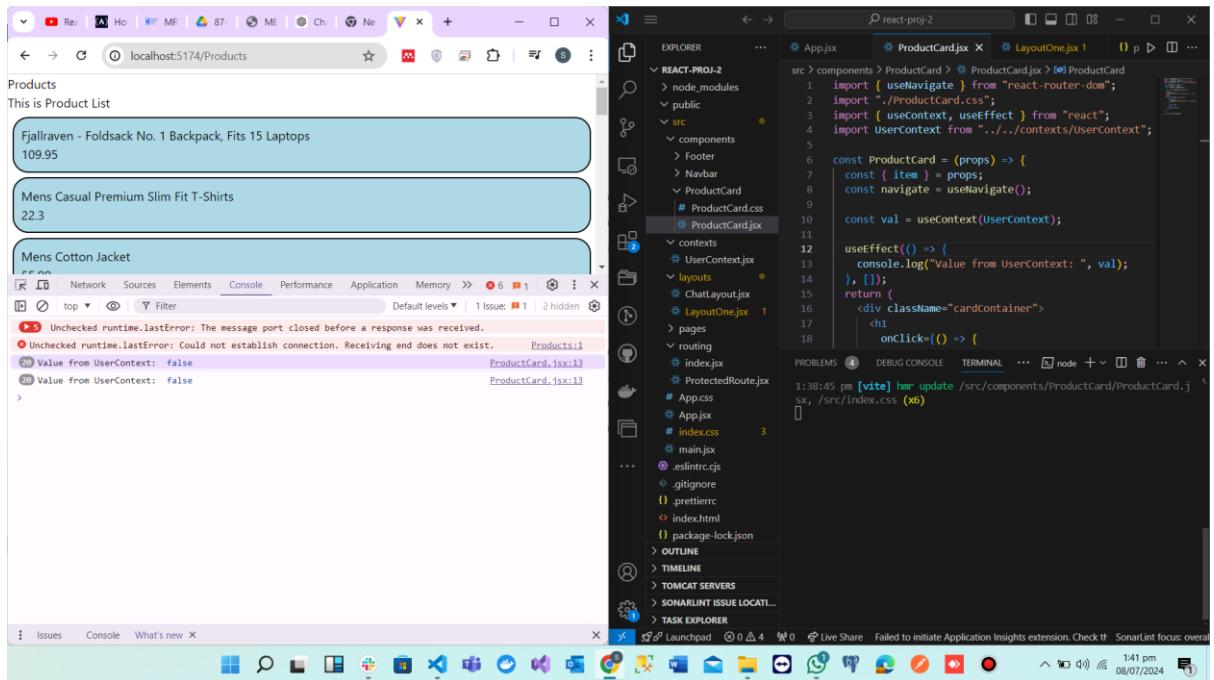


```
src > contexts > UserContext.jsx > default
1 import { createContext } from "react";
2
3
4 const UserContext = createContext();
5
6 export default UserContext;
```

Poori app ko userContext me wrap krdia app.jsx me jaake

CreateContext ka func context banane k kam aata ha

useContext usko read krne k kam aata ha



localhost:5174/Products

Products
This is Product List

- Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops
109.95
- Mens Casual Premium Slim Fit T-Shirts
22.3
- Mens Cotton Jacket
22.3

Network Sources Elements Console Performance App Issues Console What's new

```

src > components > ProductCard > ProductCard.jsx ...
1 import { useState } from "react-router-dom";
2 import "./ProductCard.css";
3 import { useContext, useEffect } from "react";
4 import UserContext from "../../contexts/UserContext";
5 import { useState, useEffect, useState } from "react";
6
7 const ProductCard = ({props}) => {
8   const [item] = props;
9   const navigate = useNavigate();
10
11   const val = useContext(UserContext);
12
13   useEffect(() => {
14     console.log("Value from UserContext: ", val);
15   }, []);
16
17   return (
18     <div className="cardContainer">
19       <a href="#" onClick={() => {
20         navigate("/product-details/" + item.id);
21       }}>
22         <h3>{item.title}</h3>
23         <p>${item.price}</p>
24       </a>
25     </div>
26   );
27 }
28
29 export default ProductCard;

```

Launchpad 0 0 0 0 Live Share Failed to initiate Application Insights extension. Check the console for more details. SonarLint focus: overall code Watch

localhost:5174/Products

Products
This is Product List

- Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops
109.95
- Mens Casual Premium Slim Fit T-Shirts
22.3
- Mens Cotton Jacket
22.3

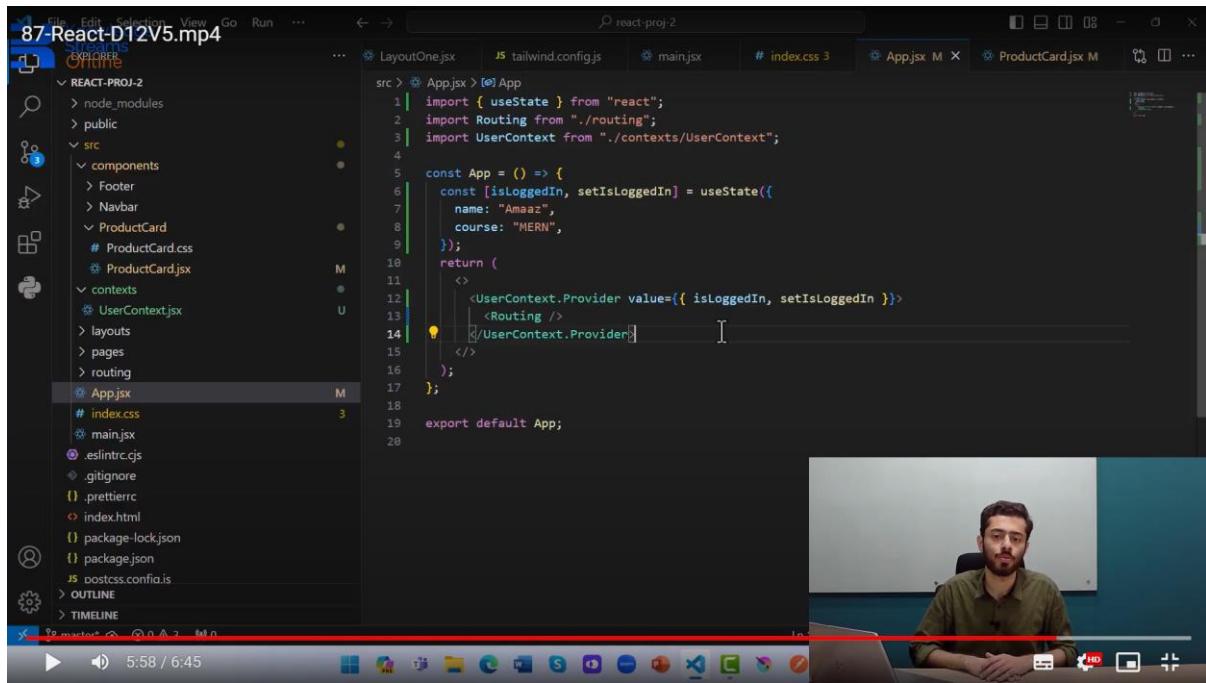
Network Sources Console Default levels 1 Issue 38 hidden Issues Console What's new

```

src > components > ProductCard > ProductCard.jsx ...
1 import { useState } from "react-router-dom";
2 import "./ProductCard.css";
3 import { useContext, useEffect } from "react";
4 import UserContext from "../../contexts/UserContext";
5
6 const ProductCard = ({props}) => {
7   const [item] = props;
8   const navigate = useNavigate();
9
10  const val = useContext(UserContext);
11
12  useEffect(() => {
13    console.log("Value from UserContext: ", val);
14  }, []);
15
16  return (
17    <div className="cardContainer">
18      <a href="#" onClick={() => {
19        navigate("/product-details/" + item.id);
20      }}>
21        <h3>{item.title}</h3>
22        <p>${item.price}</p>
23      </a>
24    </div>
25  );
26 }
27
28 export default ProductCard;

```

Launchpad 0 0 0 0 Live Share Failed to initiate Application Insights extension. Check the console for more details. SonarLint focus: overall code Watch



The screenshot shows a video call interface with a developer's face on the right and their computer screen on the left. The computer screen displays a Windows desktop with a taskbar at the bottom. An open instance of VS Code is showing the file 'App.jsx'. The code in the editor is:

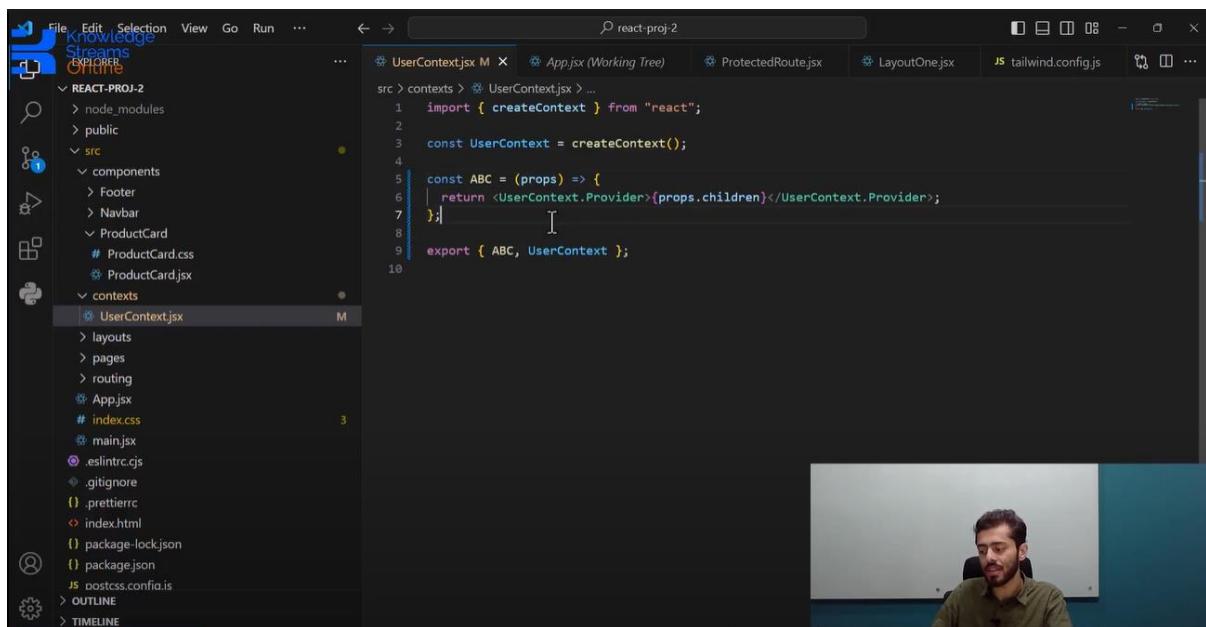
```
src > App.jsx > App
1 | import { useState } from "react";
2 | import Routing from "./routing";
3 | import UserContext from "./contexts/UserContext";
4 |
5 | const App = () => {
6 |   const [isLoggedIn, setIsLoggedIn] = useState({
7 |     name: "Amaaz",
8 |     course: "MERN",
9 |   });
10 |   return (
11 |     <>
12 |       <UserContext.Provider value={{ isLoggedIn, setIsLoggedIn }}>
13 |         <Routing />
14 |       </UserContext.Provider>
15 |     </>
16 |   );
17 | }
18 |
19 | export default App;
20 |
```

Ham na sirf read krsakte hain throughout the application jo global state me ha balke right bhi krwaskate hain aur wo jo prop drilling krni prh rahi thi bhtt ziada wo issue bhi resolved.

Note: jo components wrap hue we hain provider me sirf unhi ko access hogा value ka

Creating a Component for Provider

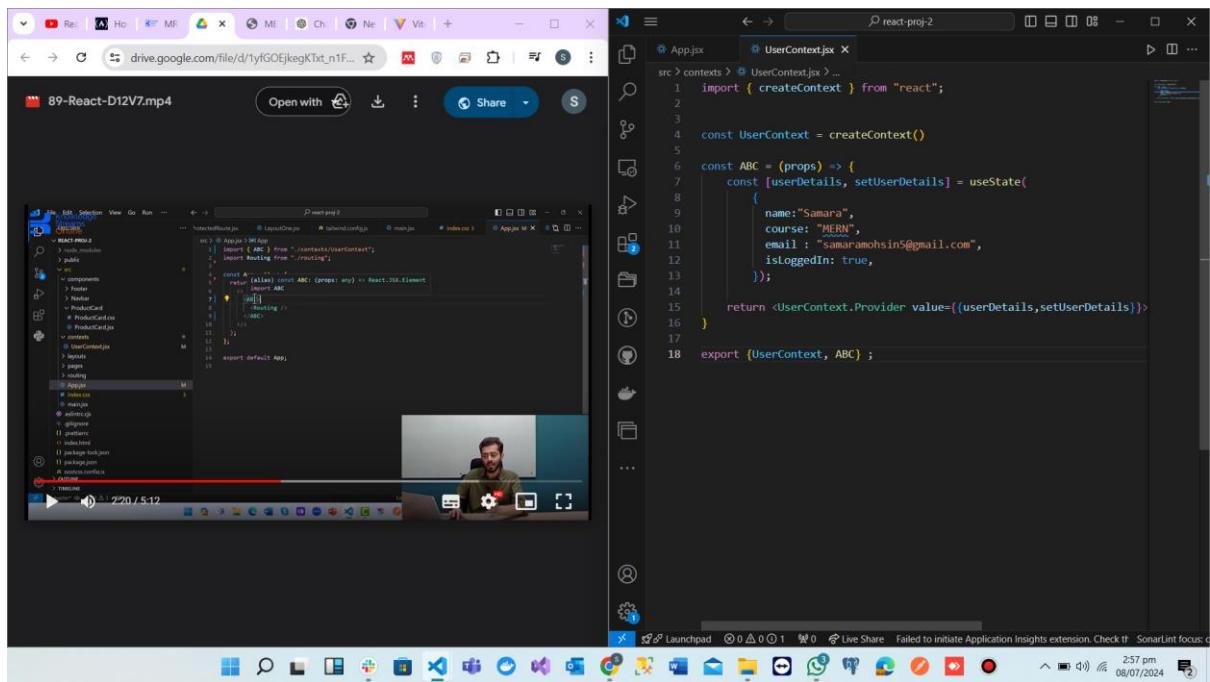
Agr hamare pas multiple context provider hain for eg user details seperate state me store krni hain cart details alg state me store krni hain tou kia kren?



The screenshot shows a video call interface with a developer's face on the right and their computer screen on the left. The computer screen displays a Windows desktop with a taskbar at the bottom. An open instance of VS Code is showing the file 'UserContext.jsx'. The code in the editor is:

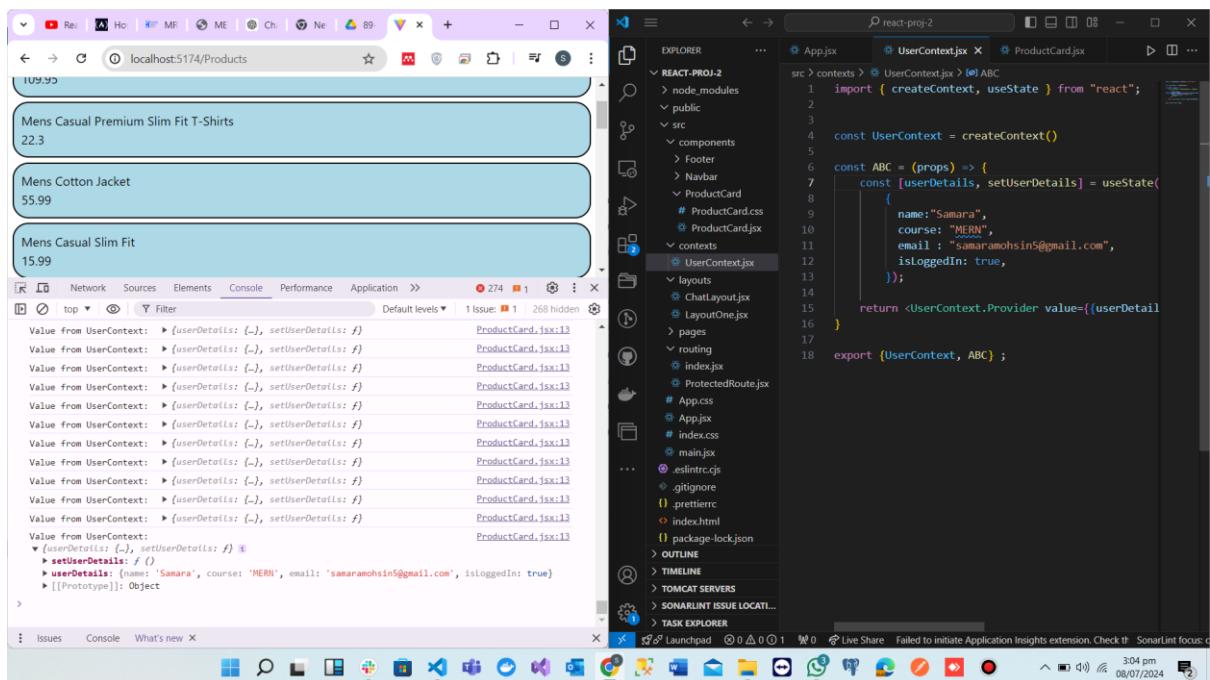
```
src > contexts > UserContext.jsx > ...
1 | import { createContext } from "react";
2 |
3 | const UserContext = createContext();
4 |
5 | const ABC = (props) => {
6 |   return <UserContext.Provider>{props.children}</UserContext.Provider>;
7 | }
8 |
9 | export { ABC, UserContext };
```

ABC component k jitne bhi children honge wo automatically wrap hojayenge usercontext k provider k ander, yani user context ki value ko ye saare children access krsakte hain



Routing ABC k lye child ha, aur ABC me ye kam hua wa ha k jo bhi child aayen unhe provider k ander wrap krdena aur provider ko value bhi ABC component me hi pass krdi ha taake hamari app.jsx file ka code clean rahe aur ziada clutter na ho

jitni bhi states req hain userContext k lye wo user Component ki file me hi aik component k ander bani wi ha aur us component ko use krke apni main application ko wrap krdenge.



Summary:

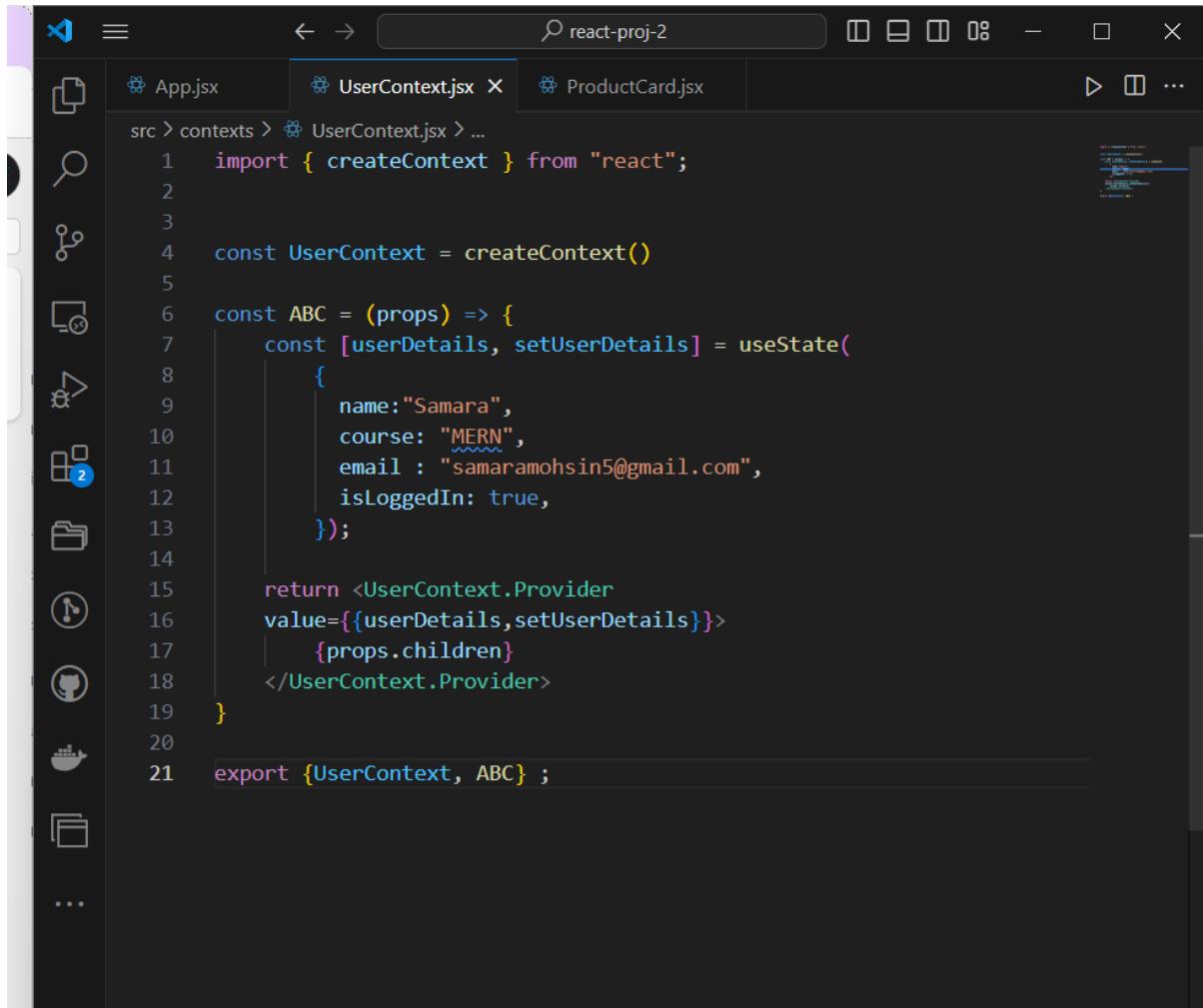
aik UserContext create kia

1 component banaya ABC userContext ki file me is component ka maqsad ye h iske jo bhi children hon wo sare children pakr k contextProvider me lagadena

&

Provider ko jo value bhejni ha wo bhi yahin define krди

Ab simple sa ye kam krna ha k jin components me in context provider ki values chahyen jaake unko ABC component me wrap krdo



```
src > contexts > UserContext.jsx > ...
1 import { createContext } from "react";
2
3
4 const UserContext = createContext()
5
6 const ABC = (props) => {
7   const [userDetails, setUserDetails] = useState(
8     {
9       name: "Samara",
10      course: "MERN",
11      email: "samarahmohsin5@gmail.com",
12      isLoggedIn: true,
13    });
14
15   return <UserContext.Provider
16     value={[userDetails, setUserDetails]}
17     {props.children}
18   </UserContext.Provider>
19 }
20
21 export {UserContext, ABC} ;
```

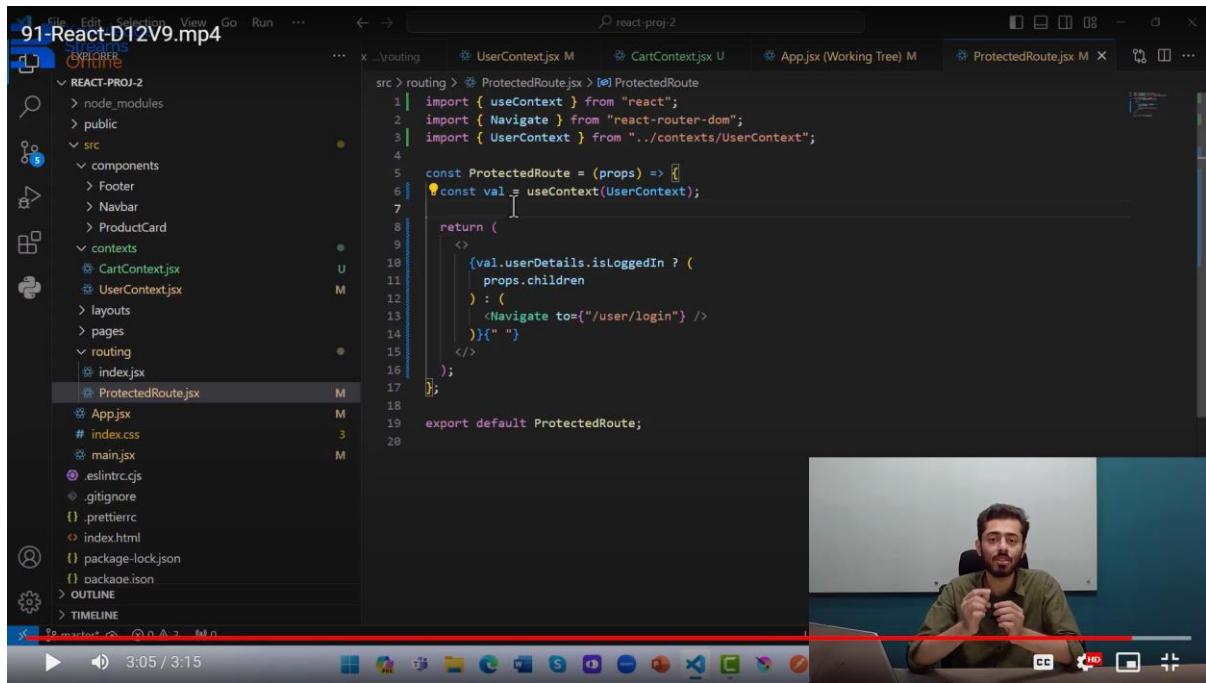
UseContext ko use krne ka method ye ha k useContext k ander context ka naam likhna ha

```
const { item } = props;
const val = useContext(UserContext);
useEffect(() => {
  console.log("Value from UserContext: ", val);
}, []);
```

```
src > App.jsx > App.jsx
1 import { ABC } from "./contexts/UserContext";
2 import Routing from "./routing";
3
4 const App = () => {
5   return (
6     <>
7       <ABC>
8         | <Routing />
9       </ABC>
10    </>
11  );
12};
13
14 export default App;
15
```

```
File Edit Selection View Go Run Terminal Help <- > react-proj-2
EXPLORER ... App.jsx UserContext.jsx CartContext.jsx main.jsx ProductCard.jsx
REACT-PROJ-2
src > contexts > UserContext.jsx
1 import { createContext } from "react";
2
3 const UserContext = createContext();
4
5 // UserContextProvider hamara apna banaya wa hi component ha uska name ham kuch bhi rakh sakte hain
6 // BrowserRouter bhi alk tarah se provider ka kam krta ha islium apni app ko browserRouter me wrap krte hain
7
8 const UserContextProvider = (props) => {
9   const [userDetails, setUserDetails] = useState(
10     {
11       name:"Samara",
12       course: "MERN",
13       email : "samaramohsin5@gmail.com",
14       isLoggedIn: true,
15     });
16
17   return <UserContext.Provider value={[userDetails, setUserDetails]}>
18     {props.children}
19   </UserContext.Provider>
20 }
21
22 export {UserContext, UserContextProvider} ;
```

```
File Edit Selection View Go Run Terminal Help <- > react-proj-2
EXPLORER ... App.jsx UserContext.jsx CartContext.jsx main.jsx ProductCard.jsx
REACT-PROJ-2
src > contexts > CartContext.jsx
1 import { createContext, useState } from "react";
2
3 const CartContext = createContext();
4
5 const CartContextProvider = (props) => {
6   const [cart, setCart] = useState([]);
7   return <CartContext.Provider value={[cart, setCart]}>{props.children}</CartContext.Provider>
8 }
9
10 export { [CartContext, CartContextProvider] };
```



Ab yehi useContextProvider hum login k pg pe use krenge wahan se set krenge loggedIn ko aur same time pe vo value is pg bhi update hojayegi this is the beauty of Context Provider(global sttae ka

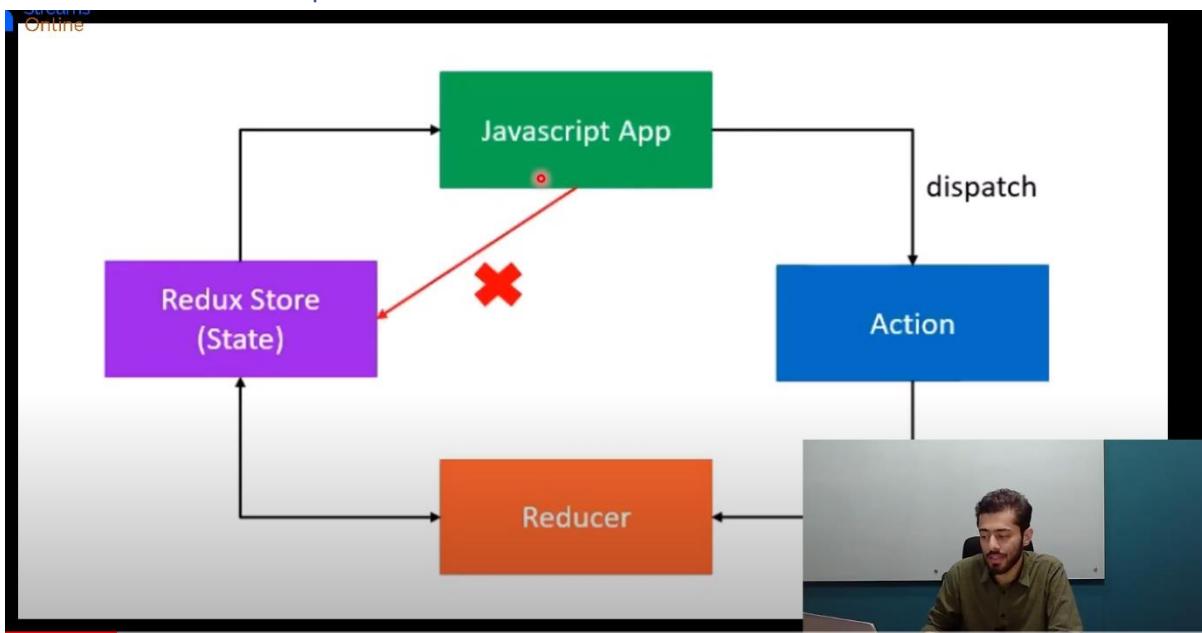
Redux

Globbal state ka kam context k through hojata h wo wo solution h jo React ka apna ha redux is not part of react k javascript ki library h

Difference b/w Redux and context:

- ❖ Context is solution provided by React while Redux is javascript libraray not react's lib.
- ❖ For large scle applications redux is preferable
- ❖ Redux ka initial code ziada hota h
- ❖ Redux works as middleware b/w
- ❖ Redux and context dono aik code m use hoskte hain possibility h

Three basic concepts of Redux



Redux Store/state : is the place where our all global states are kept

Aap kabhi bhi direct apni react app se is store ko change nhi krsakte

Apni javascript app se aik action ko dispatch krenge --> ye action aage jaake reducer chalayega --> reducer jaake state ko update krega aur phir ye updated state wapis aayegi aapki react app me

Action: aik javascript ka object h jiske ander type naam ki property hoti h

Reducer : aik function h jiske ander multiple if else ki conditions hoti hain

reducers are pure reducers means agr reducer ko X input den aur y output aaye, agr dubara x input dene pe y hi output aayega nge nhi hosakta k reducer ka behavior input ko waja se change hojaye

e.g store k ander no.of shirts 10 hain aur buy shirt ka aik func chalaya tou 10-1 no.of shirts 9 rehga in store me agr dubara yehi func chlayenge tou 8-1 7 shirts rehjayenig

The state of entire application is maintained in a single store

1



The only way to change state is to emit/dispatch an action

2



Reducers will return new objects with updated state values

3



Packages to install for Redux

```
sktop Data shifted here\React Project\react-proj-3> npm i redux react-redux
```

Provider

Context me aik hota ha provider aik hota ah consumer , consumer ka akm ab khatam hogaya ha wo hum useContext k through value read krlete hain
provider ka basically kam hota ye h k provider k ander jo bhi children wrap honge unhe access miljayega un cheezon ka jo provider provide kr raha h, one of e.g is BrowserRouter

React redux jo library ha wo bhi internaly aik provider ka use krti h redux ki functionality ko implement krne k lye

UseSelector

Ye hook milta ha react redux se

UseSelector hook redux store me se value utha rahi h aur application ko de rahi h

Hen useselector ki hook uthake value de rahi ha application ko Sabse pehle reducer ka reducer chalta ha jo initial state ko rakhwa raha ha store me

The screenshot shows a video call interface with a person speaking on the right and a code editor and browser window on the left.

VS Code Editor:

- File Structure:** REACT-PROJ-3 / src / redux / cakes / App.jsx, cakeActions.js, cakeReducer.js, store.js
- App.jsx:**

```

1 import { useSelector } from "react-redux";
2
3 const App = () => {
4   const reduxState = useSelector((state) => {
5     return state;
6   });
7
8   console.log("REDUX STORE STATE: ", reduxState);
9
10  return [
11    <div>
12      <h1>Num of Cakes</h1>
13      <button>Buy Cake</button>
14    </div>
15  ];
16}
17
18 export default App;
19

```
- cakeReducer.js:**

```

1 const initialState = {
2   numOfCakes: 10,
3 };
4
5
6 const cakeReducer = (state = initialState, action) => {
7   if (action.type === "BUY_CAKE") {
8     return {
9       numOfCakes: state.numOfCakes - 1,
10     };
11   }
12
13   export default cakeReducer;
14

```

Browser DevTools Console:

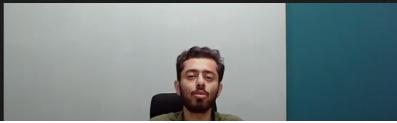
- Logs from the browser show the initial state being undefined.
- Logs from the browser show the state changing to 9 after a click event.
- Logs from the browser show the state changing to 8 after another click event.

Agr application pehli dafa run hui h tou store tou exist hi nhi krta tou initial value state ye Reducer se pick krle.

Applicatio abhi shuroo hui ha tou user ne abhi koi action dispatch hi nhi kia hogha tou reducer aik aisa function h jo khudi chalta ha

Tou ab hum usey ye bata rahe hain k agr tumhe ye store se na mile tou initial state ko use krle na ab agr koi action aaya wa ha tou if cond chaladena , agr action aaya hi nhi h aur reducer sabse pehli dafa me chal rha ha tou reducer kia return krega jo store me save hoga? Hame ye pata ha k jo kuch reducer return krega wo jake store me save hojayega tou iske lye else me cond dena zaroori h jo hamara boundary case bhi hoga.

K agr maine k agr reducer kisi aise case k lye chal gaya jiska action maine create nhi kia tou wahan bhi state return krdo



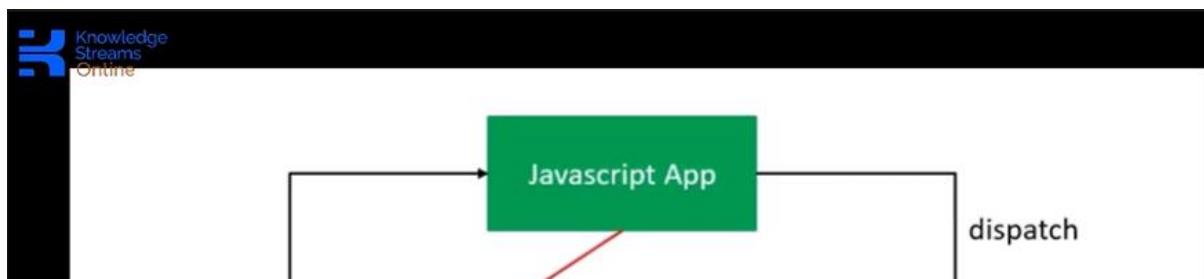
A screenshot of a code editor window titled "react-proj-3". The left sidebar shows a file tree with files like App.jsx, cakeActions.js, cakeReducer.js, store.js, and main.jsx. The right pane displays the content of cakeReducer.js:

```
src > redux > cakes > cakeReducer.js > [e] cakeReducer
1 const initialState = {
2   numOfCakes: 10,
3 };
4
5 const cakeReducer = (state = initialState, action) => {
6   if (action.type === "BUY_CAKE") {
7     return {
8       numOfCakes: state.numOfCakes - 1,
9     };
10  } else {
11    return state;
12  }
13}
14
15 export default cakeReducer;
16
```



A screenshot of a code editor window titled "react-proj-3". The left sidebar shows a file tree with files like App.jsx, cakeActions.js, cakeReducer.js, store.js, and main.jsx. The right pane displays the content of App.jsx:

```
src > [e] App.jsx > [e] App
1 import { useSelector } from "react-redux";
2
3 const App = () => {
4   const numOfCakes = useSelector((state) => {
5     return state.numOfCakes;
6   });
7
8   console.log("NUMBER OF CAKES FROM REDUX-", numOfCakes);
9
10  return (
11    <div>
12      <h1>Number of Cakes: <span>{numOfCakes}</span></h1>
13      <button>Buy Cake</button>
14    </div>
15  );
16
17
18 export default App;
19
```



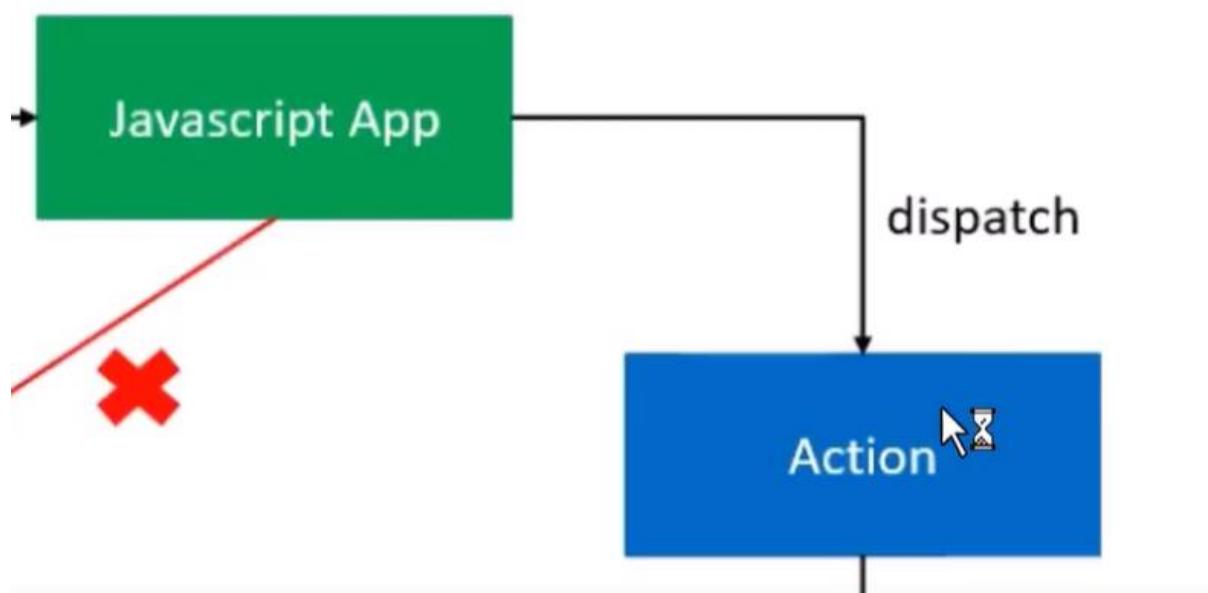
ss

s

Step 1: reducer chalega aur wo as it is state ko return krdega() reducer state ko update kr raha ha

Step 2: And redux store & app k between jo connection h ye establish hota ha useSelector ki hook se

Next connection is:



Javascript app se action ka connection useDispatch k hook se hota ha

Action and reducer ka connection automatic h

Reducer aur store ka connection createStore ke func pe bantaha

Flow of Redux

1 action dispatch krti hun ma button pe click krke
jab aik action dispatch hota ha tou reducer chalta ha automatically
jo bhi action dispatch hota ha uski behalf pe reducer pe wo wala case chalega

Aap reducer se jo kuch bhi return krwayenge wo as it is jaake hamare store k ander save hojayega
previous saari values over write hojayengin therefore keep in mind property ki spelling her jaga same
ho jo bhi naam h property ka. Ab jb reducer ne store update krdia

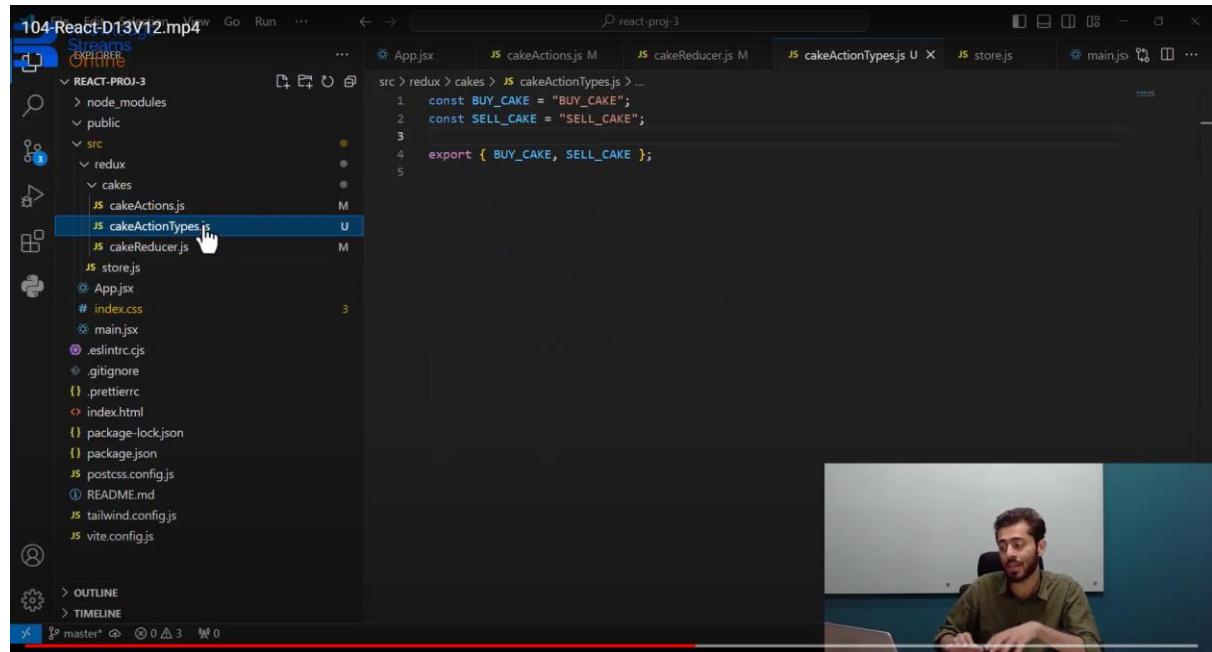
App.jsx me useSelector se hamne vo value read ki wi ha tou hamare frontend pe bhi wo value update
hojayegi,

(useDispatch)Action dispatch --> reducer run --> update the state/ redux store(overwritten prev.) ---
(useSelector)--> frontend update

When action does not dispatch:

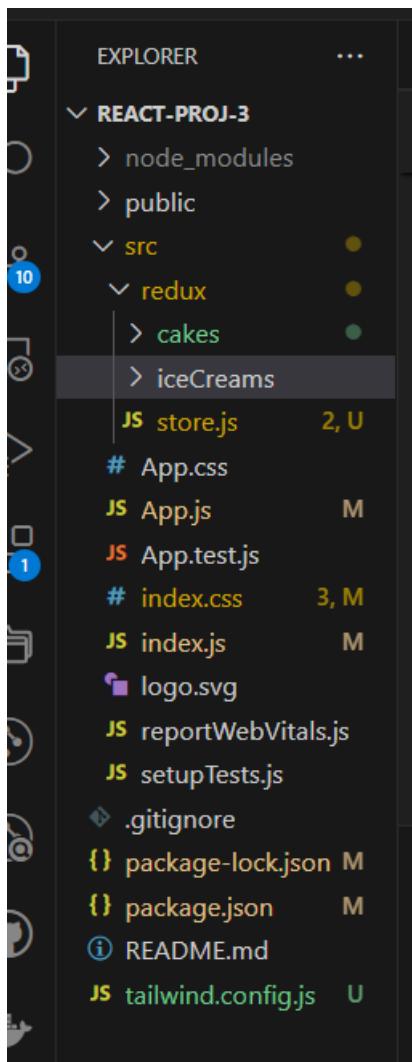
Reducer automatically run --> initial value set in state --> frontend

ActionTypes



```
const BUY_COKE = "BUY_COKE";
const SELL_COKE = "SELL_COKE";
export { BUY_COKE, SELL_COKE };
```

Multiple Reducers



Store 1 hi ha global for multiple reducers

NODE JS:

APIS

Backend pe ham aise functions define krte hain jo frontend se request accept krte hain and response return krte hain frontend ko, they are called APIS

Express JS

Express is a full stack framework jisme backend k sath sath frontend bhi create krsakte hain but we will use it for backend only

Express JS me jo apps cretae hui ei hoti hain unhe run krne k liye jo run time required hota ha wo NodeJS provide karta ha

So NodeJS should be installed



Before proceeding, ensure that you have following tools installed on your System:

1. Node JS

(Installation link: <https://nodejs.org/en/download/>)

2. Postman

(Installation link: <https://www.postman.com/downloads/>)

Postman is API tester

Create Express Application



Creating an Express application

- Create a Folder and Open in VS Code.
- Write the below commands in terminal.

```
npm install express-generator
```

To execute application:

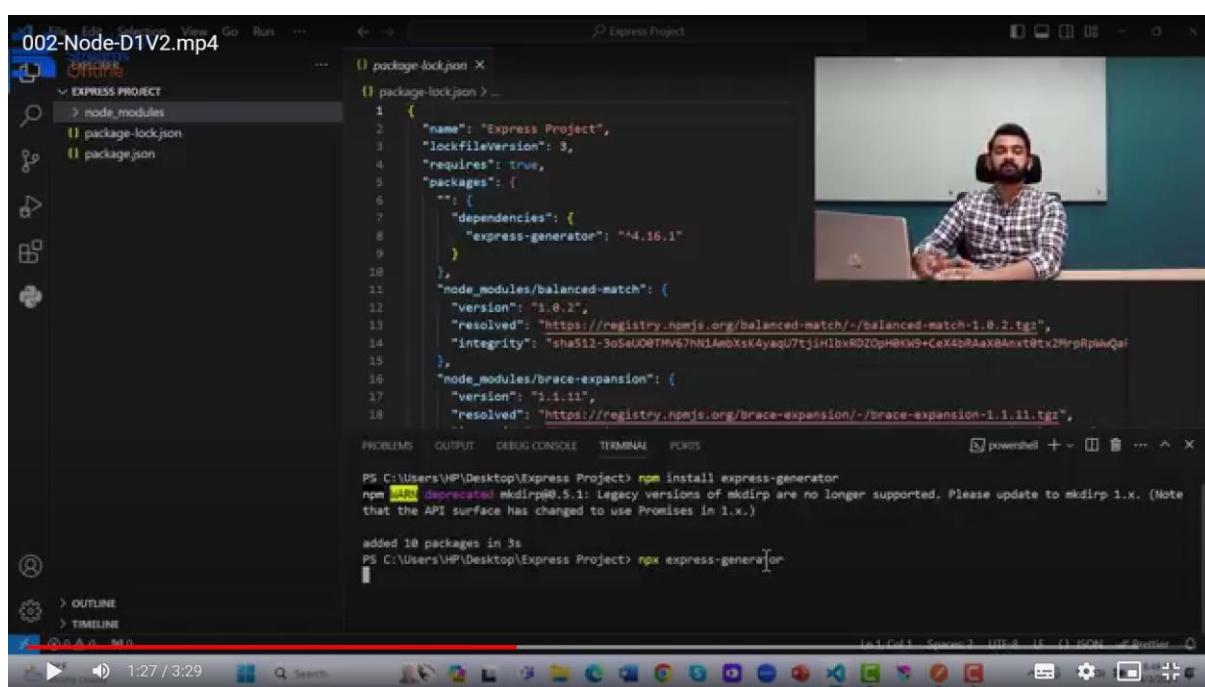


Creating an Express application

- Create a Folder and Open in VS Code.
- Write the below commands in terminal.



Delete unnecessary file related to frontend



```

PS E:\practise and udemy courses\WERN course\express js\express project> npm install express-generator
npm WARN deprecated mkdirp@0.5.1: Legacy versions of mkdirp are no longer supported. Please update to mkdirp 1.x. (Note that the API surface has changed to use Promises in 1.x.)
added 10 packages in 7s
PS E:\practise and udemy courses\WERN course\express js\express project> npx express-generator
warning: the default view engine will not be jade in future releases
warning: use `--view=jade` or `--help` for additional options
destination is not empty, continue? [y/N] y
create : public\
create : public\javascripts
create : public\images
create : public\styleheets
create : public\styleheets\style.css
create : routes
create : routes\index.js
create : routes\users.js
create : views
create : views\error.jade
create : views\index.jade
create : views\layout.jade
create : app.js
create : package.json
create : bin
create : bin\www

install dependencies:
> npm install

run the app:
> SET DEBUG=express-project:* & npm start

```

comment out these lines from App.js after deleting folders then run `npm install`

```

JS app.js 9 X
JS app.js > ...
1 var createError = require('http-errors');
2 var express = require('express');
3 // var path = require('path');
4 var cookieParser = require('cookie-parser');
5 var logger = require('morgan');
6
7 var indexRouter = require('./routes/index');
8 var usersRouter = require('./routes/users');
9
10 var app = express();
11
12 // view engine setup
13 // app.set('views', path.join(__dirname, 'views'));
14 // app.set('view engine', 'jade');
15
16 app.use(logger('dev'));
17 app.use(express.json());
18 app.use(express.urlencoded({ extended: false }));
19 app.use(cookieParser());
20 // app.use(express.static(path.join(__dirname, 'public')));
21

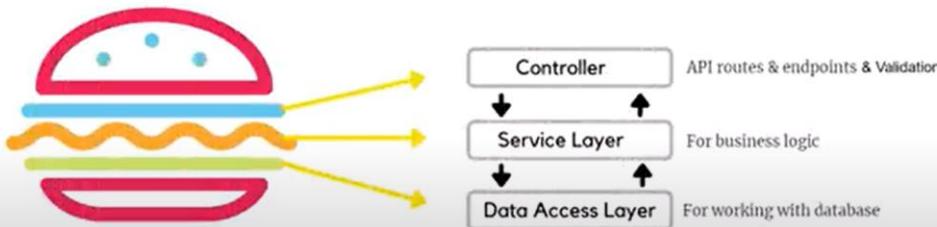
```

Folder Structure

Controller Service Model

Controller k ander hum router se request accept krte hain aur us router se aane wali request me agr koi data exist krta ha tou usko validate krte hain usko validate krne k baad hum 2nd layer service k ander bhej dete hain jahan pe koi bhi business logic implement ki jati ha then hum usko send krde hain data layer pe jahan pe database operation perform hota ha, database operation hone k baad wo service ko wapis return hota ha aur check krta ha k database se jo data receive hua ha wo validate ha ya nhi, wo authenticate ha ya nhi aur wo proper format me h ya nhi uske baad hum is request ko wapis controller ko return krde hain aur controller isko wapis frontend ko return krde hain

Layered Approach (Controller Services Model)

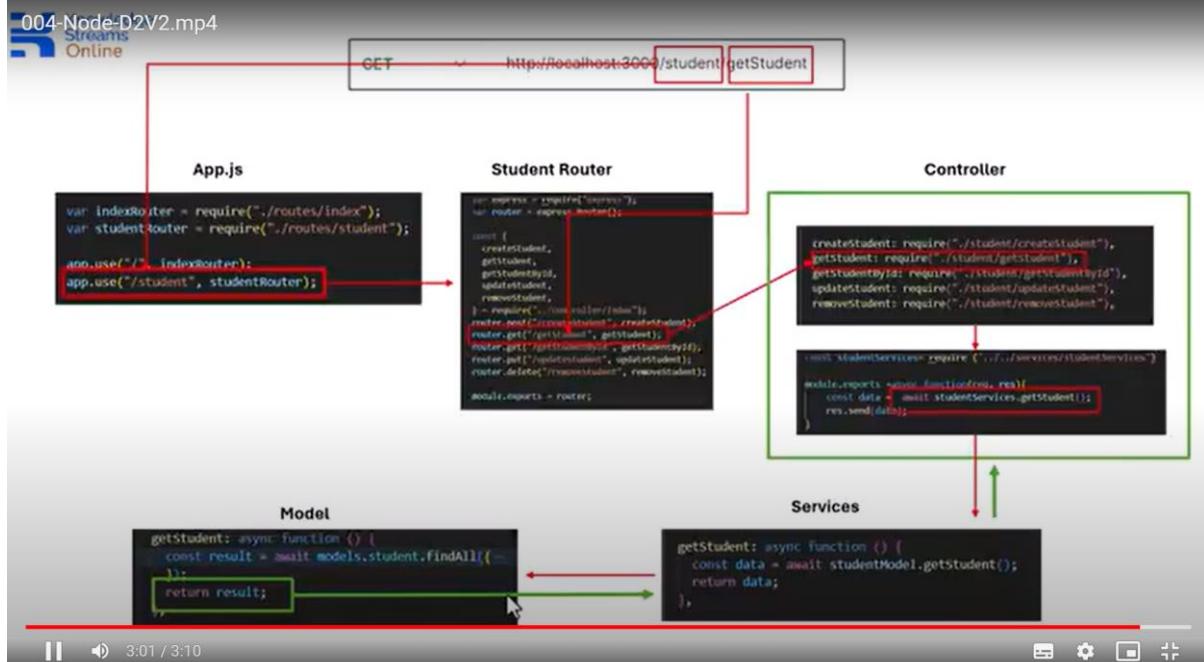
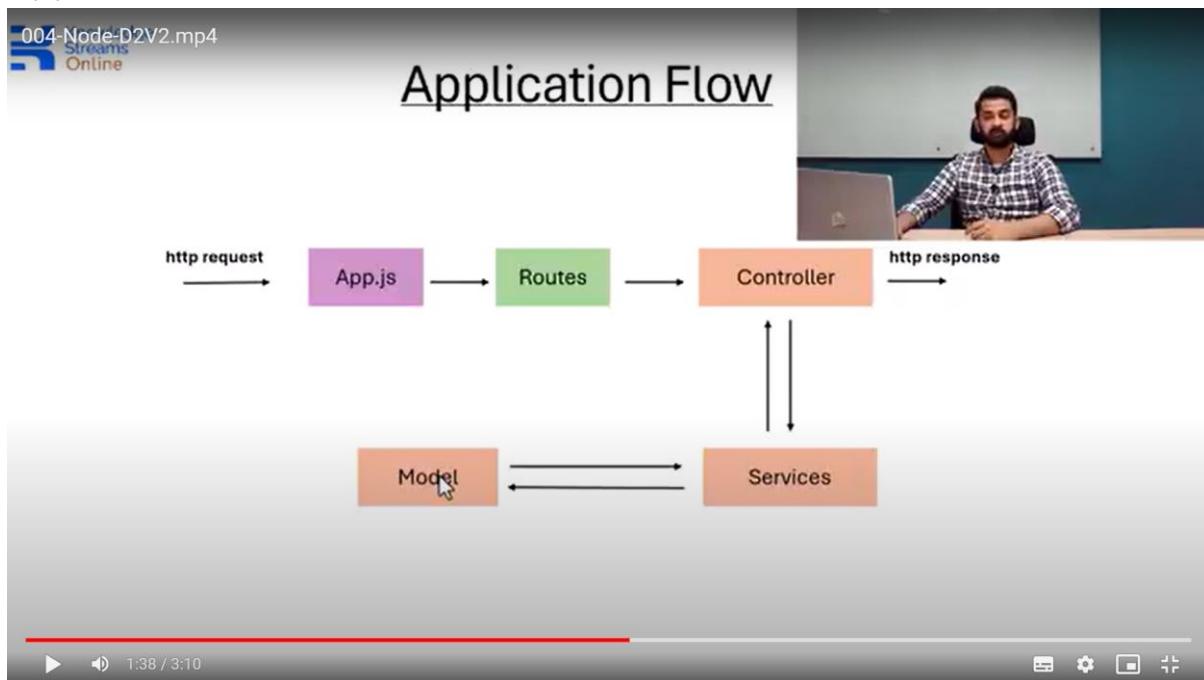


Settings

▶ 0:59 / 3:10

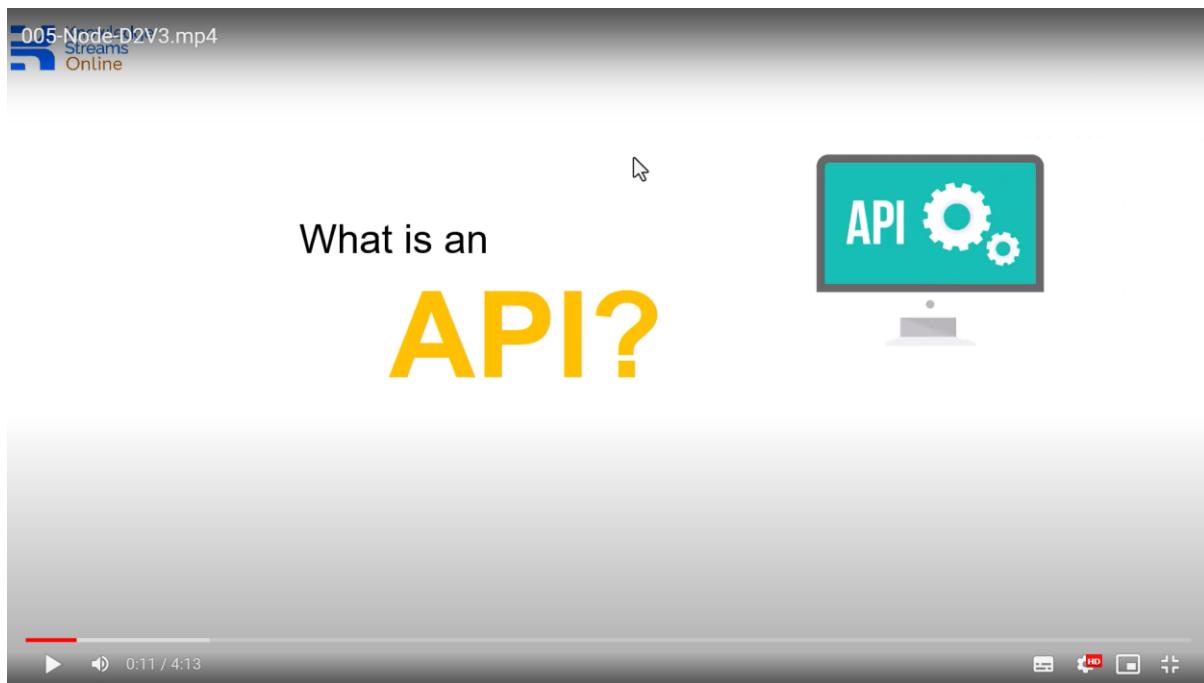
■ ⚙ □ □

Application Flow:



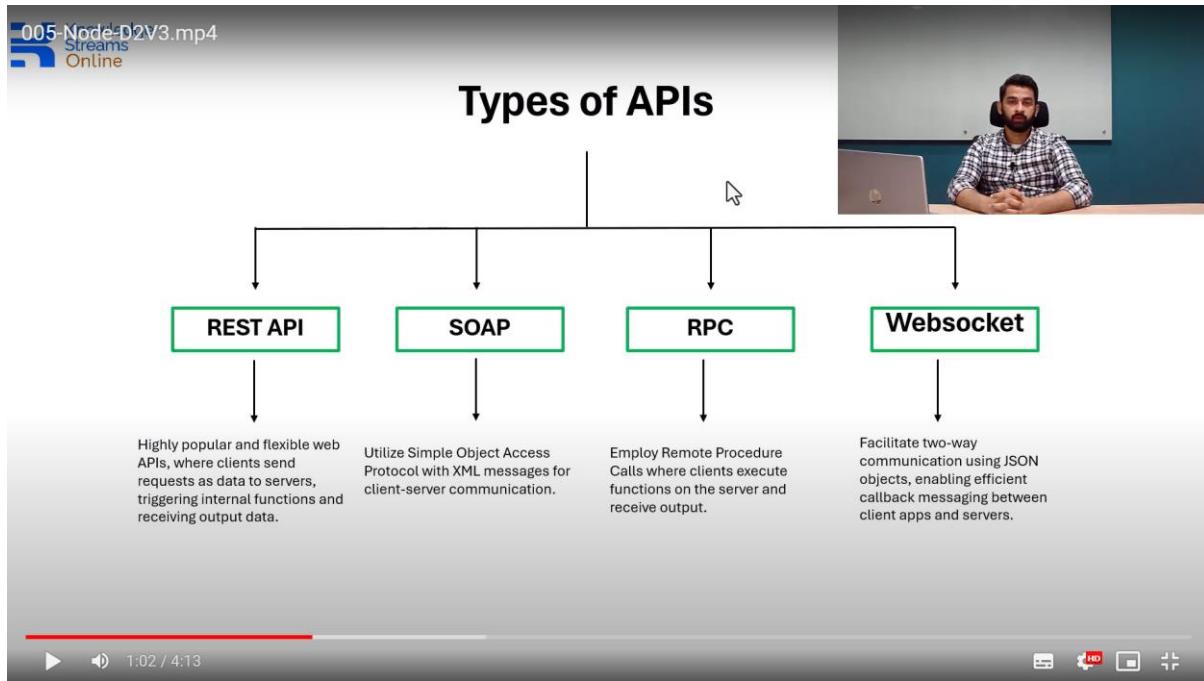
localhost which is host deployed locally then port then base path, base path k sath relevant router likha wa ha jahan se wo relevant router call hogा relevant router pe wo exact path ha jo hum hum /basepath/exactpath likhte hain /student/getstudent is the end point for e.g so ye students ka data return krega uske baad , then ye student k controller function pe jata ha aur usko validate krne k baad ye uski service ko jata ha then ye service k jane k baad dekhta ha k hame uska kosa relevant model call krna ha yani k relevant database table call krna ha, table call krne k baad hum uska sara ka sara data extract krenge aur extract krne k baad service ko bhejenge service se wo wapis controller pe return hogा aur controller se wo return hojayega httpResponse wapis service me

API

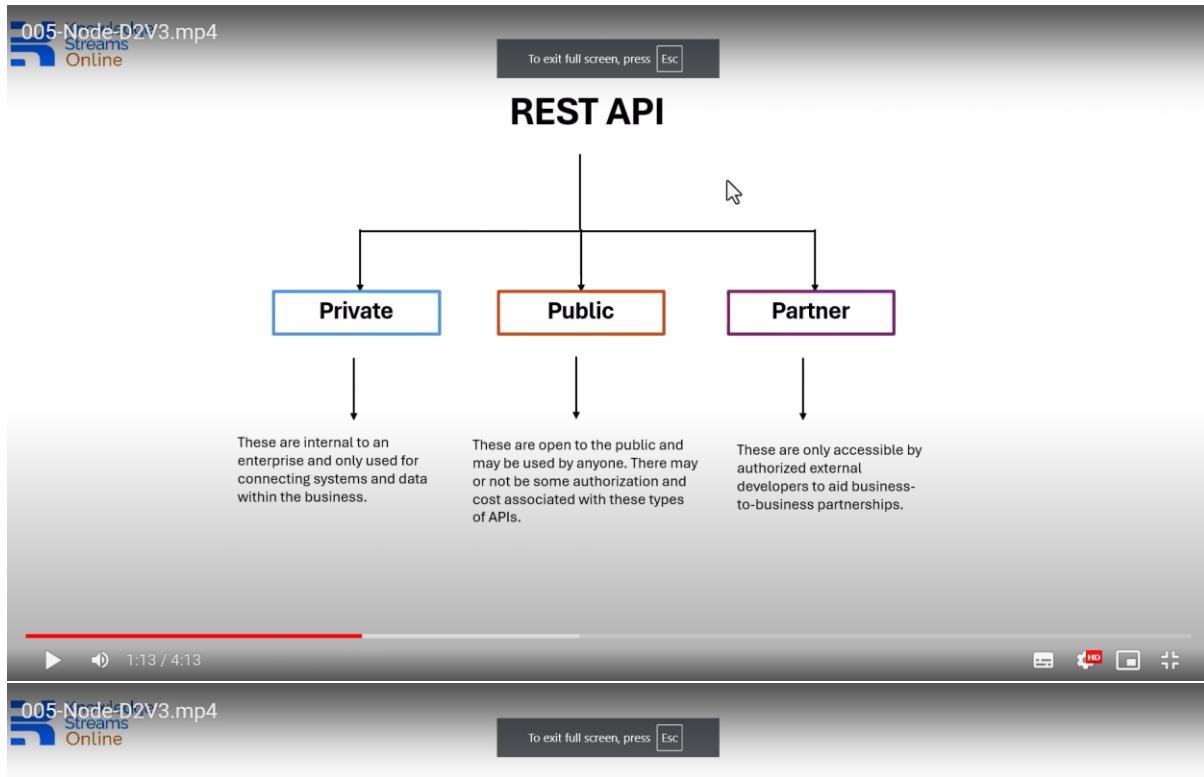


Ye aise function hote hain jo frontend se request accept krte hain aur us request ko execute krne ka baad response return krte hain

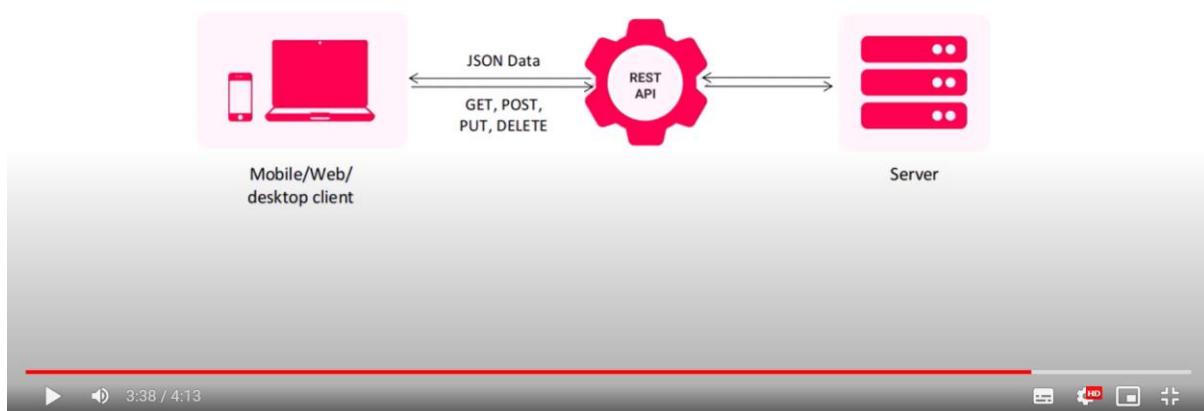
Types of APIs:



Rest API Categories



REST API Model



API Methods:

get post push delete

Nodemon

The screenshot shows the VS Code interface with an 'EXPRESS PROJECT' workspace. The 'indexController.js' file is open in the editor, containing the following code:

```
module.exports = {
  getTasks : (req,res) => {
    try {
      return res.send("message")
    }
    catch (e) {
      return res.send(error.message)
    }
  }
}
```

The 'TERMINAL' tab shows the command line output of the application running with nodemon:

```
PS E:\practise and udemy courses\WERN Course\express js\express project> npm start
> express-project@0.0.0 start
> node ./bin/www
^C[Terminal job] (Y/N)? y
PS E:\practise and udemy courses\WERN Course\express js\express project> npm start
> express-project@0.0.0 start
> node ./bin/www
GET / 200 1485.153 ms - 170
GET /stylesheets/style.css 200 3.717 ms - 111
^C[Terminal job] (Y/N)? y
PS E:\practise and udemy courses\WERN Course\express js\express project> npm i nodemon
```

The status bar at the bottom indicates the time as 12:35 pm and the date as 30/07/2024.

Jab bhi hum backend k code me kuch bhi change krenge tou ye package automatically app ko refresh kr deta ha

First API

Folder in root directory: controller

IndexController.js

The screenshot shows the VS Code interface with an 'EXPRESS PROJECT' workspace. The 'indexController.js' file is open in the editor, containing the same code as before:

```
module.exports = {
  getTasks : (req,res) => {
    try {
      return res.send("message")
    }
    catch (e) {
      return res.send(error.message)
    }
  }
}
```

The 'TERMINAL' tab shows the application starting with nodemon and then restarting due to changes in the code:

```
Node.js v20.3.0
[nodemon] app crashed - waiting for file changes before starting...
[nodemon] restarting due to changes...
[nodemon] starting 'node ./bin/www'
[nodemon] restarting due to changes...
[nodemon] starting 'node ./bin/www'
[nodemon] restarting due to changes...
[nodemon] starting 'node ./bin/www'
[nodemon] restarting due to changes...
[nodemon] starting 'node ./bin/www'
[nodemon] restarting due to changes...
[nodemon] starting 'node ./bin/www'
[nodemon] restarting due to changes...
[nodemon] starting 'node ./bin/www'
[nodemon] restarting due to changes...
[nodemon] starting 'node ./bin/www'
[nodemon] restarting due to changes...
[nodemon] starting 'node ./bin/www'
[nodemon] restarting due to changes...
[nodemon] starting 'node ./bin/www'
```

The status bar at the bottom indicates the time as 1:29 pm and the date as 30/07/2024.

Add path in index.js:

File Edit Selection View Go Run Terminal Help ↻ → express project

EXPLORER
EXPRESS PROJECT
bin
www
controller
indexController.js
node_modules
public
routes
index.js
users.js
views
error.jade
index.jade
layout.jade
app.js
package-lock.json
package.json

routes > index.js > ...
1 var express = require('express');
2 var router = express.Router();
3
4 var {getTasks} = require('../controller/indexController');
5 // var indexController = require('../controller/indexController');
6
7 /* GET home page. */
8 // router.get('/getTasks',indexController.getTasks);
9 router.get('/getTasks',getTasks);
10
11
12 module.exports = router;
13

PROBLEMS DEBUG CONSOLE TERMINAL PORTS AZURE

Node.js v20.3.0
[nodemon] app crashed - waiting for file changes before starting...
[nodemon] restarting due to changes...
[nodemon] starting node ./bin/node...
[nodemon] restarting due to changes...
[nodemon] starting node ./bin/node...
[nodemon] restarting due to changes...
[nodemon] starting node ./bin/node...
[nodemon] restarting due to changes...
[nodemon] starting node ./bin/node...
[nodemon] restarting due to changes...
[nodemon] starting node ./bin/node...
[nodemon] restarting due to changes...
[nodemon] starting node ./bin/node...
[nodemon] restarting due to changes...
[nodemon] starting node ./bin/node...
[nodemon] restarting due to changes...
[nodemon] starting node ./bin/node...
GET /getTasks 200 4.373 ms - 7

Launchpad Live Share Failed to initiate Application Insights extension. Check the console for more details or reload the extension to try again. SonarLint focus: overall code Watch LF {} JavaScript Go Live Spell Prettier

App.js remains default:

The screenshot shows a Visual Studio Code interface with the following details:

- File Explorer:** On the left, it shows the project structure under "EXPRESS PROJECT". The "app.js" file is currently selected.
- Code Editor:** The main area displays the "app.js" file content, which sets up an Express application with routes for index and users, and includes middleware like logger and error handling.
- Bottom Status Bar:** Shows the status "node +", terminal tabs, and other system information.

The image shows a dual-monitor setup. The left monitor displays the Microsoft Visual Studio Code interface, specifically an Express project workspace. The right monitor displays the Postman application.

Left Monitor (VS Code):

- EXPLORER:** Shows the file structure of an "EXPRESS PROJECT". Files listed include indexController.js, index.js, app.js, pack.json, bin/www, controller/indexController.js, routes/index.js, users.js, views/error.jade, views/index.jade, views/layout.jade, app.js, package-lock.json, and package.json.
- CODE EDITOR:** The indexController.js file is open, showing the following code:

```
1 module.exports = {
2   getTasks : (req,res) => {
3     try {
4       return res.send("message: My first API")
5     } catch (error) {
6       return res.send(error.message)
7     }
8   }
9 }
10 }
```

Bottom Status Bar: Shows "SONARLINT ISSUE LOCATI..." and "TASK EXPLORER".

Right Monitor (Postman):

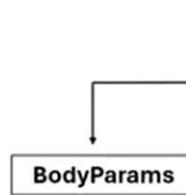
- Header:** Home, Workspaces, API Network, Search Postman, Invite, Upgrade, Save, Share, No environment.
- Left Sidebar:** My Workspace (Collections, Environments, History), New, Import.
- Request Panel:** GET localhost:3000/getTasks, Params, Authorization, Headers (7), Body, Scripts, Tests, Settings, Cookies.
- Body:** Status: 200 OK, Time: 17 ms, Size: 233 B, Save as example.
- Content:** message
- Bottom Status Bar:** Postbot, Runner, Start Proxy, Cookies, Vault, Trash, 1:31 pm, 30/07/2024.

The bottom of the image shows the Windows taskbar with various pinned icons and the system clock.

Body Parameters ; POST / PUT



Request and Response Objects



The screenshot shows the Postman interface for a POST request to `http://localhost:3000/user/createUser`. The 'Body' tab is selected, showing a raw JSON payload:

```
1 "id": 1,
2 "name": "abc",
3 "email": "abc@gmail.com"
```

Other tabs like 'Headers' and 'JSON' are also visible.

AIK JSON Obj create krlia ab ye request k sath as payload backend ko send hojayega.

Put and Post dono cases me wo request k sath bind hoke backend ko receive hojayega , hum request k obj me se usko get krsakte hain by using **request.body**

Query Parameters ; GET / DELETE

Query parameters me ham elimited data send krna hota kabhi hame saara data get krna hota ha kabhi koi specific data

aur delete k case me ya tou saare records delete krne hote hain jike lye koi data send krne ki zaroorat nhi ya specifically koi record delete krna hota ha. Ya tou id send krenge ya array of ids to be deleted not all

Request and Response Objects



BodyParams

queryParams

The screenshot displays two Postman requests. The left request is a POST to `http://localhost:3000/user/createUser`. In the 'Body' tab, the 'raw' option is selected, and the JSON content is:

```
1 "id": 1,
2 "name": "abc",
3 "email": "abc@gmail.com"
```

The right request is a GET to `http://localhost:3000/user/getUserById?id=2`. In the 'queryParams' tab, there is one entry: 'id' with a value of '2'.

both query parameters and body parameters are json objects i.e key value pairs aur hum in request.body ya request.query krke get krsakte hain

Query params:

Postman me jake query parameter add kia wo automatically url me add hogaye

Phir usko send krke dekha tou terminal pe poora url show horaha ha along with parameters

GET Request with QUERY PARAMETERS

The screenshot displays two instances of the Visual Studio Code IDE. Both instances show an Express.js project structure with files like indexController.js, index.js, app.js, and package.json.

Top Instance (Screenshot 1):

- Terminal:** Shows the command `node .` being run, followed by the output of the Node.js application which includes logs for the task ID and name.
- Output Panel:** Shows a successful GET request to `localhost:3000/getTasks?taskid=123456&taskname=Home%20Work` with a response body of "message: My first API".

Bottom Instance (Screenshot 2):

- Terminal:** Shows the code in indexController.js where the `getTasks` function logs the `req.query` object to the console.
- Output Panel:** Shows the same successful GET request and its response body.

The screenshot shows the VS Code interface with the following details:

- Explorer View:** Shows the project structure with files like indexController.js, index.js, app.js, and package.json.
- Code Editor:** Displays the indexController.js file containing code for handling GET and POST requests.
- Terminal:** Shows the output of a command, likely related to the application's configuration or logs.
- Postman:** A browser-based API testing tool showing a GET request to `localhost:3000/getTasks?taskId=123456&taskname=Home%20Work`. The response body is `{"taskid": "123456", "taskname": "Home Work"}`.
- Status Bar:** Shows the status bar with various icons and information, including "Live Share" and "Failed to initiate Application Insights extension".

Hamare pas jitni requests hongi unke utne hi controller banengy

Res.send hame single line return krta ha multiple lines agr send krni hain tou object banate hain

Jaise her request k lye alg controller hogा isi tarah her request k lye different router func bhi hogा

POST Request

Contoller for post req

The screenshot shows the VS Code interface with the following details:

- Explorer View:** Shows the project structure with files like indexController.js, index.js, app.js, and package.json.
- Code Editor:** Displays the indexController.js file with both GET and POST routes defined.
- Terminal:** Shows the output of a command, likely related to the application's configuration or logs.
- Status Bar:** Shows the status bar with various icons and information, including "Live Share" and "Failed to initiate Application Insights extension".

Route for post request:

```

indexController.js
index.js
app.js
package.json

routes > JS index.js > ...
1 var express = require('express');
2 var router = express.Router();
3
4 var {getTasks, createTask} = require("../controller/indexController");
5 // var indexController = require("./controller/indexController");
6
7 /* GET home page. */
8 // router.get('/getTasks',indexController.getTasks);
9 router.get('/getTasks',getTasks);
10 router.post('/createTask',createTask);
11
12
13 module.exports = router;
14

```

PROBLEMS DEBUG CONSOLE TERMINAL PORTS AZURE

Create an Azure resource

View all of your recent activities and quickly access resources you've recently created.

OUTLINE TIMELINE TOMCAT SERVERS SONARLINT ISSUE LOCATOR TASK EXPLORER

Launched 0 Live Share Failed to initiate Application Insights extension. Check the console for more details or reload the extension to try again. SonarLint focus: overall code Watch LF (JavaScript Go Live Spell Prettier

Home Workspaces API Network

New Import Overview index GET New Request POST createTask + No environment

to do list / index / createTask

POST localhost:3000/createTask

Params Authorization Headers (8) Body Scripts Tests Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```

1 {
2   "message": "my post request",
3   "data": "data"
4 }

```

200 OK 26 ms 278 B Save as example

FIND Enter text to find... Find

Regex Ignore Case

Postbot Runner Start Proxy Cookies Vault Trash

POST method API using body-params

```

controller > js indexController.js > (o) <unknown> > ⚡ getTasks
1 module.exports = {
2   "getTasks : (req,res) => {
3     console.log("request data: ", req.query);
4     try {
5       return res.send(req.query);
6     }
7     catch (error) {
8       return res.send(error.message)
9     }
10  },
11  "createTask : (req,res) => {
12    try {
13      return res.send({
14        message : "my post request",
15        // data : "data"
16        data : req.body
17      })
18    } catch (error) {
19      return res.send(error.message)
20    }
21  }
22 }

at Object.

PROBLEMS DEBUG CONSOLE TERMINAL PORTS AZURE



at Object.at Module._compile (node:internal/modules/cjs/loader:1255:14)



Node.js v20.3.0  
[nodemon] app crashed - waiting for file changes before starting...  
[nodemon] restarting due to changes...  
[nodemon] starting `node ./bin/www`  
POST /createTask 200 12.325 ms - 82



Launchpad Live Share Failed to initiate Application Insights extension. Check the console for more details or reload the extension to try again. SonarLint focus: overall code Watch JavaScript Go Live Spell Prettier



My Workspace New Import Overview GET New Request POST createTask



HTTP to do list / index / createTask



POST localhost:3000/createTask



Params Authorization Headers (9) Body Scripts Tests Settings Cookies



none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify



```

1 {
2 ...
3 "taskName": "home_task",
4 ...
5 "desc": "description"
6 }

```



Body Cookies Headers (7) Test Results 200 OK 21 ms 317 B Save as example



Pretty Raw Preview Visualize JSON



```

1 {
2 ...
3 "message": "my post request",
4 "data": [
5 {
6 "taskName": "home task",
7 "desc": "description"
8 }
9]
10 }

```



Postbot Runner Start Proxy Cookies Vault Trash


```

PUT Request:

You must have a unique parameter of record jisse aap us particular record ko target krsaken aur update krsaken

```
File Edit Selection View Go Run Terminal Help express project
EXPLORE indexController.js index.js app.js package.json
controller
indexController.js
1 module.exports = {
2   getTasks : (req,res) => {
3     try {
4       return res.send({
5         message : "my post request",
6         // data : "data"
7         data : req.body
8       })
9     } catch (error) {
10       return res.send(error.message)
11     }
12   },
13   createTask : (req,res) => {
14     try {
15       return res.send({
16         message : "my post request",
17         // data : "data"
18         data : req.body
19       })
20     } catch (error) {
21       return res.send(error.message)
22     }
23   },
24   updateTask: (req,res) => {
25     try {
26       return res.send({
27         message:"this request is to update task",
28         data : req.body
29       })
30     } catch (error) {
31       return res.send(error.message);
32     }
33   }
}

```

OUTLINE
TIMELINE
TOMCAT SERVERS
SONARLINT ISSUE LOCATIONS
TASK EXPLORER

PROBLEMS DEBUG CONSOLE TERMINAL PORTS AZURE

[node:0] starting 'node ./bin/www'
PUT /updateTask 200 33.401 ms - 93

```
File Edit Selection View Go Run Terminal Help express project
EXPLORE indexController.js index.js app.js package.json
controller
index.js
1 var express = require('express');
2 var router = express.Router();
3
4 var {getTasks, createTask, updateTask} = require('../controller/indexController');
5 // var indexController = require("../controller/indexController");
6
7 /* GET home page. */
8 // router.get('/getTasks',indexController.getTasks);
9 router.get('/getTasks',getTasks);
10 router.post('/createTask',createTask)
11 router.put('/updateTask',updateTask)
12
13
14 module.exports = router;

```

OUTLINE
TIMELINE
TOMCAT SERVERS
SONARLINT ISSUE LOCATIONS
TASK EXPLORER

PROBLEMS DEBUG CONSOLE TERMINAL PORTS AZURE

[node:0] starting 'node ./bin/www'
PUT /updateTask 200 33.401 ms - 93

≡ ← → Home Workspaces API Network

Search Postman [P Invite](#) [Upgrade](#)

My Workspace New Import Overview index GET New Requests: ● POST createTask PUT update Task + No environment

Collections Environments History

to do list index

GET New Request POST createTask PUT update Task

PUT to do list / index / update Task

localhost:3000/updateTask

Params Authorization Headers (9) Body Scripts Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {  
2   "taskId": "123456",  
3   "taskName": "New name"  
4 }
```

Body Cookies Headers (7) Test Results 200 OK 57 ms 328 B Save as example

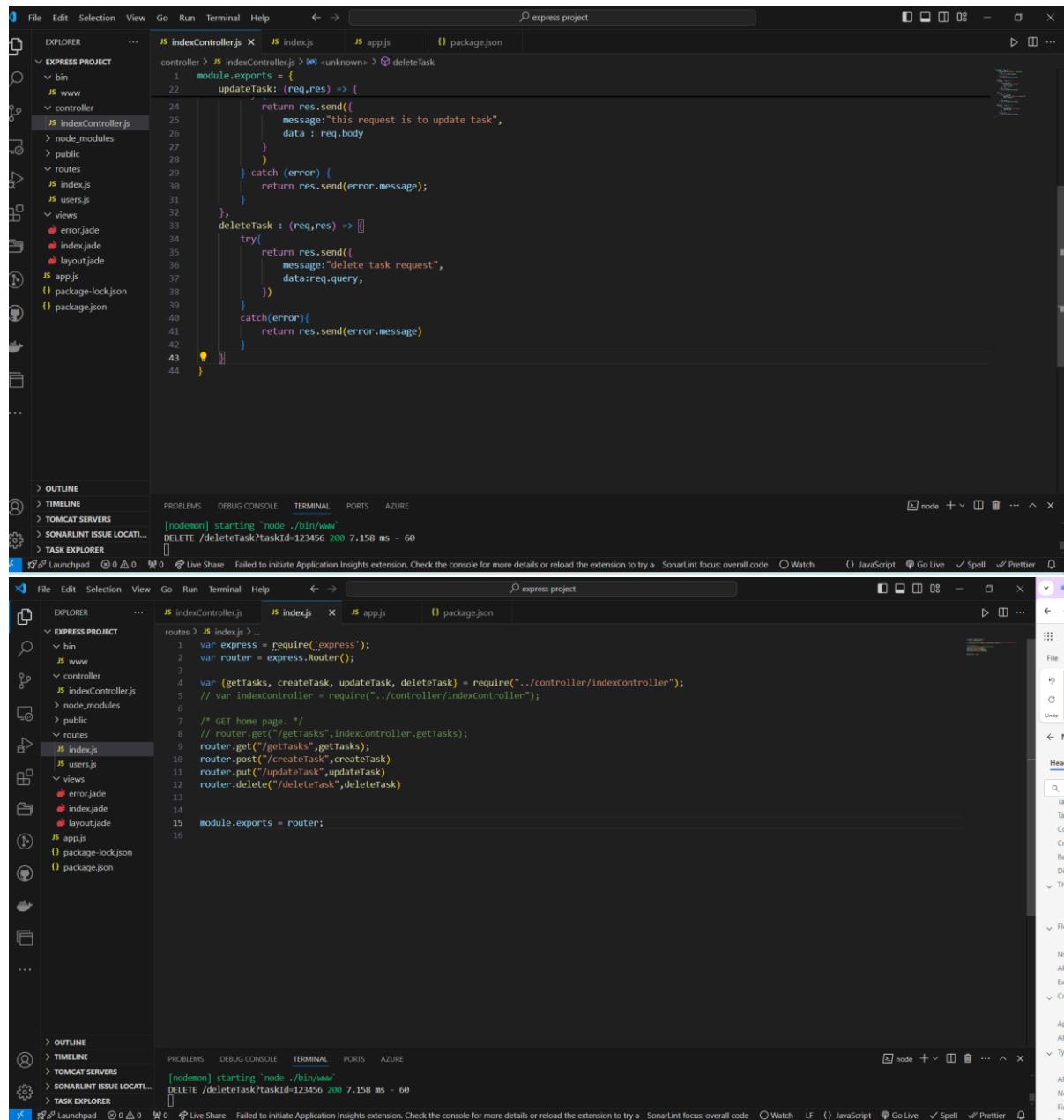
Pretty Raw Preview Visualize JSON

```
1 {  
2   "message": "this request is to update task",  
3   "data": {  
4     "taskId": "123456",  
5     "taskName": "New name"  
6   }  
7 }
```

Postbot Runner Start Proxy Cookies Vault Trash

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'Collections', 'Environments', and 'History'. Below that is a tree view of a collection named 'to do list' with a single item 'index'. Under 'index', there are three requests: 'GET New Request', 'POST createTask', and 'PUT update Task', with 'PUT update Task' being the active one. The main panel shows the 'PUT to do list / index / update Task' request with the URL 'localhost:3000/updateTask'. The 'Body' tab is selected, showing a JSON payload with two fields: 'taskId' set to '123456' and 'taskName' set to 'New name'. Below the request, the response is displayed: a 200 OK status with a message saying 'this request is to update task' and a data object containing the updated task details. At the bottom, there are buttons for 'Postbot', 'Runner', 'Start Proxy', 'Cookies', 'Vault', and 'Trash'.

DeleteTask:



The screenshot displays two instances of the Visual Studio Code interface, each showing a file named `indexController.js` within an "EXPRESS PROJECT" folder structure.

Top Instance:

```
JS indexController.js x JS index.js JS app.js package.json
controller > JS indexController.js > [empty] <unknown> > deleteTask
  1 module.exports = {
  22     updateTask: (req,res) => [
  24         return res.send({
  25             message:"this request is to update task",
  26             data : req.body
  27         }
  28     ) catch (error) {
  29         return res.send(error.message);
  30     }
  31 },
  32 deleteTask : (req,res) => [
  33     try{
  34         return res.send({
  35             message:"delete task request",
  36             data:req.query,
  37         })
  38     } catch(error){
  39         return res.send(error.message)
  40     }
  41 ]
  42 }
  43
  44 }
```

Bottom Instance:

```
JS indexController.js x JS index.js x JS app.js package.json
routes > JS index.js > ...
  1 var express = require('express');
  2 var router = express.Router();
  3
  4 var {getTasks, createTask, updateTask, deleteTask} = require("../controller/indexController");
  5 // var indexController = require("./controller/indexController");
  6
  7 /* GET home page. */
  8 // router.get("/getTasks",indexController.getTasks);
  9 router.get("/getTasks",getTasks);
10 router.post("/createTask",createTask)
11 router.put("/updateTask",updateTask)
12 router.delete("/deleteTask",deleteTask)
13
14
15 module.exports = router;
16
```

In the top instance, there is a syntax error where the `updateTask` function is being used instead of `deleteTask`. The bottom instance shows the correct implementation where `deleteTask` is used.

The screenshot shows the Postman application interface. On the left, the 'My Workspace' sidebar lists collections: 'to do list' (with 'index', 'New Request', 'createTask', 'update Task', and 'delete Task') and environments. The main workspace displays a DELETE request to 'localhost:3000/deleteTask?taskId=123456'. The 'Params' tab shows a 'taskId' parameter with value '123456'. The 'Body' tab shows a JSON response:

```
1 {
2   "message": "delete task request",
3   "data": {
4     "taskId": "123456"
5   }
6 }
```

Validation- JOI

Both frontend and backend validations are necessary

Validation

Joi — an awesome code validation library for Node.js and Express



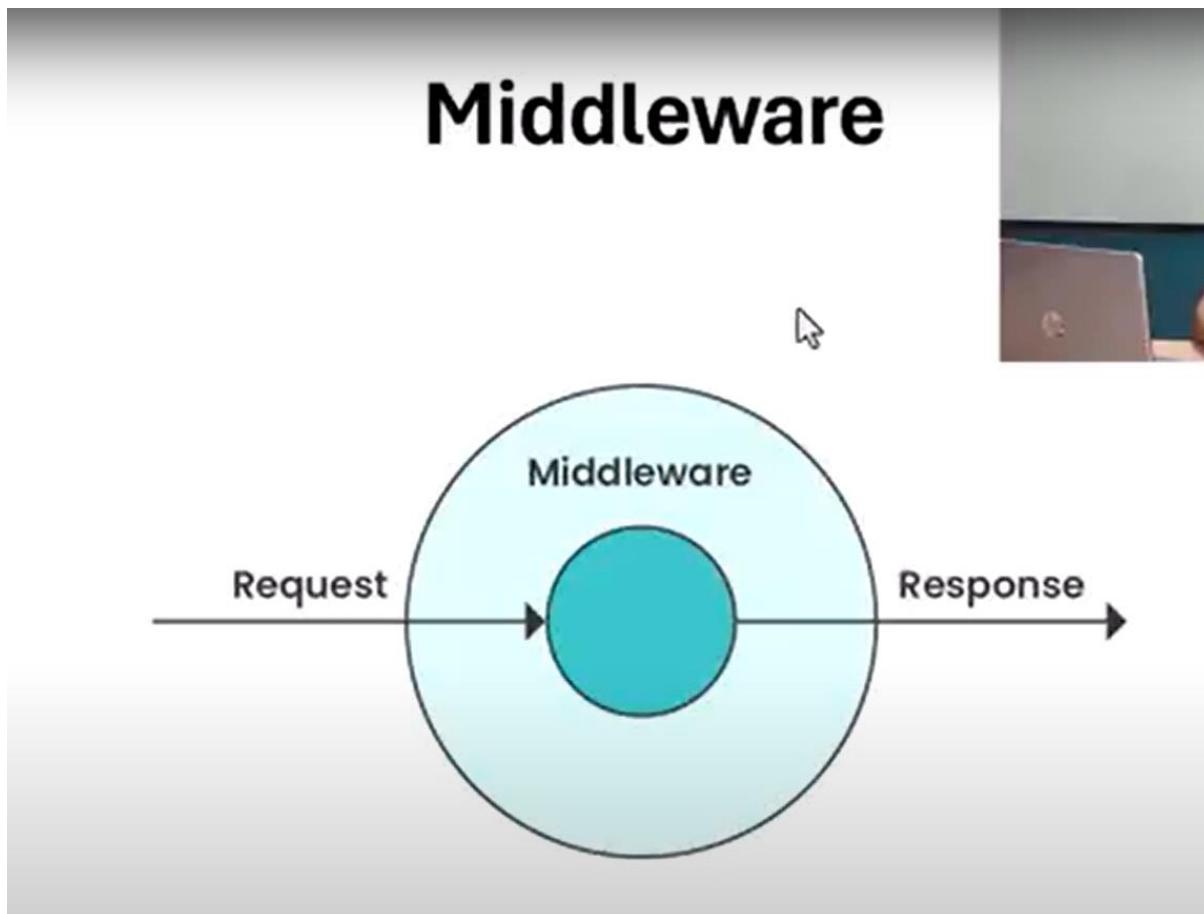
Installing Joi is quite easy. We just need to type:

npm install joi

```
const Joi = require('joi');
const schema = Joi.object().keys({
  name: Joi.string().alphanum().min(3).max(30).required(),
  birthyear: Joi.number().integer().min(1970).max(2013),
});
const dataToValidate = {
  name: 'chris',
  birthyear: 1971
}
const result = Joi.validate(dataToValidate, schema);
```

1. Pkg ko import kia ha aik variable k ander
2. Create schema
 - a. What is schema?
 - b. Q k JOI aik json object ko validate kr raha ha q k hamara input data ya tou body parameters hote hain ya query parameters in both the cases aik json obj ha then obj.keys ka matlab ha k wo un keys ko in eys k upper map krega jo aapne yhaan define ki hain pehle key match hogi agr wo key exit kregi tou wo validate krega aur wo key hi nhi hogi tou error generate hojayega

Middleware



Request jb client se accept krte hain tou pehle usko aik function me pas krke validate krte hain phir backend ko send krte hain un functions ko middleware kehte hain

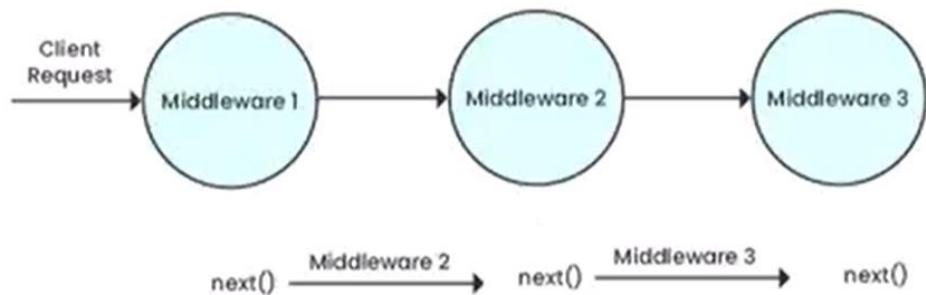
Request aur controller k dermian me aik middle function hota ha -->middleware

Middleware if validate --> controller otherwise sent response back to client

Next function

e

Middleware



The screenshot shows two instances of the Visual Studio Code (VS Code) interface, both displaying an Express.js project named "express project".

Top Window:

- Explorer View:** Shows the project structure under "EXPRESS PROJECT". The "indexValidation.js" file is selected.
- Terminal:** Displays the command `E:\practise and udemy courses\WERN Course\express js\express project>npm install joi`.
- Code Editor:** Shows the content of the "indexValidation.js" file:

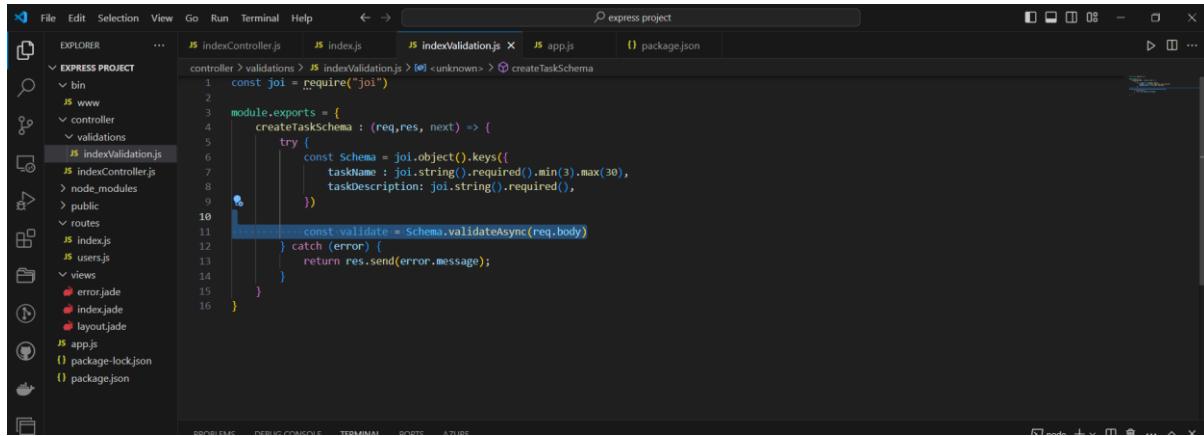
```
const joi = require("joi");
module.exports = {
  createTaskSchema: (req, res, next) => {
    try {
      const Schema = joi.object().keys({
        taskName: joi.string().required().min(3).max(30),
        taskDescription: joi.string().required(),
      });
      const validate = Schema.validateAsync();
      validate.catch(error) {
        return res.send(error.message);
      }
    }
  }
};
```

Bottom Window:

- Explorer View:** Shows the project structure under "EXPRESS PROJECT". The "indexValidation.js" file is selected.
- Terminal:** Displays the output of the command `node ./bin/www` from nodemon, showing repeated starts and restarts due to changes.
- Code Editor:** Shows the content of the "indexValidation.js" file, identical to the one in the top window.

ValidateAsync():

We are using validateAsync and not validate because validate promise nhi h , we want jab tk validation complete na hojaye request aage na brhe. Therefore we are using validateAsync jo k 1 promise h jo make sure krega k wo kuch return hoga tou uska hum wait krsakte hain



```
const Joi = require("joi");

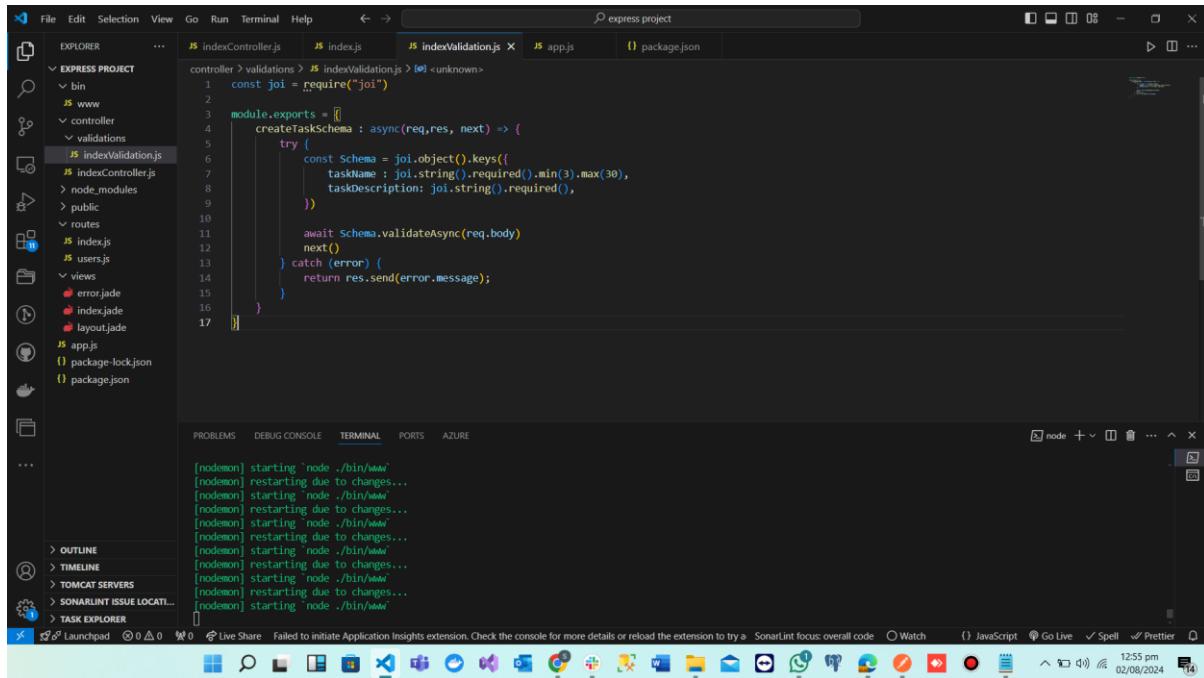
module.exports = {
  createTaskSchema: (req, res, next) => {
    try {
      const Schema = Joi.object().keys({
        taskName: Joi.string().required().min(3).max(30),
        taskDescription: Joi.string().required(),
      });
      const validate = Schema.validateAsync(req.body);
    } catch (error) {
      return res.send(error.message);
    }
  }
};
```

Q k ye aik post req ha aur post req front end se request aur body accept krti h tou ham yahan req.body add krenge, req

.body k ander jitni keys ha wo un keys ko schema ki keys k sath compare krega aur unko validate krega, in that case agr kahin pe bhi comparison equal nhi hota tou wo usko return krdega aur error generate krdega

Promise k lye zaroorat hoti h k hum await add kren

Await hamesha async func k sath aata ha



```
const Joi = require("joi");

module.exports = {
  createTaskSchema: async(req, res, next) => {
    try {
      const Schema = Joi.object().keys({
        taskName: Joi.string().required().min(3).max(30),
        taskDescription: Joi.string().required(),
      });
      await Schema.validateAsync(req.body);
      next();
    } catch (error) {
      return res.send(error.message);
    }
  }
};
```

The terminal output shows the application starting and restarting due to changes:

```
[nodemon] starting 'node ./bin/www'
[nodemon] restarting due to changes...
[nodemon] starting 'node ./bin/www'
[nodemon] restarting due to changes...
[nodemon] starting 'node ./bin/www'
[nodemon] restarting due to changes...
[nodemon] starting 'node ./bin/www'
[nodemon] restarting due to changes...
[nodemon] starting 'node ./bin/www'
```