



[View, add and edit your notes in the app](#)

# Complete React course with projects | part 1

Generated on December 29, 2023

## Summary

Notes

Screenshots

Bookmarks

12

54

0

↳ Why to learn React?

- hype, job, trend, build UI
- makes easy to manage & build complex front end

↳ When should I learn React?

- After mastering JS
- most project don't need react in initial phase

↳ Why react was created

→ Ghost message Problem

No consistency in UI

▷ 4:55

State → JS & UI → DOM  
Khan Academy → Unsplash

↳ Don't learn React if:  
→ you don't know how JS works OR DOM works

Watch my Browser's inner working video

↳ React Learning Process

↳ go in-depth  
↳ by making projects (one topic at a time)

Babel, fibre, Virtual DOM, diff algo, hydration  
Todo, calculator, Github API,

↳ React is a library  
↳ framework VS Library

▷ 15:56

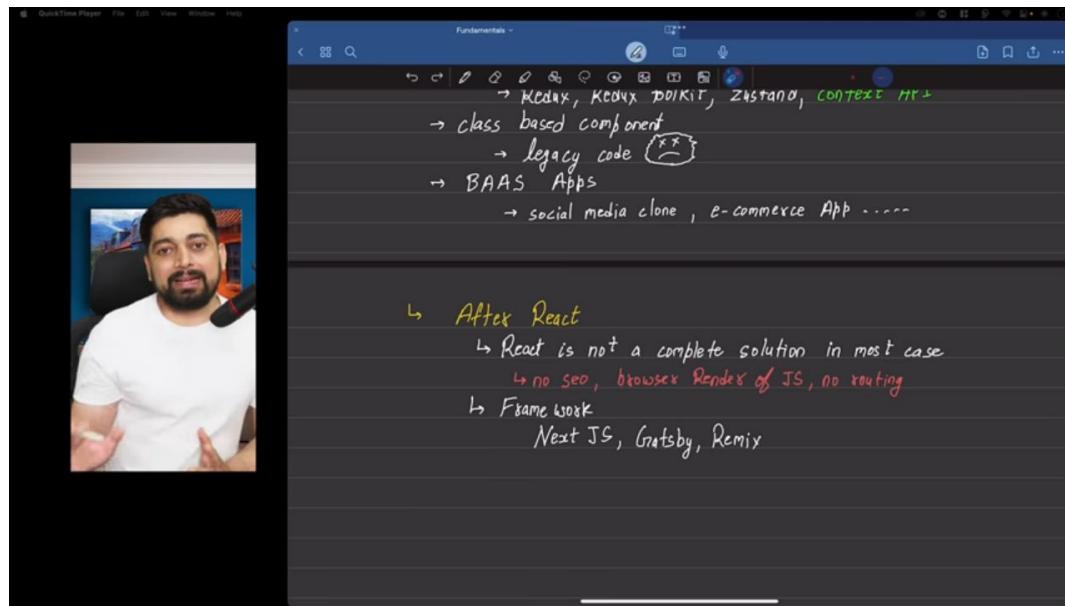
↳ Topics to learn

- core of React (State or UI manipulation, JSX)
- component Reusability
- Reusing of component (Props)
- How to propagate change (hooks)

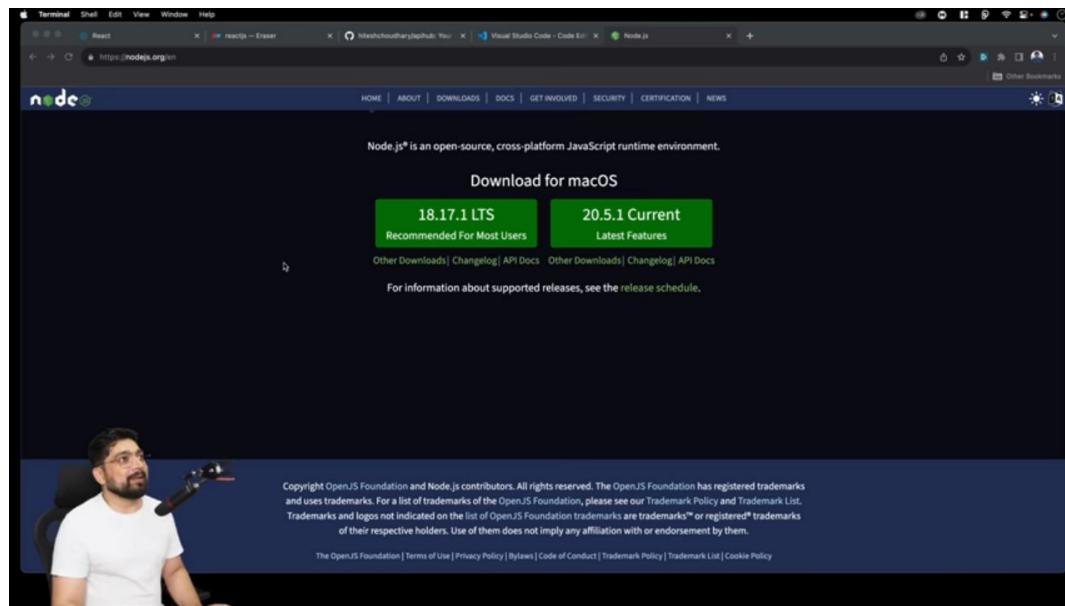
↳ Additional Addon to React

- Router (React don't have Router)
- state management (React don't have state management)
  - Redux, Redux Toolkit, zustand, context API
- class based component
  - legacy code 😞
- BAAS Apps
  - social media clone, e-commerce App .....

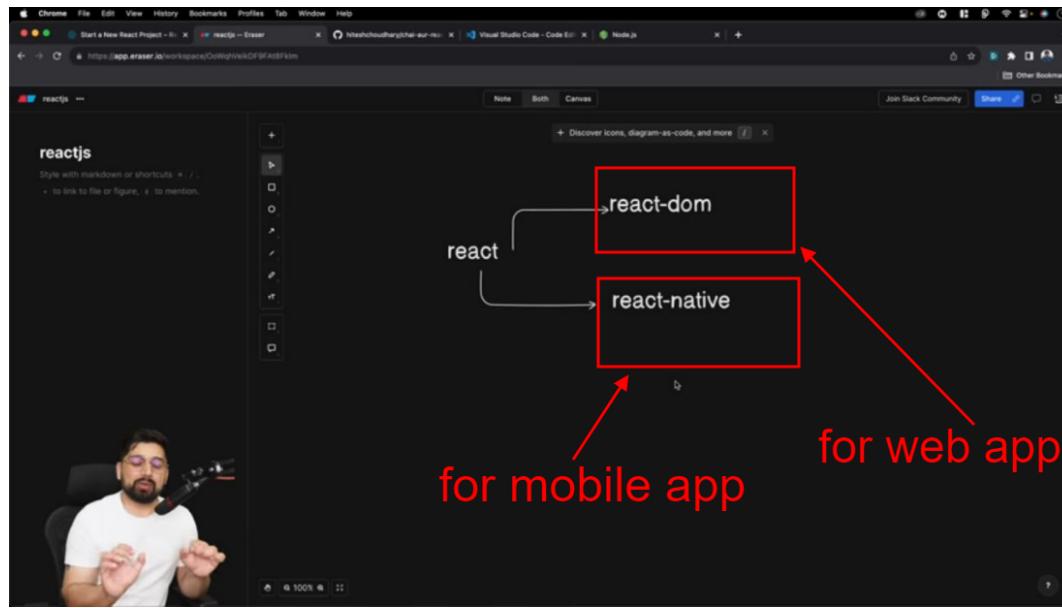
▷ 23:04



▷ 29:12



▷ 33:23



▶ 40:55

EXPLORER: CHAI-AUR-...

package.json

```

1 {
2   "name": "01basicreact",
3   "version": "0.1.0",      js k via testing k leya de gai hai
4   "private": true,
5   "dependencies": {
6     "@testing-library/jest-dom": "^5.17.0",
7     "@testing-library/react": "^13.4.0",
8     "@testing-library/user-event": "^13.5.0",
9     "react": "^18.2.0",
10    "react-dom": "^18.2.0",
11    "react-scripts": "5.0.1",
12    "web-vitals": "^2.1.4"
13  },
14  "scripts": {
15    "start": "react-scripts start",
16    "build": "react-scripts build",
17    "test": "react-scripts test",
18    "eject": "react-scripts eject"
19  }

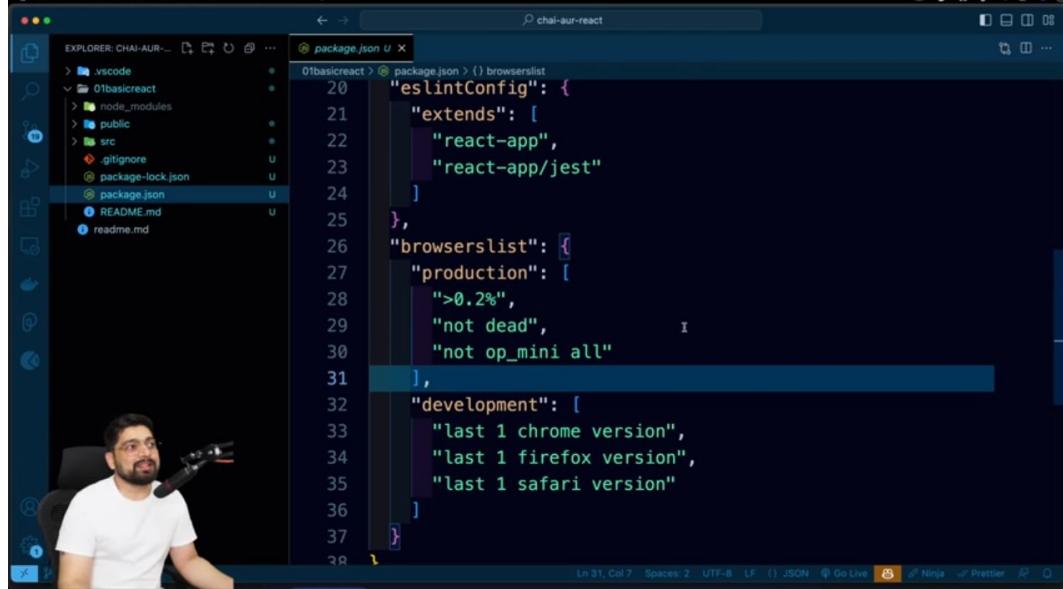
```

js k via testing k leya de gai hai

ya do main librares hai

performance check karta hai

▶ 45:02

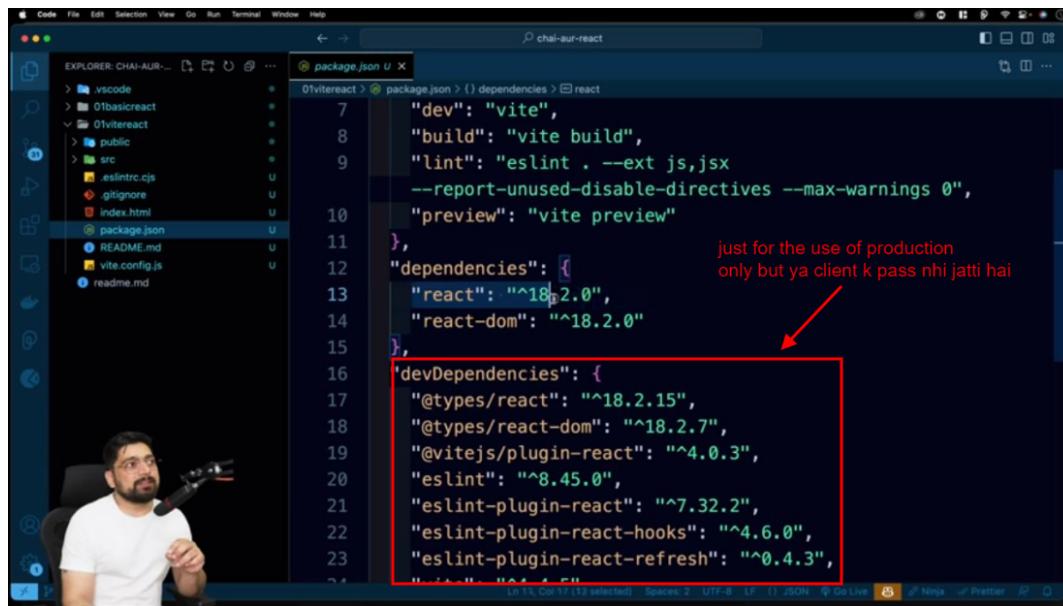


A screenshot of a video call interface. On the left, there's a sidebar with icons for file operations like Open, Save, and Close. The main area shows a code editor for a file named 'package.json'. The code is as follows:

```
20 "eslintConfig": {
21   "extends": [
22     "react-app",
23     "react-app/jest"
24   ],
25 },
26 "browserslist": [
27   "production": [
28     ">0.2%",
29     "not dead",
30     "not op_mini all"
31   ],
32   "development": [
33     "last 1 chrome version",
34     "last 1 firefox version",
35     "last 1 safari version"
36   ]
37 }
```

The status bar at the bottom of the code editor shows 'Ln 31, Col 7' and other details like 'Spaces: 2', 'UTF-8', 'LF', 'JSON', 'Go Live', 'Ninja', and 'Prettier'.

▶ 45:48

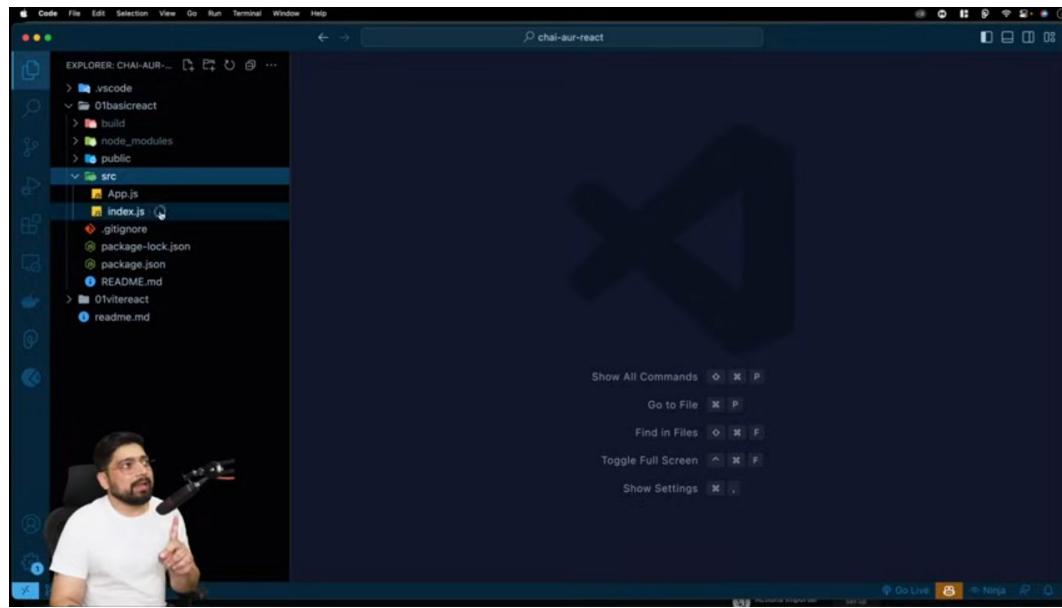


A screenshot of a video call interface. On the left, there's a sidebar with icons for file operations like Open, Save, and Close. The main area shows a code editor for a file named 'package.json'. The code is as follows:

```
7   "dev": "vite",
8   "build": "vite build",
9   "lint": "eslint . --ext js,jsx
--report-unused-disable-directives --max-warnings 0",
10  "preview": "vite preview"
},
11 "dependencies": just for the use of production
12  "react": "^18.2.0", only but ya client k pass nhi jatti hai
13  "react-dom": "^18.2.0"
},
14
15 "devDependencies": {
16   "@types/react": "^18.2.15",
17   "@types/react-dom": "^18.2.7",
18   "@vitejs/plugin-react": "^4.0.3",
19   "eslint": "^8.45.0",
20   "eslint-plugin-react": "^7.32.2",
21   "eslint-plugin-react-hooks": "^4.6.0",
22   "eslint-plugin-react-refresh": "^0.4.3",
23 }
```

A red box highlights the 'devDependencies' section, and a red arrow points from the text 'just for the use of production' to this box. The status bar at the bottom of the code editor shows 'Ln 15, Col 7' and other details like 'Spaces: 2', 'UTF-8', 'LF', 'JSON', 'Go Live', 'Ninja', and 'Prettier'.

▶ 53:06

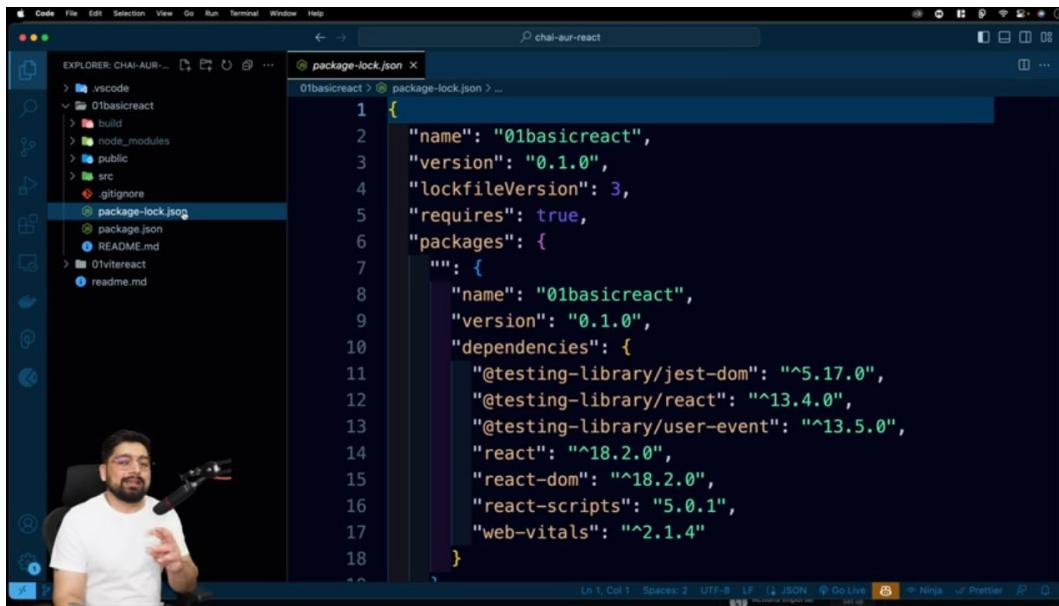


▶ 1:08:05

```
1 {  
2   "name": "01basicreact",  
3   "version": "0.1.0",  
4   "lockfileVersion": 3,  
5   "requires": true,  
6   "packages": {  
7     "": {  
8       "name": "01basicreact",  
9       "version": "0.1.0",  
10      "dependencies": {  
11        "@testing-library/jest-dom": "^5.17.0",  
12        "@testing-library/react": "^13.4.0",  
13        "@testing-library/user-event": "^13.5.0",  
14        "react": "^18.2.0",  
15        "react-dom": "^18.2.0",  
16        "react-scripts": "5.0.1",  
17        "web-vitals": "^2.1.4"  
18      }  
19    }  
20  }  
21
```

The screenshot shows the 'package-lock.json' file open in the main editor area of VS Code. The JSON content defines a package named '01basicreact' with version '0.1.0' and lockfile version 3. It includes a single entry for the root package, which has dependencies for '@testing-library/jest-dom', '@testing-library/react', '@testing-library/user-event', 'react', 'react-dom', 'react-scripts', and 'web-vitals'. The status bar at the bottom indicates the file is in JSON format.

▶ 1:09:12

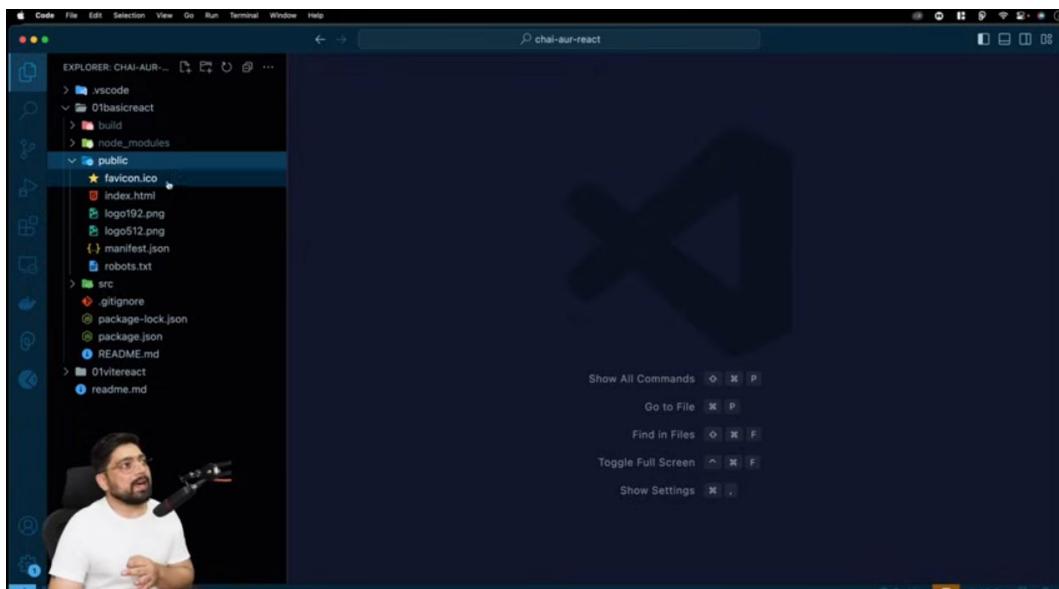


A screenshot of a video call interface. On the left, there's a video feed of a man with a beard and glasses, wearing a white t-shirt, sitting in a black chair. He is gesturing with his right hand. To his right is a code editor window titled "package-lock.json". The file content is as follows:

```
1 {  
2   "name": "01basicreact",  
3   "version": "0.1.0",  
4   "lockfileVersion": 3,  
5   "requires": true,  
6   "packages": {  
7     "": {  
8       "name": "01basicreact",  
9       "version": "0.1.0",  
10      "dependencies": {  
11        "@testing-library/jest-dom": "^5.17.0",  
12        "@testing-library/react": "^13.4.0",  
13        "@testing-library/user-event": "^13.5.0",  
14        "react": "^18.2.0",  
15        "react-dom": "^18.2.0",  
16        "react-scripts": "5.0.1",  
17        "web-vitals": "^2.1.4"  
18      }  
19    }  
20  }  
21  
22 }  
23  
24 }  
25  
26 }  
27  
28 }  
29  
30 }  
31  
32 }  
33  
34 }  
35  
36 }  
37  
38 }  
39  
40 }  
41  
42 }  
43  
44 }  
45  
46 }  
47  
48 }  
49  
50 }  
51  
52 }  
53  
54 }  
55  
56 }  
57  
58 }  
59  
60 }  
61  
62 }  
63  
64 }  
65  
66 }  
67  
68 }  
69  
70 }  
71  
72 }  
73  
74 }  
75  
76 }  
77  
78 }  
79  
80 }  
81  
82 }  
83  
84 }  
85  
86 }  
87  
88 }  
89  
90 }  
91  
92 }  
93  
94 }  
95  
96 }  
97  
98 }  
99  
100 }  
101 }  
102 }  
103 }  
104 }  
105 }  
106 }  
107 }  
108 }  
109 }  
110 }  
111 }  
112 }  
113 }  
114 }  
115 }  
116 }  
117 }  
118 }  
119 }  
120 }  
121 }  
122 }  
123 }  
124 }  
125 }  
126 }  
127 }  
128 }  
129 }  
130 }  
131 }  
132 }  
133 }  
134 }  
135 }  
136 }  
137 }  
138 }  
139 }  
140 }  
141 }  
142 }  
143 }  
144 }  
145 }  
146 }  
147 }  
148 }  
149 }  
150 }  
151 }  
152 }  
153 }  
154 }  
155 }  
156 }  
157 }  
158 }  
159 }  
160 }  
161 }  
162 }  
163 }  
164 }  
165 }  
166 }  
167 }  
168 }  
169 }  
170 }  
171 }  
172 }  
173 }  
174 }  
175 }  
176 }  
177 }  
178 }  
179 }  
180 }  
181 }  
182 }  
183 }  
184 }  
185 }  
186 }  
187 }  
188 }  
189 }  
190 }  
191 }  
192 }  
193 }  
194 }  
195 }  
196 }  
197 }  
198 }  
199 }  
200 }  
201 }  
202 }  
203 }  
204 }  
205 }  
206 }  
207 }  
208 }  
209 }  
210 }  
211 }  
212 }  
213 }  
214 }  
215 }  
216 }  
217 }  
218 }  
219 }  
220 }  
221 }  
222 }  
223 }  
224 }  
225 }  
226 }  
227 }  
228 }  
229 }  
230 }  
231 }  
232 }  
233 }  
234 }  
235 }  
236 }  
237 }  
238 }  
239 }  
240 }  
241 }  
242 }  
243 }  
244 }  
245 }  
246 }  
247 }  
248 }  
249 }  
250 }  
251 }  
252 }  
253 }  
254 }  
255 }  
256 }  
257 }  
258 }  
259 }  
260 }  
261 }  
262 }  
263 }  
264 }  
265 }  
266 }  
267 }  
268 }  
269 }  
270 }  
271 }  
272 }  
273 }  
274 }  
275 }  
276 }  
277 }  
278 }  
279 }  
280 }  
281 }  
282 }  
283 }  
284 }  
285 }  
286 }  
287 }  
288 }  
289 }  
290 }  
291 }  
292 }  
293 }  
294 }  
295 }  
296 }  
297 }  
298 }  
299 }  
300 }  
301 }  
302 }  
303 }  
304 }  
305 }  
306 }  
307 }  
308 }  
309 }  
310 }  
311 }  
312 }  
313 }  
314 }  
315 }  
316 }  
317 }  
318 }  
319 }  
320 }  
321 }  
322 }  
323 }  
324 }  
325 }  
326 }  
327 }  
328 }  
329 }  
330 }  
331 }  
332 }  
333 }  
334 }  
335 }  
336 }  
337 }  
338 }  
339 }  
340 }  
341 }  
342 }  
343 }  
344 }  
345 }  
346 }  
347 }  
348 }  
349 }  
350 }  
351 }  
352 }  
353 }  
354 }  
355 }  
356 }  
357 }  
358 }  
359 }  
360 }  
361 }  
362 }  
363 }  
364 }  
365 }  
366 }  
367 }  
368 }  
369 }  
370 }  
371 }  
372 }  
373 }  
374 }  
375 }  
376 }  
377 }  
378 }  
379 }  
380 }  
381 }  
382 }  
383 }  
384 }  
385 }  
386 }  
387 }  
388 }  
389 }  
390 }  
391 }  
392 }  
393 }  
394 }  
395 }  
396 }  
397 }  
398 }  
399 }  
400 }  
401 }  
402 }  
403 }  
404 }  
405 }  
406 }  
407 }  
408 }  
409 }  
410 }  
411 }  
412 }  
413 }  
414 }  
415 }  
416 }  
417 }  
418 }  
419 }  
420 }  
421 }  
422 }  
423 }  
424 }  
425 }  
426 }  
427 }  
428 }  
429 }  
430 }  
431 }  
432 }  
433 }  
434 }  
435 }  
436 }  
437 }  
438 }  
439 }  
440 }  
441 }  
442 }  
443 }  
444 }  
445 }  
446 }  
447 }  
448 }  
449 }  
450 }  
451 }  
452 }  
453 }  
454 }  
455 }  
456 }  
457 }  
458 }  
459 }  
460 }  
461 }  
462 }  
463 }  
464 }  
465 }  
466 }  
467 }  
468 }  
469 }  
470 }  
471 }  
472 }  
473 }  
474 }  
475 }  
476 }  
477 }  
478 }  
479 }  
480 }  
481 }  
482 }  
483 }  
484 }  
485 }  
486 }  
487 }  
488 }  
489 }  
490 }  
491 }  
492 }  
493 }  
494 }  
495 }  
496 }  
497 }  
498 }  
499 }  
500 }  
501 }  
502 }  
503 }  
504 }  
505 }  
506 }  
507 }  
508 }  
509 }  
510 }  
511 }  
512 }  
513 }  
514 }  
515 }  
516 }  
517 }  
518 }  
519 }  
520 }  
521 }  
522 }  
523 }  
524 }  
525 }  
526 }  
527 }  
528 }  
529 }  
530 }  
531 }  
532 }  
533 }  
534 }  
535 }  
536 }  
537 }  
538 }  
539 }  
540 }  
541 }  
542 }  
543 }  
544 }  
545 }  
546 }  
547 }  
548 }  
549 }  
550 }  
551 }  
552 }  
553 }  
554 }  
555 }  
556 }  
557 }  
558 }  
559 }  
560 }  
561 }  
562 }  
563 }  
564 }  
565 }  
566 }  
567 }  
568 }  
569 }  
570 }  
571 }  
572 }  
573 }  
574 }  
575 }  
576 }  
577 }  
578 }  
579 }  
580 }  
581 }  
582 }  
583 }  
584 }  
585 }  
586 }  
587 }  
588 }  
589 }  
590 }  
591 }  
592 }  
593 }  
594 }  
595 }  
596 }  
597 }  
598 }  
599 }  
600 }  
601 }  
602 }  
603 }  
604 }  
605 }  
606 }  
607 }  
608 }  
609 }  
610 }  
611 }  
612 }  
613 }  
614 }  
615 }  
616 }  
617 }  
618 }  
619 }  
620 }  
621 }  
622 }  
623 }  
624 }  
625 }  
626 }  
627 }  
628 }  
629 }  
630 }  
631 }  
632 }  
633 }  
634 }  
635 }  
636 }  
637 }  
638 }  
639 }  
640 }  
641 }  
642 }  
643 }  
644 }  
645 }  
646 }  
647 }  
648 }  
649 }  
650 }  
651 }  
652 }  
653 }  
654 }  
655 }  
656 }  
657 }  
658 }  
659 }  
660 }  
661 }  
662 }  
663 }  
664 }  
665 }  
666 }  
667 }  
668 }  
669 }  
670 }  
671 }  
672 }  
673 }  
674 }  
675 }  
676 }  
677 }  
678 }  
679 }  
680 }  
681 }  
682 }  
683 }  
684 }  
685 }  
686 }  
687 }  
688 }  
689 }  
690 }  
691 }  
692 }  
693 }  
694 }  
695 }  
696 }  
697 }  
698 }  
699 }  
700 }  
701 }  
702 }  
703 }  
704 }  
705 }  
706 }  
707 }  
708 }  
709 }  
710 }  
711 }  
712 }  
713 }  
714 }  
715 }  
716 }  
717 }  
718 }  
719 }  
720 }  
721 }  
722 }  
723 }  
724 }  
725 }  
726 }  
727 }  
728 }  
729 }  
730 }  
731 }  
732 }  
733 }  
734 }  
735 }  
736 }  
737 }  
738 }  
739 }  
740 }  
741 }  
742 }  
743 }  
744 }  
745 }  
746 }  
747 }  
748 }  
749 }  
750 }  
751 }  
752 }  
753 }  
754 }  
755 }  
756 }  
757 }  
758 }  
759 }  
760 }  
761 }  
762 }  
763 }  
764 }  
765 }  
766 }  
767 }  
768 }  
769 }  
770 }  
771 }  
772 }  
773 }  
774 }  
775 }  
776 }  
777 }  
778 }  
779 }  
780 }  
781 }  
782 }  
783 }  
784 }  
785 }  
786 }  
787 }  
788 }  
789 }  
790 }  
791 }  
792 }  
793 }  
794 }  
795 }  
796 }  
797 }  
798 }  
799 }  
800 }  
801 }  
802 }  
803 }  
804 }  
805 }  
806 }  
807 }  
808 }  
809 }  
810 }  
811 }  
812 }  
813 }  
814 }  
815 }  
816 }  
817 }  
818 }  
819 }  
820 }  
821 }  
822 }  
823 }  
824 }  
825 }  
826 }  
827 }  
828 }  
829 }  
830 }  
831 }  
832 }  
833 }  
834 }  
835 }  
836 }  
837 }  
838 }  
839 }  
840 }  
841 }  
842 }  
843 }  
844 }  
845 }  
846 }  
847 }  
848 }  
849 }  
850 }  
851 }  
852 }  
853 }  
854 }  
855 }  
856 }  
857 }  
858 }  
859 }  
860 }  
861 }  
862 }  
863 }  
864 }  
865 }  
866 }  
867 }  
868 }  
869 }  
870 }  
871 }  
872 }  
873 }  
874 }  
875 }  
876 }  
877 }  
878 }  
879 }  
880 }  
881 }  
882 }  
883 }  
884 }  
885 }  
886 }  
887 }  
888 }  
889 }  
890 }  
891 }  
892 }  
893 }  
894 }  
895 }  
896 }  
897 }  
898 }  
899 }  
900 }  
901 }  
902 }  
903 }  
904 }  
905 }  
906 }  
907 }  
908 }  
909 }  
910 }  
911 }  
912 }  
913 }  
914 }  
915 }  
916 }  
917 }  
918 }  
919 }  
920 }  
921 }  
922 }  
923 }  
924 }  
925 }  
926 }  
927 }  
928 }  
929 }  
930 }  
931 }  
932 }  
933 }  
934 }  
935 }  
936 }  
937 }  
938 }  
939 }  
940 }  
941 }  
942 }  
943 }  
944 }  
945 }  
946 }  
947 }  
948 }  
949 }  
950 }  
951 }  
952 }  
953 }  
954 }  
955 }  
956 }  
957 }  
958 }  
959 }  
960 }  
961 }  
962 }  
963 }  
964 }  
965 }  
966 }  
967 }  
968 }  
969 }  
970 }  
971 }  
972 }  
973 }  
974 }  
975 }  
976 }  
977 }  
978 }  
979 }  
980 }  
981 }  
982 }  
983 }  
984 }  
985 }  
986 }  
987 }  
988 }  
989 }  
990 }  
991 }  
992 }  
993 }  
994 }  
995 }  
996 }  
997 }  
998 }  
999 }  
1000 }
```

dependencey yahi lock hoti hai

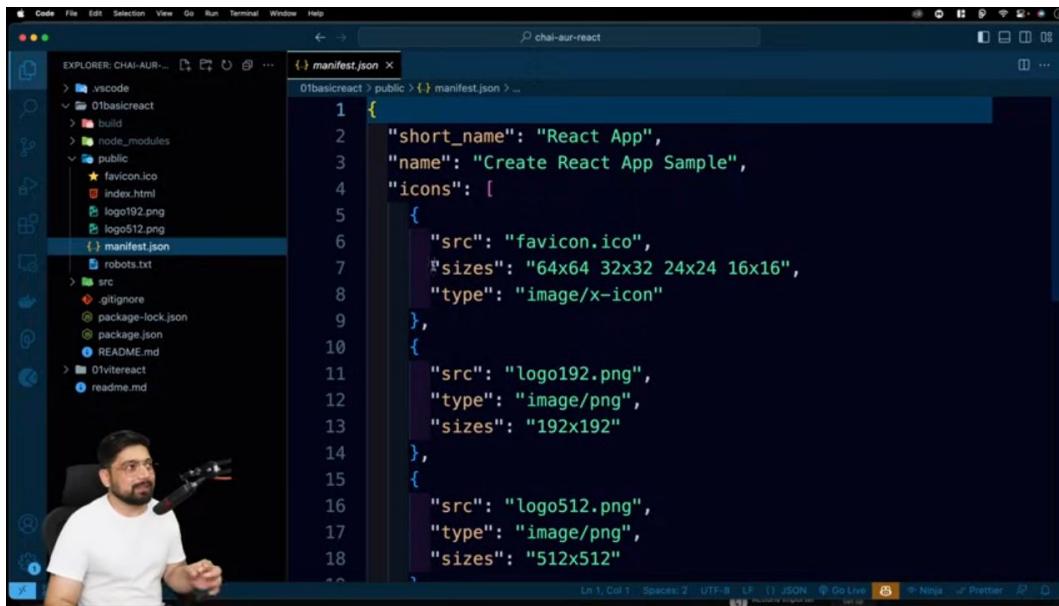
▶ 1:09:18



A screenshot of a video call interface. On the left, there's a video feed of a man with a beard and glasses, wearing a white t-shirt, sitting in a black chair. He is gesturing with his hands. To his right is a code editor window showing the contents of the "public" folder. The folder contains the following files and folders:

- favicon.ico
- index.html
- logo192.png
- logo512.png
- manifest.json
- robots.txt

▶ 1:09:35

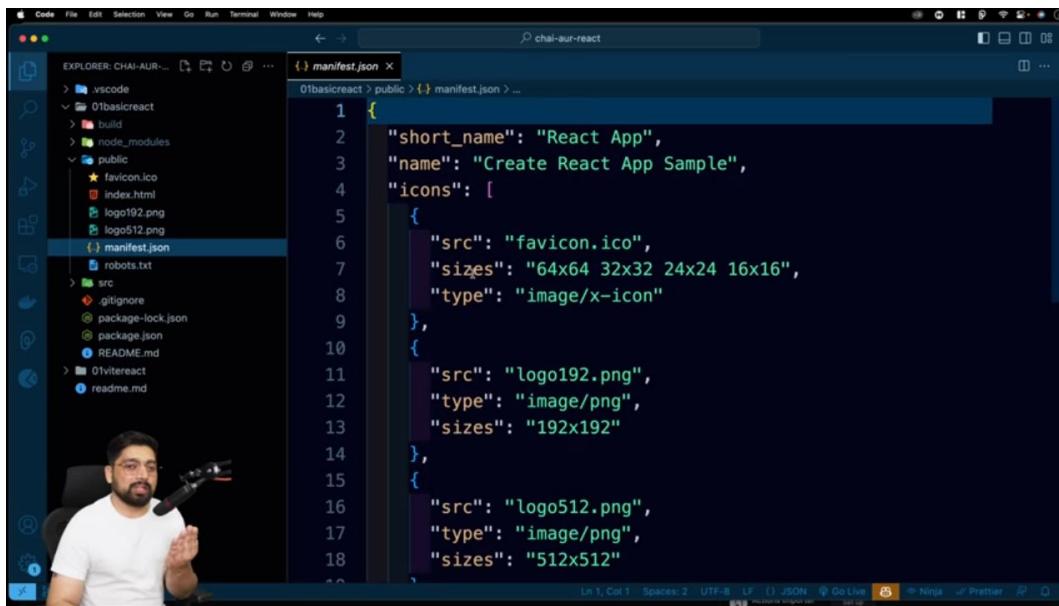


A screenshot of a video call interface. On the left, there's a video feed of a man with a beard and glasses, wearing a white t-shirt, sitting in a black chair and gesturing with his hands. He appears to be speaking. To his right is a code editor window titled "manifest.json". The code editor shows the following JSON configuration:

```
1 {
2   "short_name": "React App",
3   "name": "Create React App Sample",
4   "icons": [
5     {
6       "src": "favicon.ico",
7       "sizes": "64x64 32x32 24x24 16x16",
8       "type": "image/x-icon"
9     },
10    {
11      "src": "logo192.png",
12      "type": "image/png",
13      "sizes": "192x192"
14    },
15    {
16      "src": "logo512.png",
17      "type": "image/png",
18      "sizes": "512x512"
19    }
]
```

The code editor has a dark theme with syntax highlighting. Below the code editor, the status bar shows "Ln 1, Col 1" and other standard editor icons.

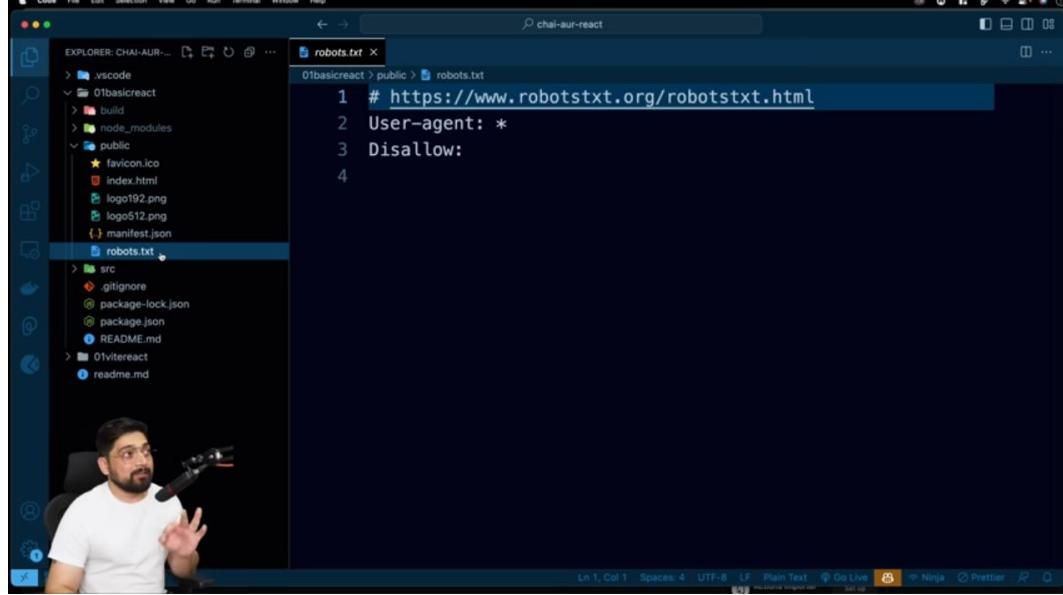
▶ 1:09:55



A screenshot of a video call interface, identical to the one above. It shows the same man speaking and the same code editor window displaying the manifest.json file. The JSON configuration is the same as in the previous screenshot.

manifest file mobile app ka kam aeyti hai

▶ 1:09:58

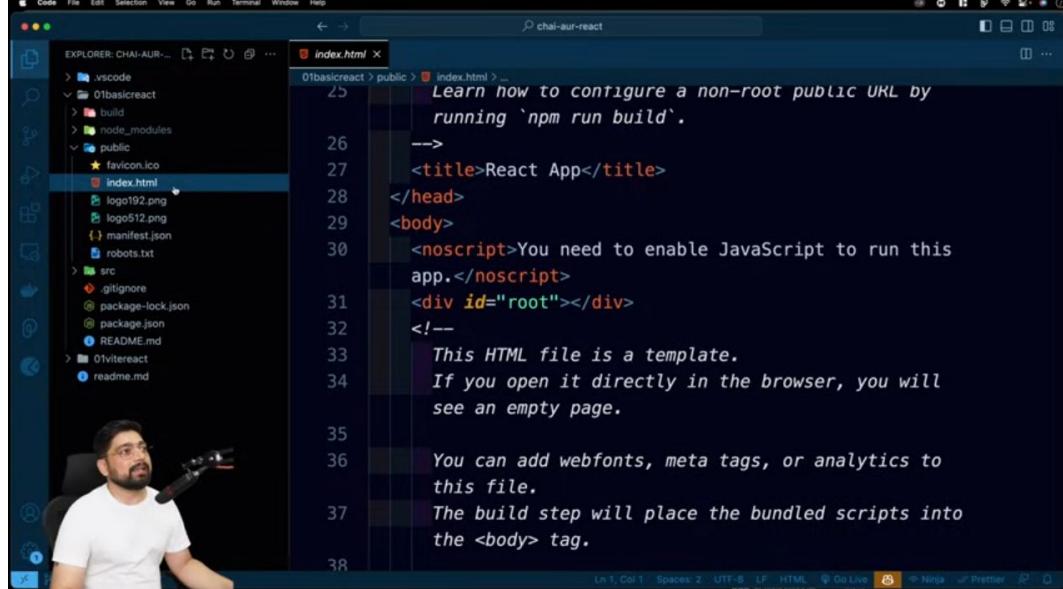


A screenshot of a video call interface. In the foreground, a person with a beard and white shirt is speaking into a microphone. In the background, a VS Code window is open, showing the contents of a 'robots.txt' file. The file contains the following code:

```
1 # https://www.robotstxt.org/robotstxt.html
2 User-agent: *
3 Disallow:
4
```

🔊 ya searchengain k leya hai ho na ho farka nhi

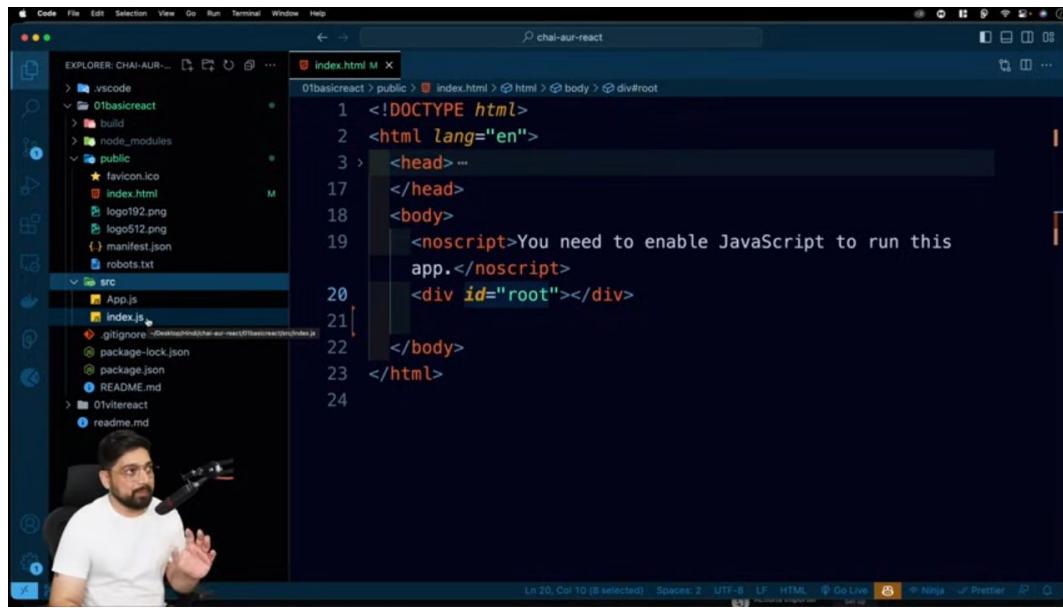
▶ 1:10:13



A screenshot of a video call interface. In the foreground, a person with a beard and white shirt is speaking into a microphone. In the background, a VS Code window is open, showing the contents of an 'index.html' file. The file contains the following code and documentation:

```
25 <!--
26 <!-->
27 <title>React App</title>
28 </head>
29 <body>
30 <noscript>You need to enable JavaScript to run this
31 app.</noscript>
32 <div id="root"></div>
33 <!--
34 This HTML file is a template.
35 If you open it directly in the browser, you will
36 see an empty page.
37
38 You can add webfonts, meta tags, or analytics to
this file.
The build step will place the bundled scripts into
the <body> tag.
```

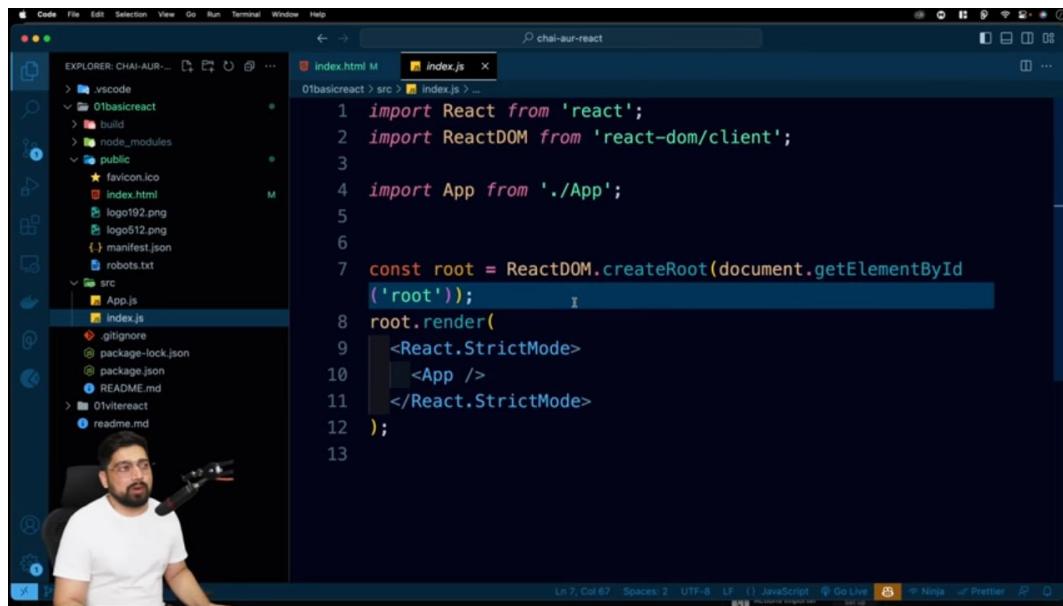
▶ 1:10:14



A screenshot of a video call interface. On the left, there's a video feed of a man with a beard, wearing a white t-shirt, sitting in a chair and gesturing with his hands. He appears to be speaking. To his right is a code editor window titled "chai-aur-react". The editor shows the file "index.html" with the following content:

```
<!DOCTYPE html>
<html lang="en">
  <head> ...
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>
  </body>
</html>
```

▶ 1:11:54



A screenshot of a video call interface. On the left, there's a video feed of a man with a beard, wearing a white t-shirt, sitting in a chair and gesturing with his hands. To his right is a code editor window titled "chai-aur-react". The editor shows the file "index.js" with the following content:

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

import App from './App';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);

```

react apny ap main khod ka dom bhi bna ta hai

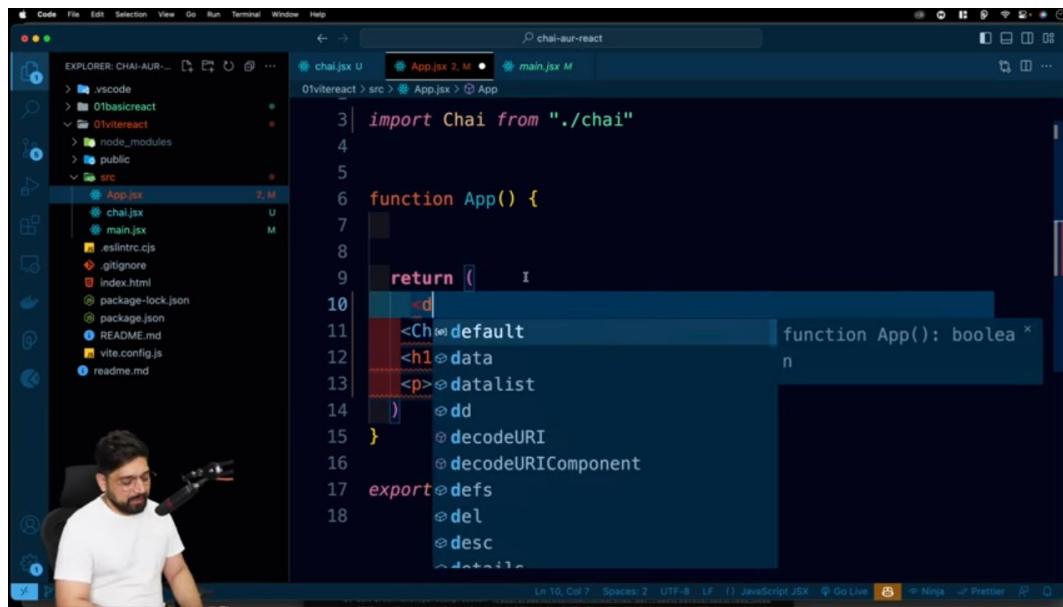
▶ 1:13:58

- react function lay rah hai or function k underunder html readnder kar wa rah hai

▶ 1:17:44

funtion ka name capital main hona chahya

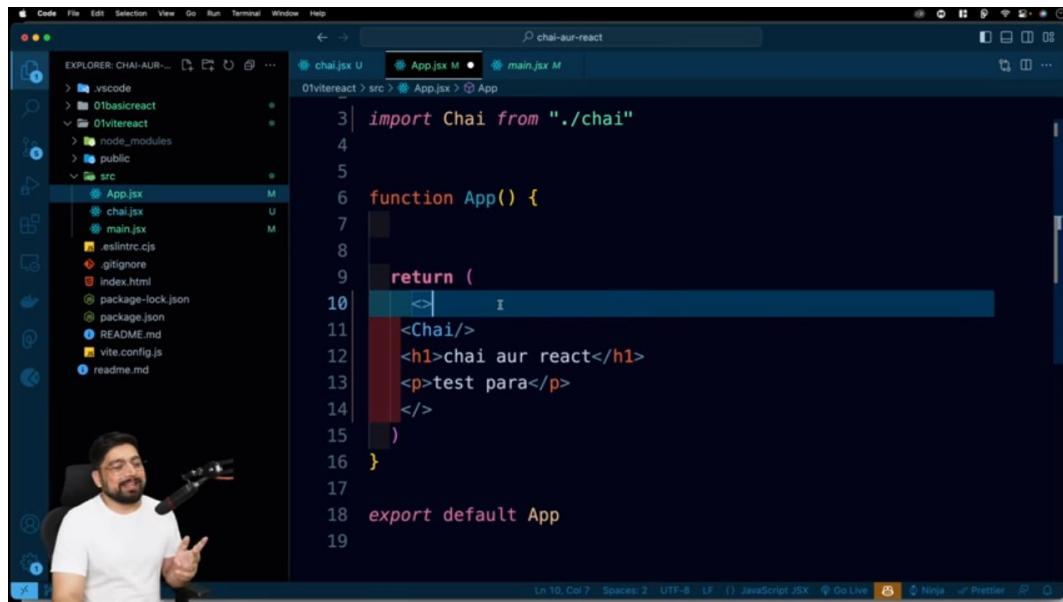
▷ 1:25:14



```
import Chai from "./chai"
function App() {
  return (
    <Chai>default
    <h1>data
    <p>>datalist
  )
}
export default App
```

Ek he element export kar saktya ho

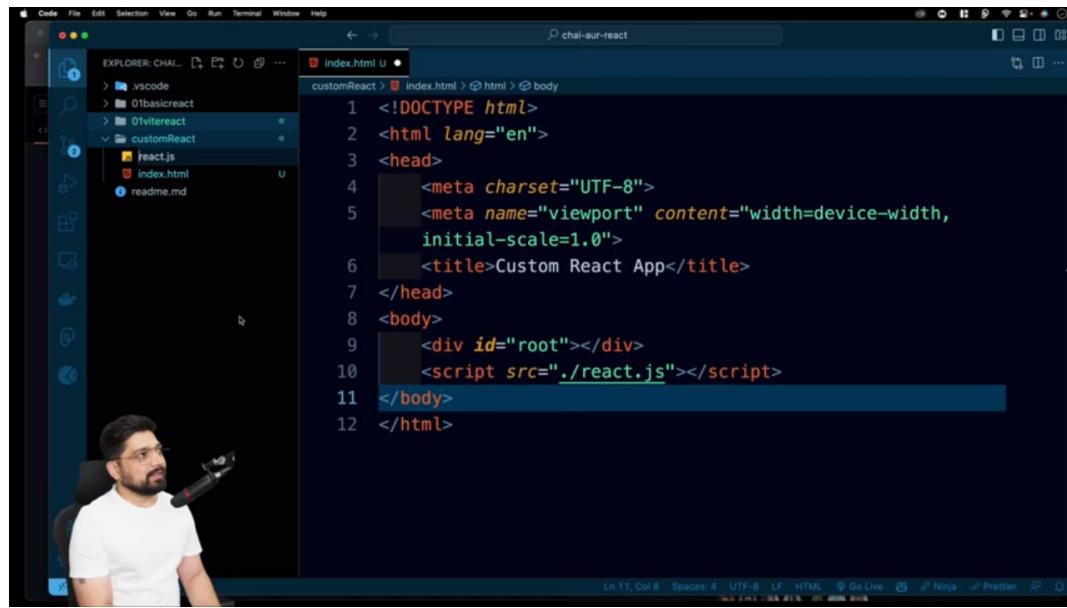
▷ 1:29:31



```
import Chai from "./chai"
function App() {
  return (
    <Chai/>
    <h1>chai aur react</h1>
    <p>test para</p>
  )
}
export default App
```

fragment return kar nay say issue resolve hojata hai

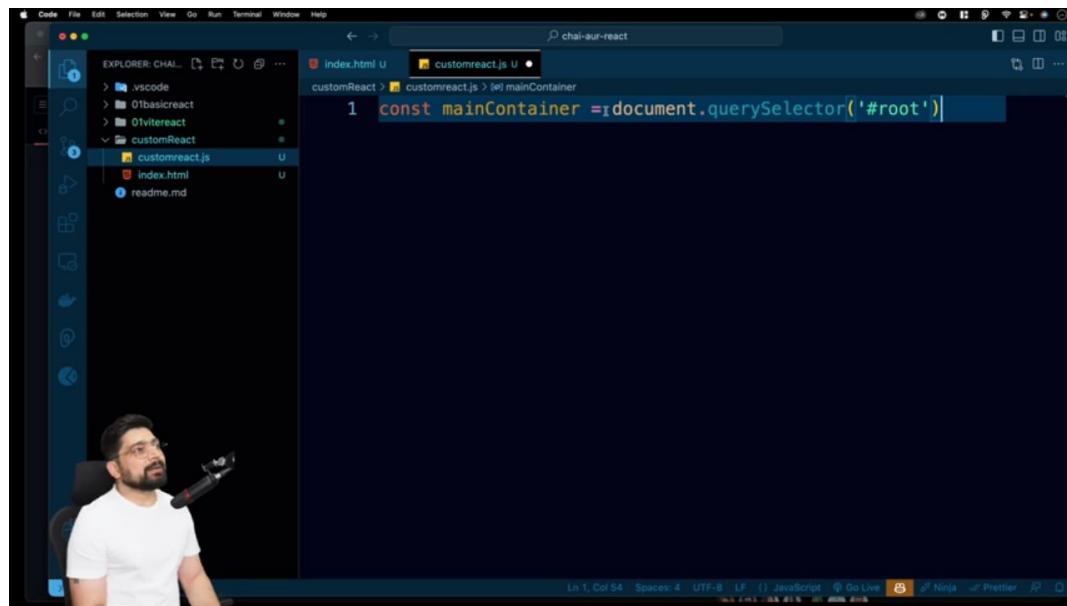
▷ 1:29:58



A screenshot of a video call interface. On the left, there is a video feed of a man with a beard and glasses, wearing a white t-shirt, sitting in a chair and speaking into a microphone. On the right, there is a code editor window titled "chai-aur-react". The editor shows the file "index.html" with the following content:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Custom React App</title>
  </head>
  <body>
    <div id="root"></div>
    <script src="./react.js"></script>
  </body>
</html>
```

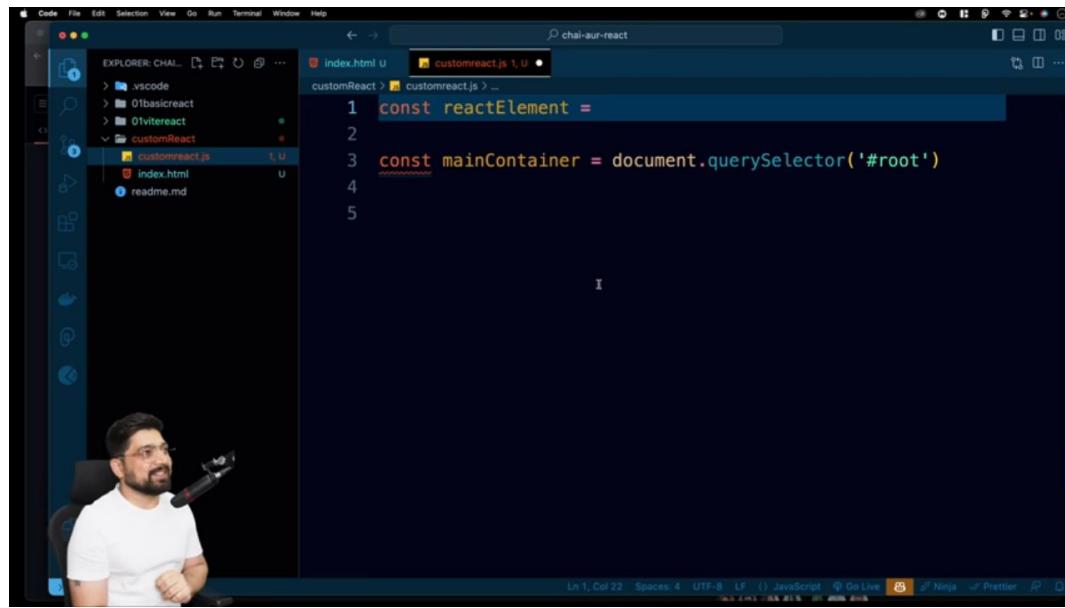
▶ 1:45:17



A screenshot of a video call interface. On the left, there is a video feed of the same man, now looking slightly upwards and to the right. On the right, there is a code editor window titled "chai-aur-react". The editor shows the file "customreact.js" with the following content:

```
const mainContainer = document.querySelector('#root')
```

▶ 1:46:07

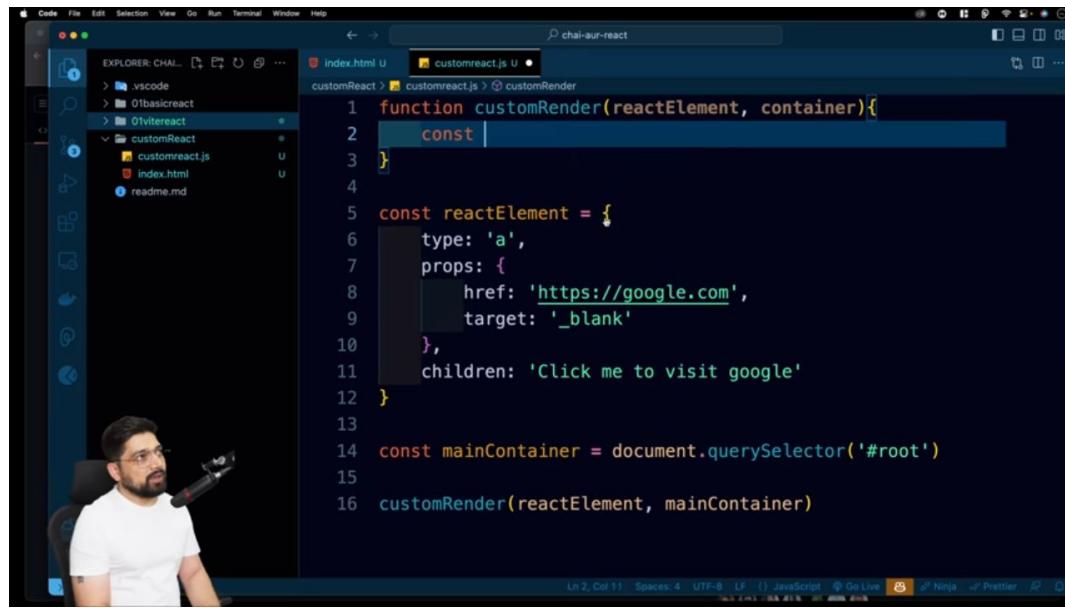


▷ 1:47:25

```
const reactElement = {  
    type: 'a',  
    props: {  
        href: 'https://google.com',  
        target: '_blank'  
    },  
    children: 'Click me to visit google'  
}  
  
const mainContainer = document.querySelector('#root')
```

The status bar at the bottom indicates 'Ln 8, Col 2 (114 selected)' and 'JavaScript'.

▷ 1:49:59

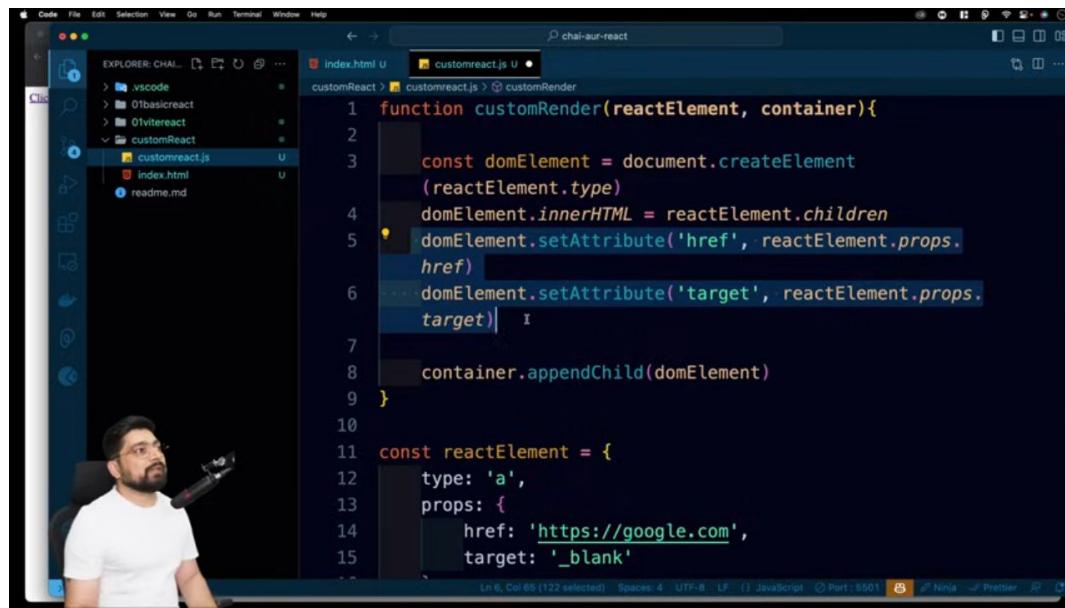


A screenshot of a video player interface. On the left, there's a video frame showing a man with a beard and glasses, wearing a white t-shirt, sitting in front of a computer monitor. The monitor displays the Visual Studio Code (VS Code) interface. In the VS Code editor, the file 'customreact.js' is open, showing the following code:

```
1 function customRender(reactElement, container){  
2     const |  
3 }  
  
5 const reactElement = {  
6     type: 'a',  
7     props: {  
8         href: 'https://google.com',  
9         target: '_blank'  
10    },  
11    children: 'Click me to visit google'  
12 }  
13  
14 const mainContainer = document.querySelector('#root')  
15  
16 customRender(reactElement, mainContainer)
```

The status bar at the bottom of the VS Code window shows: Line 2, Col 11, Spaces: 4, UTF-8, LF, JavaScript, Go Live, ⚡ Ninja, Prettier.

▶ 1:50:12

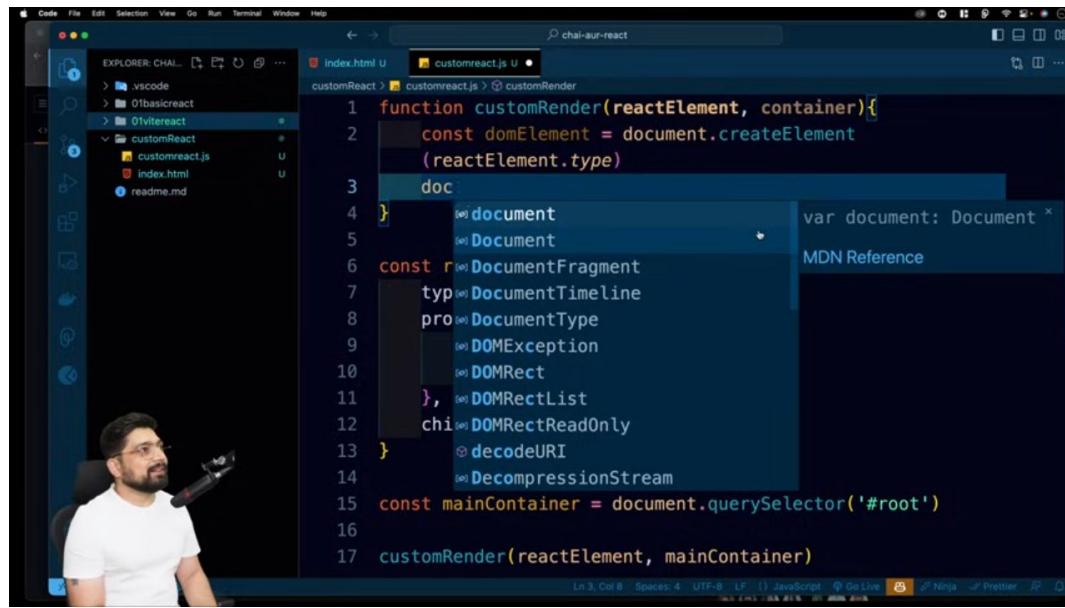


A screenshot of a video player interface, similar to the one above. It shows the same man speaking in front of the VS Code interface. The code in 'customreact.js' has been updated to show the implementation of the 'customRender' function:

```
1 function customRender(reactElement, container){  
2     const domElement = document.createElement  
3         (reactElement.type)  
4     domElement.innerHTML = reactElement.children  
5     domElement.setAttribute('href', reactElement.props.  
6         href)  
7     domElement.setAttribute('target', reactElement.props.  
8         target)|  
9     container.appendChild(domElement)  
10 }  
11  
12 const reactElement = {  
13     type: 'a',  
14     props: {  
15         href: 'https://google.com',  
16         target: '_blank'
```

The status bar at the bottom of the VS Code window shows: Line 6, Col 65 (122 selected), Spaces: 4, UTF-8, LF, JavaScript, Port: 5501, ⚡ Ninja, Prettier.

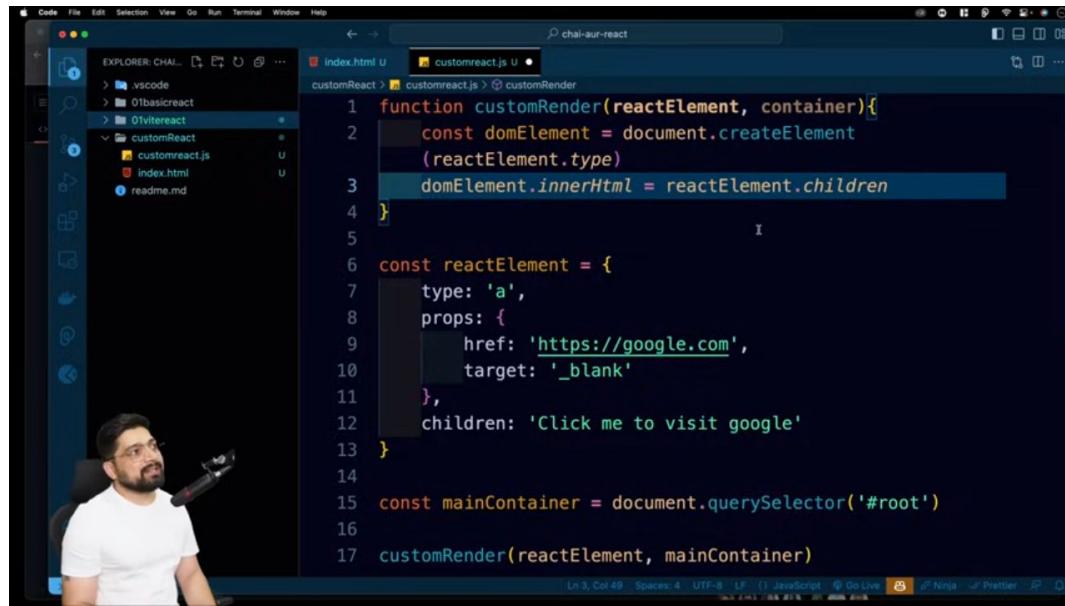
▶ 1:50:16



A screenshot of a video call interface. On the left, a video feed shows a man with a beard and glasses, wearing a white t-shirt, sitting in a chair and speaking into a microphone. On the right, a code editor window is open in a dark-themed IDE. The file being edited is 'customreact.js' under a 'customReact' folder. The code implements a custom render function that creates a new DOM element based on the type of the React element passed to it. A tooltip from the IDE's documentation system is visible, providing information about the 'document' object.

```
function customRender(reactElement, container){  
  const domElement = document.createElement  
    (reactElement.type)  
  doc  
}  
const r DocumentFragment  
typ DocumentTimeline  
pro DocumentType  
DOMException  
DOMRect  
}, DOMRectList  
chi DOMRectReadOnly  
@decodeURI  
@DecompressionStream  
const mainContainer = document.querySelector('#root')  
customRender(reactElement, mainContainer)
```

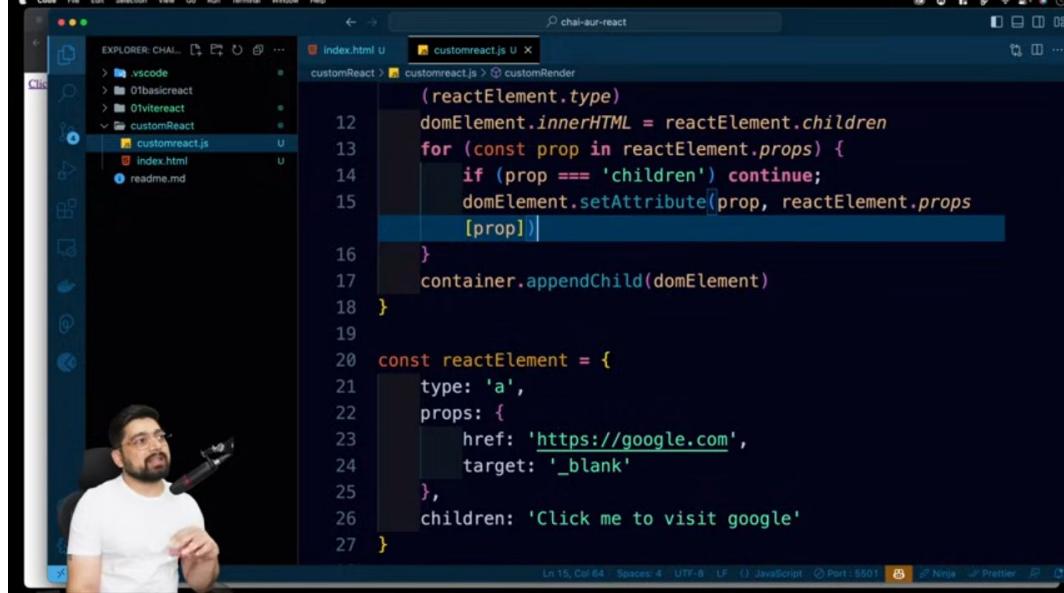
▶ 1:52:35



A screenshot of a video call interface. On the left, a video feed shows the same man as before, now looking slightly to the side. On the right, the code editor window shows the completed 'customRender' function. The final line of code, 'domElement.innerHTML = reactElement.children', is highlighted. The rest of the code defines a 'reactElement' object with properties like 'type' (set to 'a'), 'props' (containing 'href' and 'target' properties), and 'children' (a string). The code editor interface includes various status bars at the bottom.

```
function customRender(reactElement, container){  
  const domElement = document.createElement  
    (reactElement.type)  
  domElement.innerHTML = reactElement.children  
}  
  
const reactElement = {  
  type: 'a',  
  props: {  
    href: 'https://google.com',  
    target: '_blank'  
  },  
  children: 'Click me to visit google'  
}  
const mainContainer = document.querySelector('#root')  
customRender(reactElement, mainContainer)
```

▶ 1:53:07

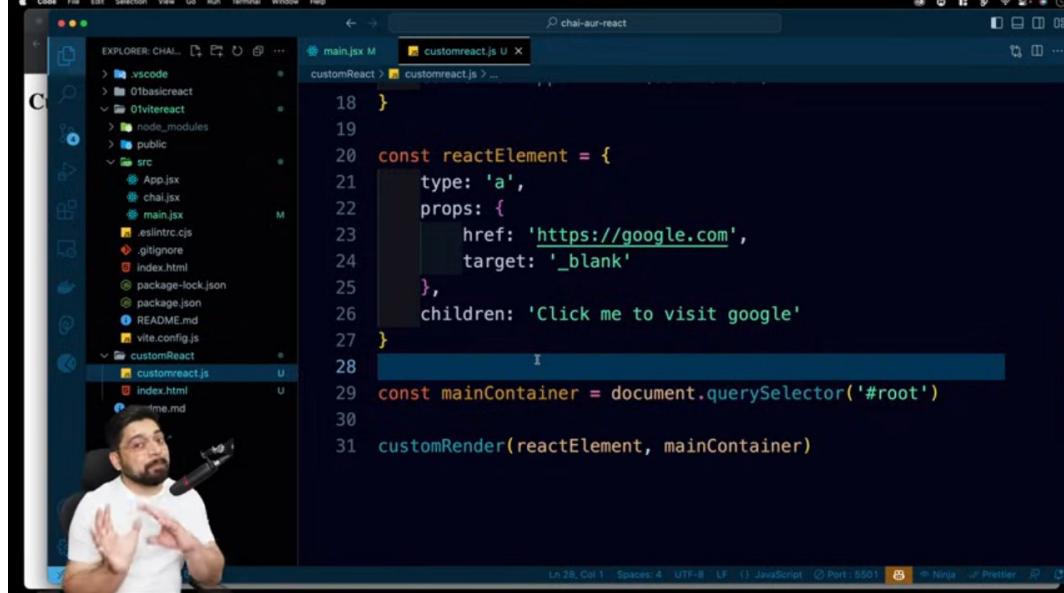


```
(reactElement.type)
domElement.innerHTML = reactElement.children
for (const prop in reactElement.props) {
  if (prop === 'children') continue;
  domElement.setAttribute(prop, reactElement.props[prop])
}
container.appendChild(domElement)

}

const reactElement = {
  type: 'a',
  props: {
    href: 'https://google.com',
    target: '_blank'
  },
  children: 'Click me to visit google'
}
```

▶ 1:55:32



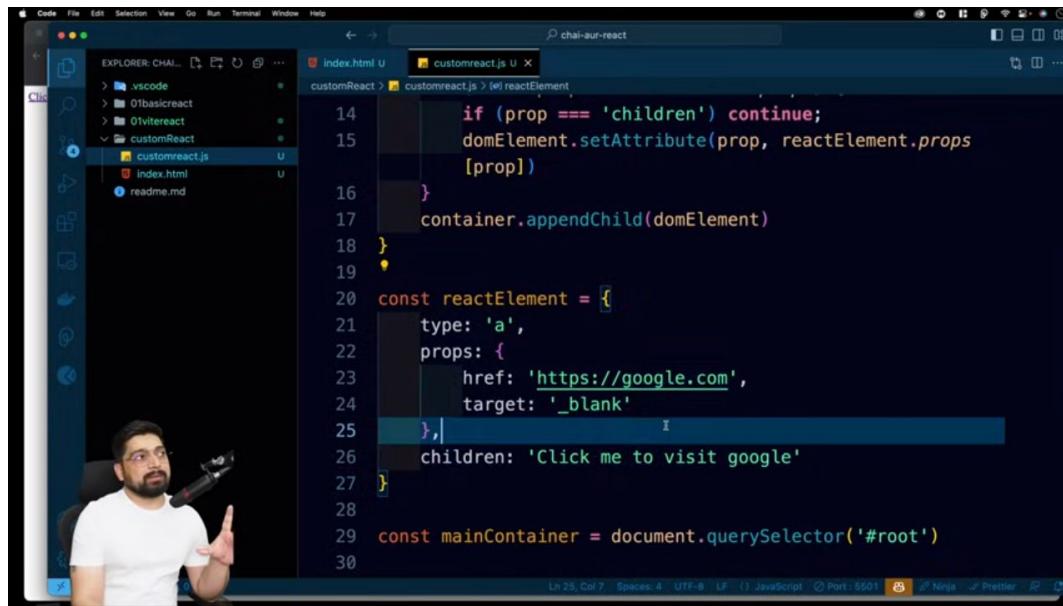
```

}

const reactElement = {
  type: 'a',
  props: {
    href: 'https://google.com',
    target: '_blank'
  },
  children: 'Click me to visit google'
}

const mainContainer = document.querySelector('#root')
customRender(reactElement, mainContainer)
```

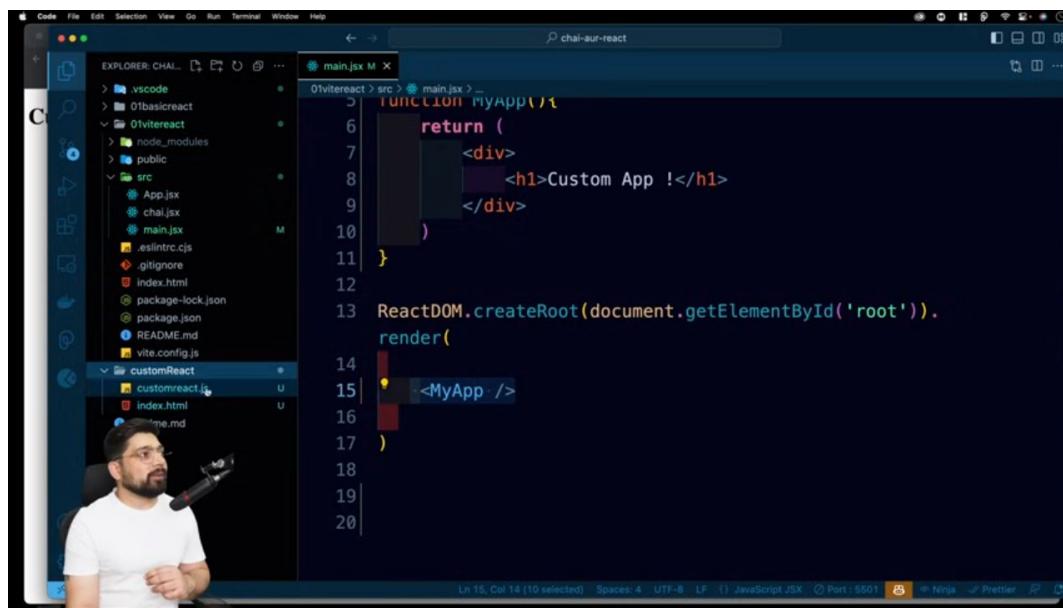
▶ 1:59:44



A screenshot of the Visual Studio Code interface. The title bar shows "chai-aur-react". The left sidebar has a video player icon and a timestamp "2:00:15". The main editor area displays a file named "customreact.js" with the following code:

```
14 if (prop === 'children') continue;
15 domElement.setAttribute(prop, reactElement.props
16 [prop])
17 }
18 }
19
20 const reactElement = {
21   type: 'a',
22   props: {
23     href: 'https://google.com',
24     target: '_blank'
25   },
26   children: 'Click me to visit google'
27 }
28
29 const mainContainer = document.querySelector('#root')
30
```

The code defines a custom React element ("a") with a href of "https://google.com" and a target of "\_blank". It also includes a click handler with the text "Click me to visit google".



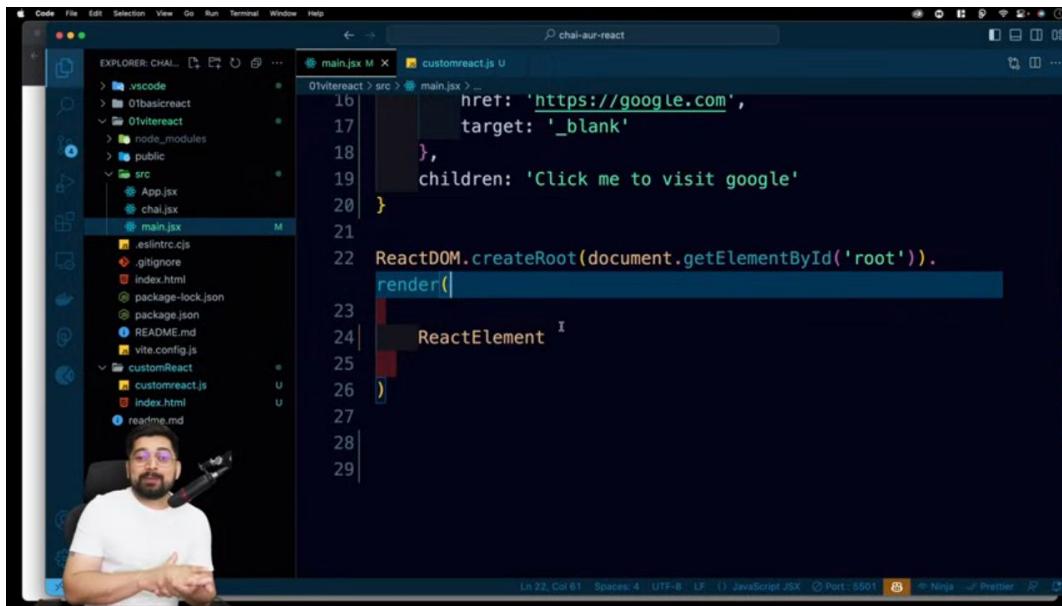
A screenshot of the Visual Studio Code interface. The title bar shows "chai-aur-react". The left sidebar has a video player icon and a timestamp "2:00:48". The main editor area displays a file named "main.jsx" with the following code:

```
5 function MyApp() {
6   return (
7     <div>
8       <h1>Custom App !</h1>
9     </div>
10   )
11 }
12
13 ReactDOM.createRoot(document.getElementById('root')).render(
14   <MyApp />
15 )
16
17
18
19
20
```

The code defines a functional component named "MyApp" which returns a single 

element containing a 

# element with the text "Custom App !". It then uses ReactDOM.createRoot to render this component at the root of the application.

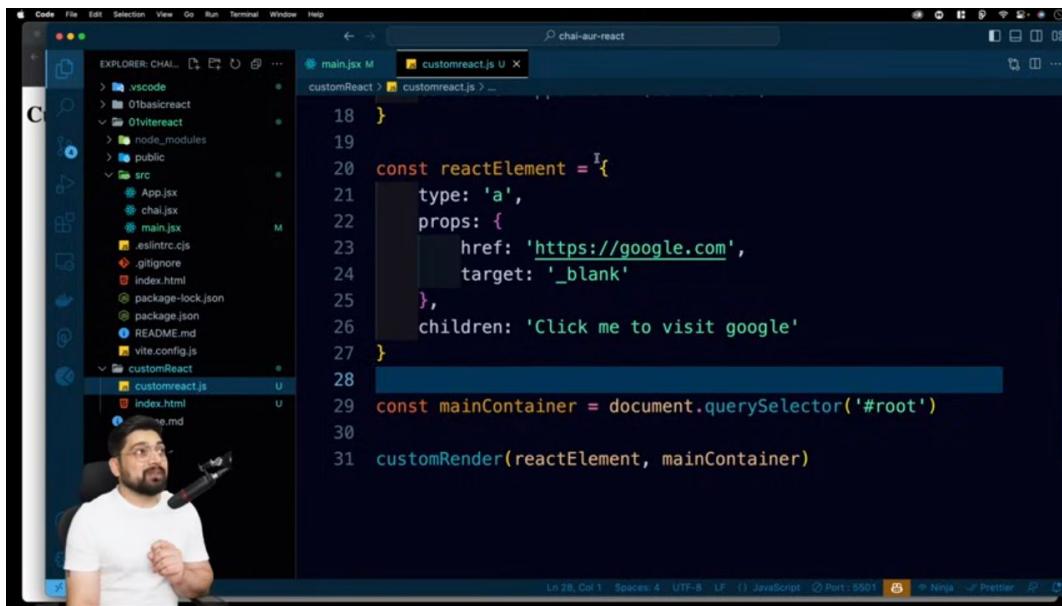


A screenshot of a video call interface. On the left, there's a video feed of a man with a beard and glasses wearing a white t-shirt. He is gesturing with his hands while speaking. To his right is a code editor window titled "chai-aur-react". The editor shows a file named "main.jsx" with the following code:

```
16 href: 'https://google.com',
17 target: '_blank'
18 },
19 children: 'Click me to visit google'
20 }

22 ReactDOM.createRoot(document.getElementById('root')).  
render(  
23   ReactElement  
24 )  
25  
26 )  
27  
28  
29
```

▶ 2:02:30



A screenshot of a video call interface. On the left, there's a video feed of the same man. He is gesturing with his hands while speaking. To his right is a code editor window titled "chai-aur-react". The editor shows a file named "main.jsx" with the following code:

```
18 }

19

20 const reactElement = {
21   type: 'a',
22   props: {
23     href: 'https://google.com',
24     target: '_blank'
25   },
26   children: 'Click me to visit google'
27 }

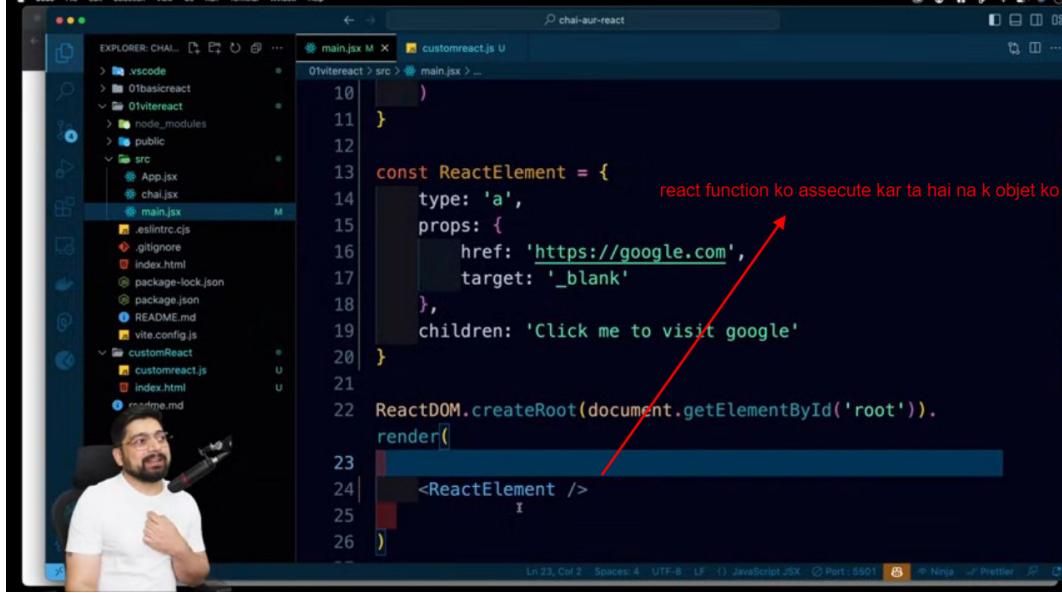
28 const mainContainer = document.querySelector('#root')

29 customRender(reactElement, mainContainer)
```

▶ 2:03:04

&lt;App /&gt; ka tage kahn say aey rah hai ya aey rah hai  
bundler say jo is

▶ 2:03:30



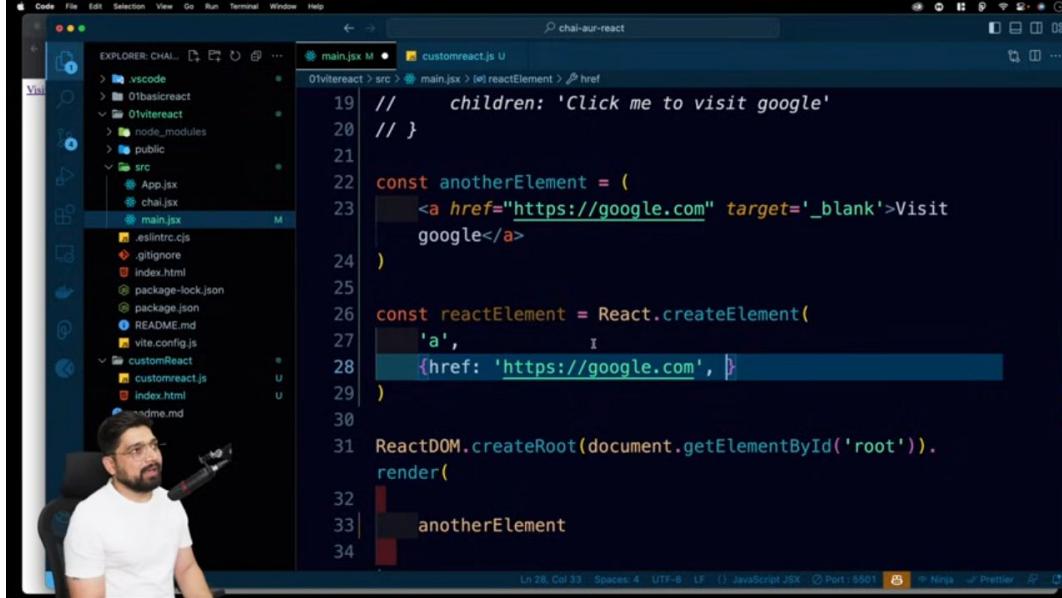
Code File Edit Selection View Go Run Terminal Window Help

EXPLORER: CHAI... 0vitereact main.jsx M customreact.js U

```
10 )
11 }
12
13 const ReactElement = {
14   type: 'a',           react function ko asseute kar ta hai na k objet ko
15   props: {
16     href: 'https://google.com',
17     target: '_blank'
18   },
19   children: 'Click me to visit google'
20 }
21
22 ReactDOM.createRoot(document.getElementById('root')).render(
23   <ReactElement />
24 )
25
26 )
```

Ln 23, Col 2 Spaces: 4 UTF-8 LF ( JavaScript JSX Port: 5501 ⚡ Ninja ⌂ Prettier ⌂ ⌂

▶ 2:06:08



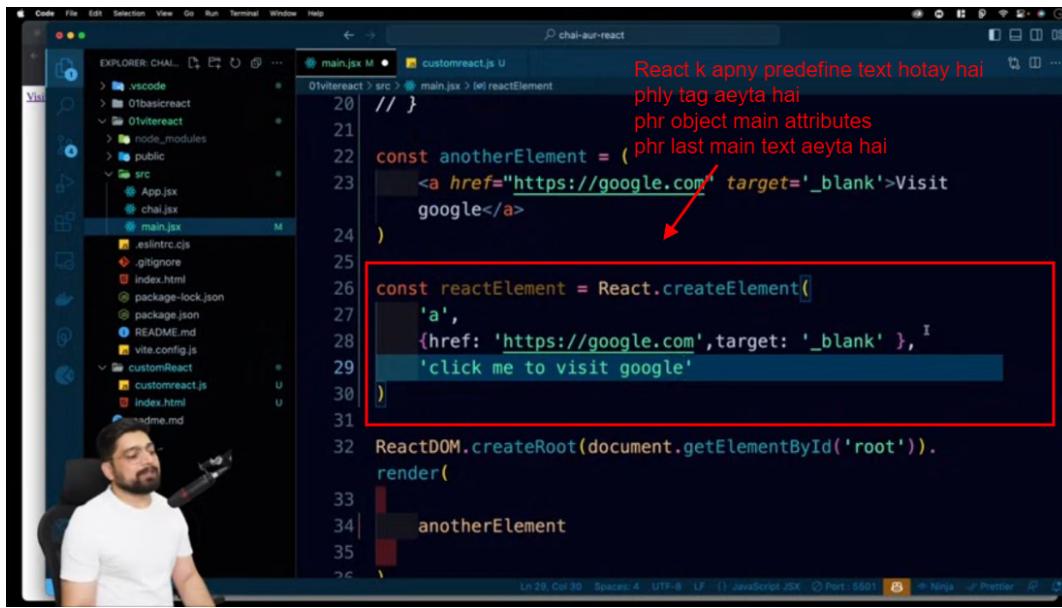
Code File Edit Selection View Go Run Terminal Window Help

EXPLORER: CHAI... 0vitereact main.jsx M customreact.js U

```
19 //   children: 'Click me to visit google'
20 //
21
22 const anotherElement = (
23   <a href="https://google.com" target='_blank'>Visit
24   google</a>
25 )
26
27 const reactElement = React.createElement(
28   'a',           I
29   {href: 'https://google.com', })
30
31 ReactDOM.createRoot(document.getElementById('root')).render(
32   anotherElement
33 )
34
```

Ln 28, Col 33 Spaces: 4 UTF-8 LF ( JavaScript JSX Port: 5501 ⚡ Ninja ⌂ Prettier ⌂ ⌂

▶ 2:11:10



React k apny predefine text hotay hai  
phly tag aeyta hai  
phr object main attributes  
phr last main text aeyta hai

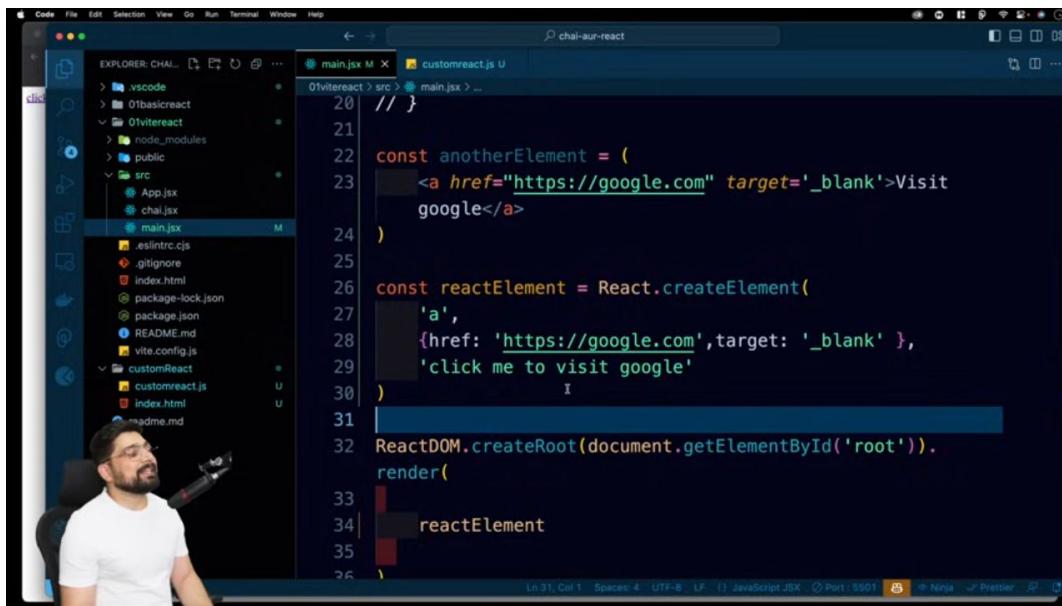
```
// }

const anotherElement = (
  <a href="https://google.com" target='_blank'>Visit
  google</a>
)

const reactElement = React.createElement(
  'a',
  {href: 'https://google.com', target: '_blank' },
  'click me to visit google'
)

ReactDOM.createRoot(document.getElementById('root')).render(
  anotherElement
)
```

▶ 2:13:59



```
// }

const anotherElement = (
  <a href="https://google.com" target='_blank'>Visit
  google</a>
)

const reactElement = React.createElement(
  'a',
  {href: 'https://google.com', target: '_blank' },
  'click me to visit google'
)

ReactDOM.createRoot(document.getElementById('root')).render(
  reactElement
)
```

▶ 2:14:48

A screenshot of a video conference interface. On the right is a video feed of a man with a beard, wearing a white t-shirt, sitting at a desk with a microphone. On the left is a code editor window titled "chai-aur-react". The file "App.jsx" is open, showing the following JSX code:

```
function App() {
  const username = "chai aur react"
  return (
    <>
    <Chai/>
    <h1>chai aur react</h1>
    <p>test para</p>
  )
}

export default App
```

▷ 2:15:24

javascript kasay inject ho gi jsx main?

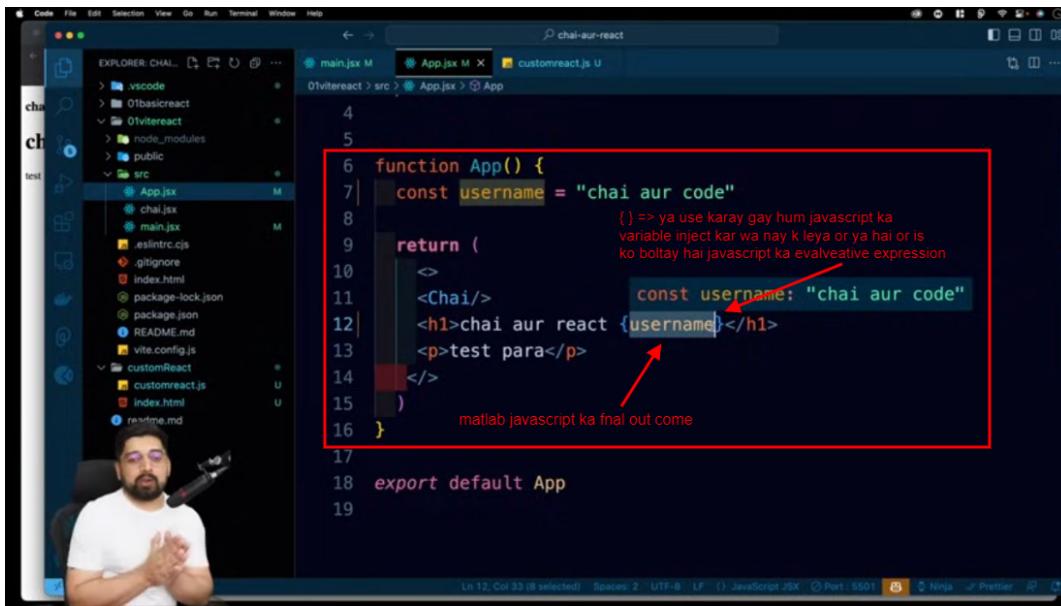
▷ 2:15:50

A screenshot of a video conference interface. On the right is a video feed of the same man with a beard, wearing a white t-shirt. On the left is a code editor window titled "chai-aur-react". The file "App.jsx" is open, showing the following JSX code with a cursor on the line "const username: "chai aur code"":

```
function App() {
  const username = "chai aur code"
  return (
    <>
    <Chai/> const username: "chai aur code"
    <h1>chai aur react {username}</h1>
    <p>test para</p>
  )
}

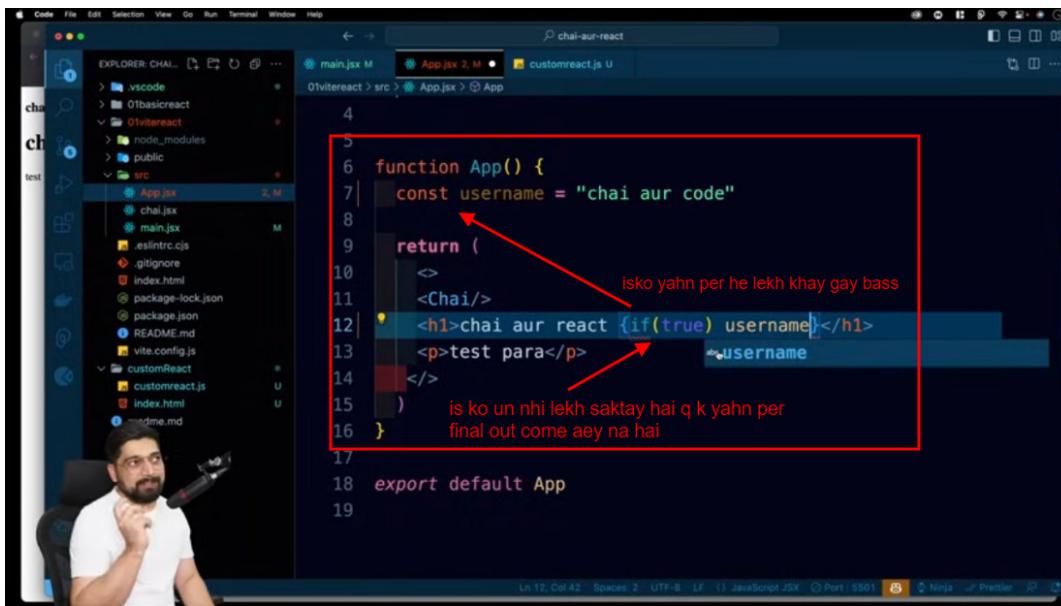
export default App
```

▷ 2:16:32



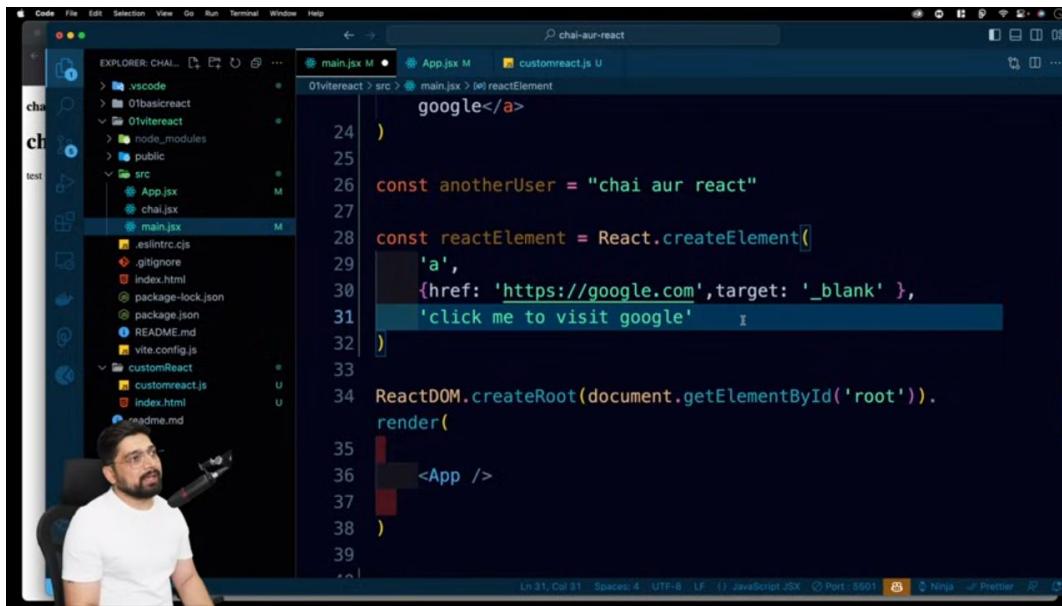
```
4
5
6 function App() {
7   const username = "chai aur code"
8   // => ya use karay gay hum javascript ka
9   // variable inject kar wa nay k leya or ya hai or is
10  // ko boltay hai javascript ka evaluative expression
11  <>
12  <Chai>          const username: "chai aur code"
13  <h1>chai aur react {username}</h1>
14  <p>test para</p>
15  </>
16 }           matlab javascript ka final out come
17
18 export default App
19
```

▶ 2:16:45



```
4
5
6 function App() {
7   const username = "chai aur code"
8   return (
9     <>
10    <Chai>
11    <h1>chai aur react {if(true) username}</h1>
12    <p>test para</p>
13    </>
14  )
15 }           isko yahan per he lekh khay gay bass
16 }           is ko un nhi lekh sakay hai q k yahan per
17                                         final out come aey na hai
18
19 export default App
20
```

▶ 2:17:10



```
const anotherUser = "chai aur react"

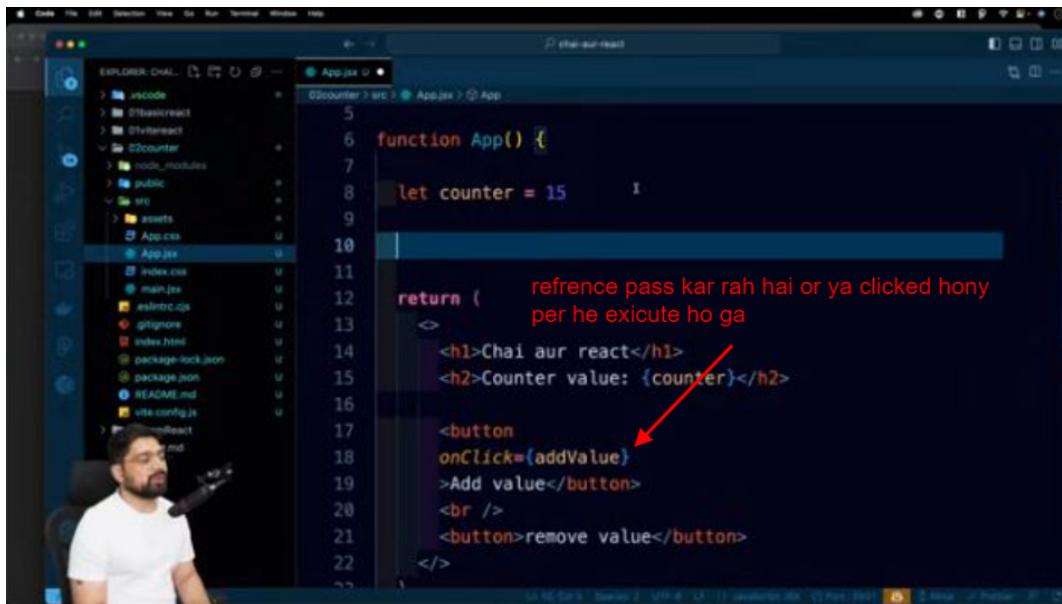
const reactElement = React.createElement(
  'a',
  {href: 'https://google.com', target: '_blank'},
  'click me to visit google'
)

ReactDOM.createRoot(document.getElementById('root')).render(
  <App />
)
```

▶ 2:18:22

babale transpilaor behande the scen work karta haii react main

▶ 2:23:28



```
function App() {
  let counter = 15

  return (
    <div>
      <h1>Chai aur react</h1>
      <h2>Counter value: {counter}</h2>

      <button onClick={addValue}>Add value</button>
      <br />
      <button>remove value</button>
    </div>
  )
}

function addValue() {
  counter++
}
```

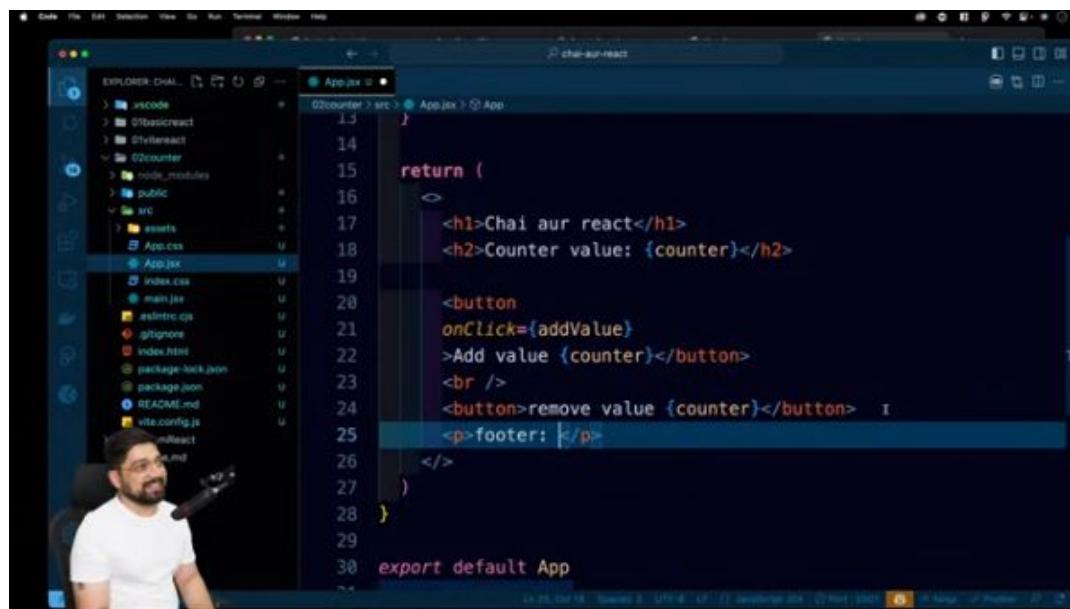
▶ 2:34:27

react main problem aeyti hai UI updataion main

▶ 2:38:23

react, react karti hai ui ki updatation pee

▷ 2:38:48



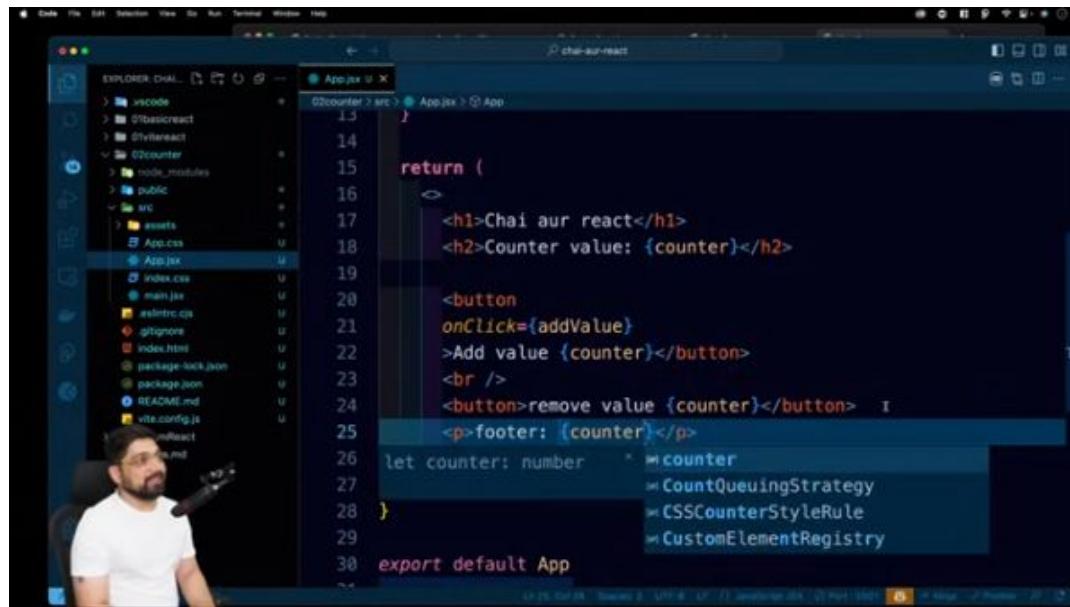
A screenshot of a video call interface. On the left, there is a video feed of a man with a beard, wearing a white t-shirt, sitting in front of a blackboard. On the right, there is a screenshot of a code editor in VS Code. The editor shows a file named 'App.jsx' with the following code:

```
return (
  <h1>Chai aur react</h1>
  <h2>Counter value: {counter}</h2>

  <button
    onClick={addValue}
  >Add value {counter}</button>
  <br />
  <button>remove value {counter}</button>
  <br />
  <p>Footer: {counter}</p>
)

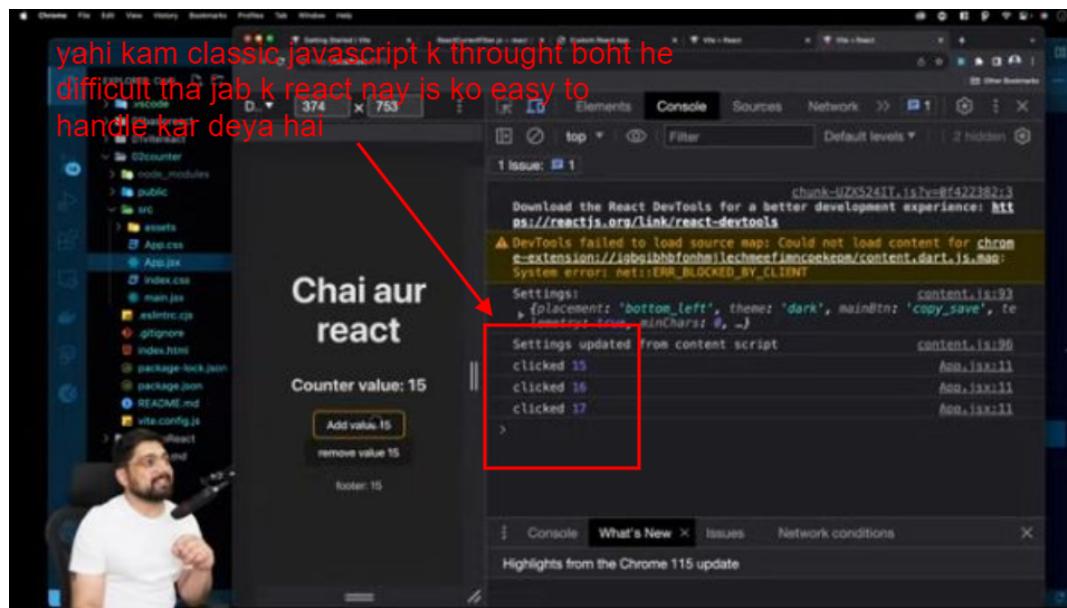
export default App
```

▷ 2:39:21



A screenshot of a video call interface, similar to the previous one. On the left is a video feed of the same man. On the right is a screenshot of the VS Code code editor. The code editor shows the same 'App.jsx' file, but now with code completion dropdowns visible over the 'counter' variable in the footer paragraph. The dropdown includes options like 'CountQueueingStrategy', 'CSSCounterStyleRule', and 'CustomElementRegistry'. The code completion dropdown is highlighted with a blue background.

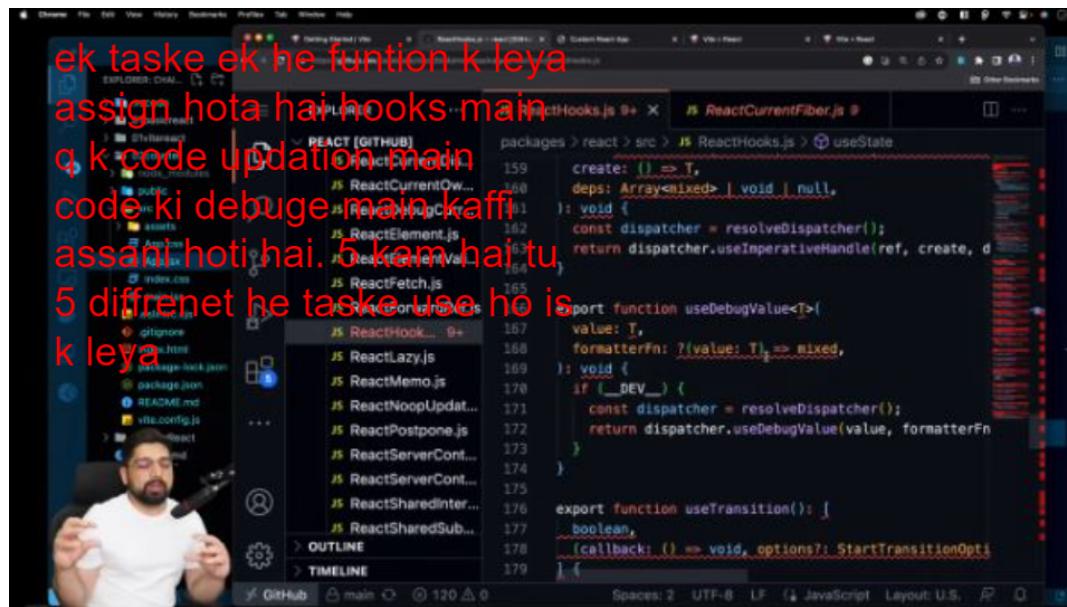
▷ 2:39:25



▷ 2:39:38

hooks k through he UI main data updation hoti hai react mia

▷ 2:40:25



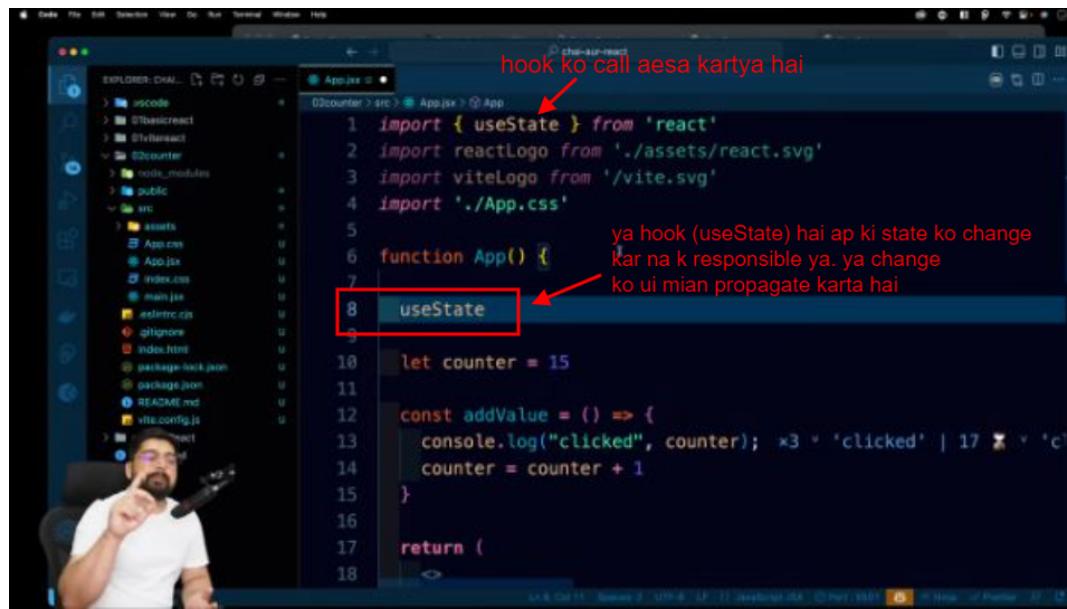
▷ 2:41:52

react/packages/react/src/ReactHooks.js at main · facebook/react

▷ 2:42:00

## hook useState ki waja say aey ta hai

▷ 2:43:00

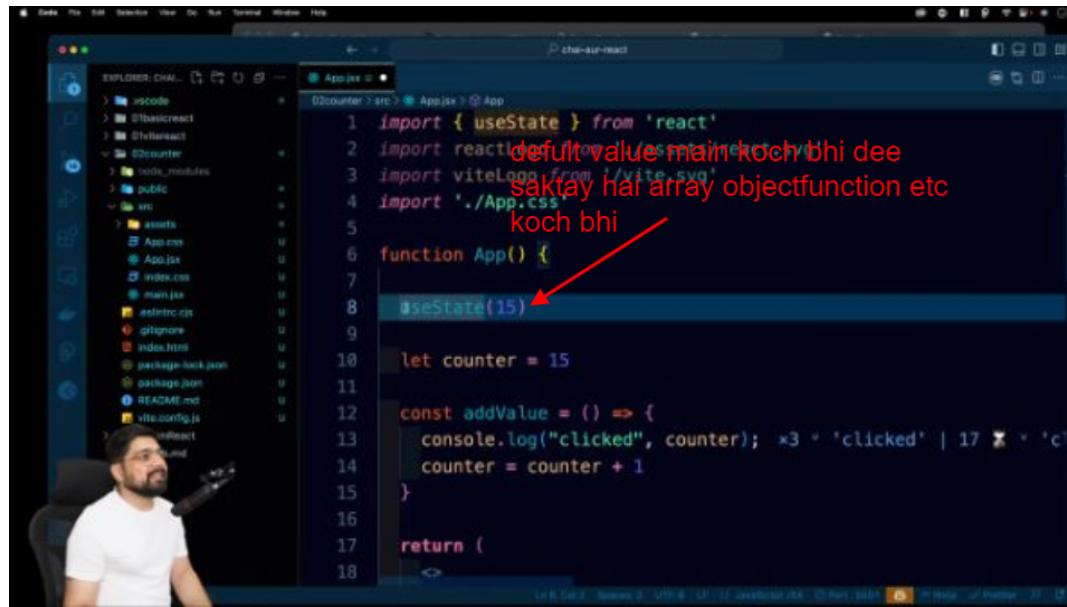


```
hook ko call aesa kartya hai
ya hook (useState) hai ap ki state ko change
Kar na k responsible ya. ya change
ko ui mian propagate karta hai

import { useState } from 'react'
function App() {
  let counter = 15
  const addValue = () => {
    console.log("clicked", counter);
    counter = counter + 1
  }
  return (
    <div>
      <h1>{counter}</h1>
      <p>button</p>
      <button onClick={addValue}>+</button>
    </div>
  )
}

export default App
```

▷ 2:44:00

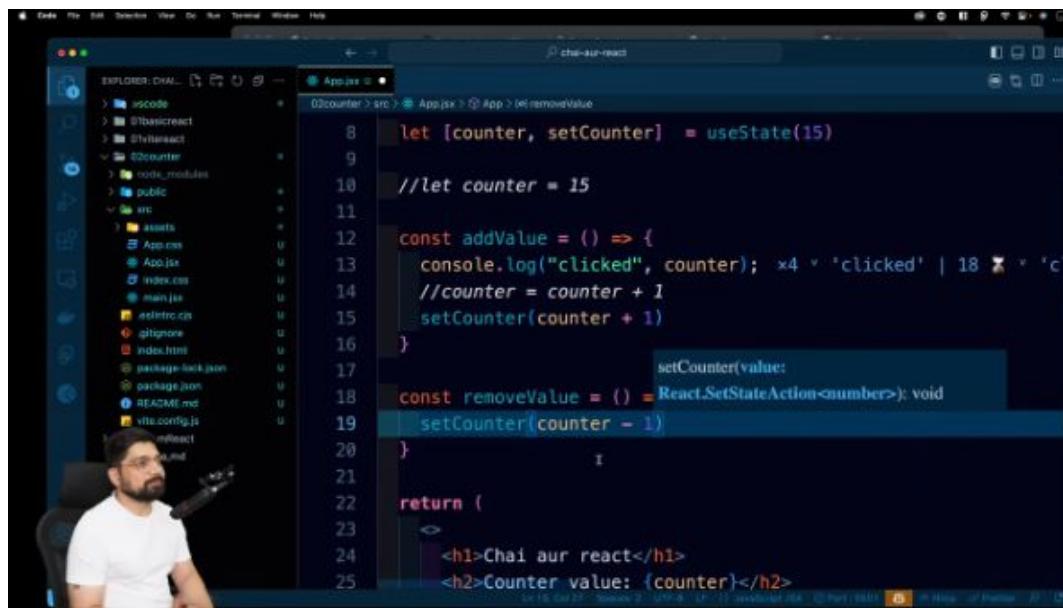


```
default value main koch bhi dee
sak�ay hai array objectfunction etc
koch bhi

import { useState } from 'react'
function App() {
  let counter = 15
  const addValue = () => {
    console.log("clicked", counter);
    counter = counter + 1
  }
  return (
    <div>
      <h1>{counter}</h1>
      <p>button</p>
      <button onClick={addValue}>+</button>
    </div>
  )
}

export default App
```

▷ 2:45:12



A screenshot of a video call interface. On the left, there is a video feed of a man with a beard and glasses, wearing a white t-shirt, sitting in a black office chair. He is looking towards the camera. On the right, a window of a code editor (VS Code) is displayed. The file being edited is named 'App.js' and contains React code for a counter application. The code includes useState, useEffect, and functional state updates. A tooltip is visible over the 'setCounter' function, showing its type as 'React.Dispatch<React.SetStateAction<number>>'. The code editor has a dark theme with syntax highlighting.

```
let [counter, setCounter] = useState(15)
//let counter = 15
const addValue = () => {
  console.log("clicked", counter);
  //counter = counter + 1
  setCounter(counter + 1)
}
const removeValue = () => React.Dispatch<React.SetStateAction<number>>: void
  setCounter(counter - 1)
}

return (
  <h1>Chai aur react</h1>
  <h2>Counter value: {counter}</h2>
)
```

▶ 2:51:46



▶ 2:56:29

jasa browser k pass ek dom hot hai  
wasa he ya bhi dom **create** karta hai  
ta k browser k dom k sat comparission  
karwa sakay or unhi chezo ko update kary  
hai jo actual ain update hoi hai. taken browser  
pora dom remove karta hai or wapess say page  
ko repaint karta hai is ko reload bhi boltyai hai

```
import React from 'react'
import ReactDOM from 'react-dom/client'
import App from './App.jsx'
import './index.css'

ReactDOM.createRoot(document.getElementById('root'))
render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
)

```

▶ 2:59:00

virtual dom ko pora ka pora trase kar sakty ho tree like structure  
main or jo jo values change hoi hai us ko dom say nekal kar  
wapess laga do

▶ 2:59:48