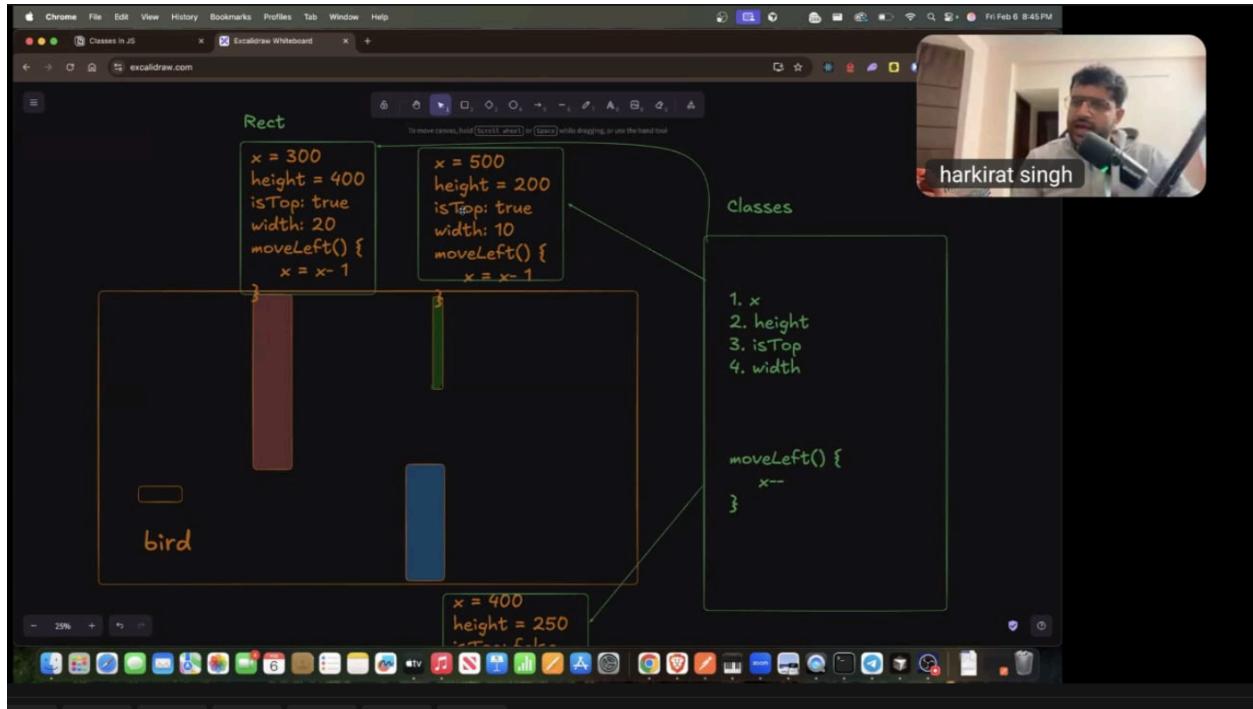


## Classes in JS

<https://petal-estimate-4e9.notion.site/Classes-in-JS-2ff7dfd107358002937ccae0fcc7d985>



## Inheritance in classes

<https://petal-estimate-4e9.notion.site/Inheritance-in-classes-2ff7dfd10735800597b5ed9ef3eec7d2>

A screenshot of a video conference interface. On the right, a video feed of Harkirat Singh is visible, holding a microphone. The main area shows a whiteboard with three classes defined:

```
class Rectangle {
    constructor(width, height, color) {
        this.width = width;
        this.height = height;
        this.color = color;
    }

    area() {
        return this.width * this.height;
    }

    perimeter() {
        return 2 * this.width + 2 * this.height;
    }

    paint() {
        console.log("painting with color " + this.color);
    }
}

class Circle {
    constructor(radius, color) {
        this.radius = radius;
        this.color = color;
    }

    area() {
        return 3.14 * this.radius * this.radius;
    }

    perimeter() {
        return 2 * 3.14 * this.radius;
    }

    paint() {
        console.log("painting with color " + this.color);
    }
}

class Square {
    constructor(side, color) {
        this.side = side;
        this.color = color;
    }

    area() {
        return this.side * this.side;
    }

    perimeter() {
        return 4 * this.side;
    }

    paint() {
        console.log("painting with color " + this.color);
    }
}
```

The code is annotated with arrows pointing from the inheritance lines in the classes back to a separate box containing the definition of the `Shape` class.

```
class Shape {
    constructor(color) {
        this.color = color;
    }

    paint() {
        console.log("painting with color " + this.color);
    }
}
```

singh's screen

A screenshot of a video conference interface. On the right, a video feed of Harkirat Singh is visible, holding a microphone. The main area shows a terminal window with the file `index.js` open. The code is identical to the one shown on the whiteboard, illustrating the inheritance structure.

```
JS index.js > ⌂ Shape > ⌂ constructor
  You, now | 1 author (You)
  1 > class Rectangle extends Shape {-
  14 }
  15
  You, now | 1 author (You)
  16 > class Circle extends Shape {-
  17 }
  18
  You, now | 1 author (You)
  19 > class Square extends Shape {-
  20 }
  21
  You, now | 1 author (You)
  22 class Shape {
  23     constructor(color) {
  24         this.color = color;
  25     }
  26
  27     paint() {
  28         console.log("painting with color " + this.color);
  29     }
  30 }
```

## Some more classes

### Date

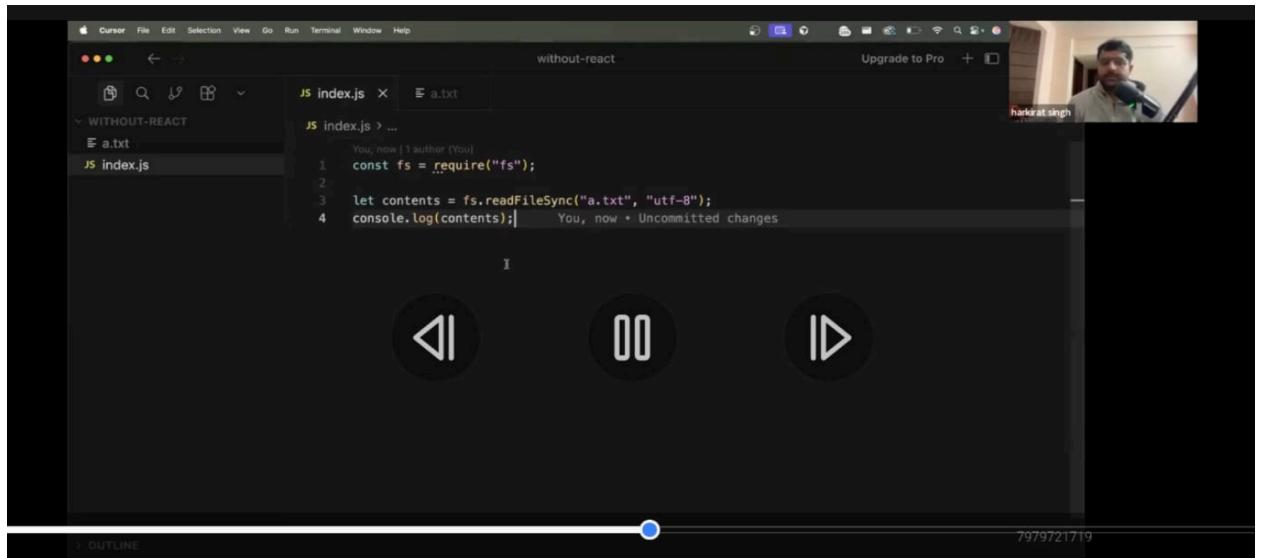
```
const now = new Date(); // Current date and time
console.log(now.toISOString()); // Outputs the date in ISO format
```

### Maps

```
const map = new Map();
map.set('name', 'Alice');
map.set('age', 30);
console.log(map.get('name'));
```

## Promise class

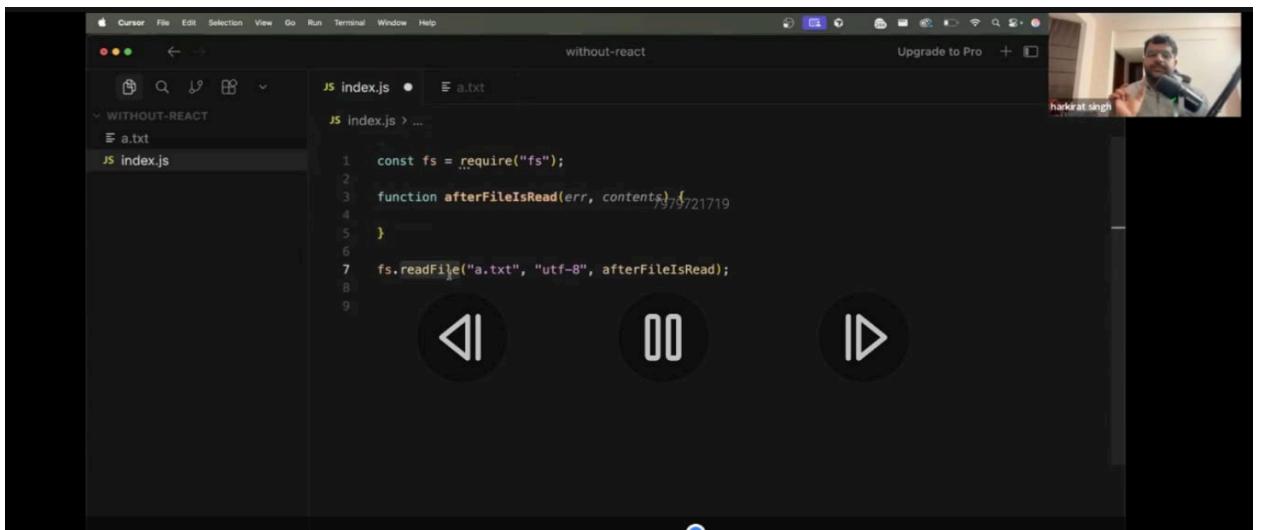
### 1. Synchronous code



A screenshot of a dark-themed code editor, likely VS Code, showing a file named `index.js`. The code reads a file named `a.txt` using `fs.readFileSync` and logs its contents to the console. A video call interface is visible in the top right corner, showing a person named Harkarat Singh.

```
You, now | 1 author (You)
1 const fs = require("fs");
2
3 let contents = fs.readFileSync("a.txt", "utf-8");
4 console.log(contents);| You, now + Uncommitted changes
```

## 2. Asynchronous code



```
Cursor File Edit Selection View Go Run Terminal Window Help
without-react
Upgrade to Pro + ⌘
JS index.js • a.txt
WITHOUT-REACT
a.txt
JS index.js > ...
1 const fs = require("fs");
2
3 function afterFileIsRead(err, contents) {
4
5 }
6
7 fs.readFile("a.txt", "utf-8", afterFileIsRead);
8
9
```

### Using a function that returns a promise

Ignore the function definition of `setTimeoutPromisified` for now

```
function setTimeoutPromisified(ms) {
  return new Promise(resolve => setTimeout(resolve, ms));
}

function callback() {
  console.log("3 seconds have passed");
}

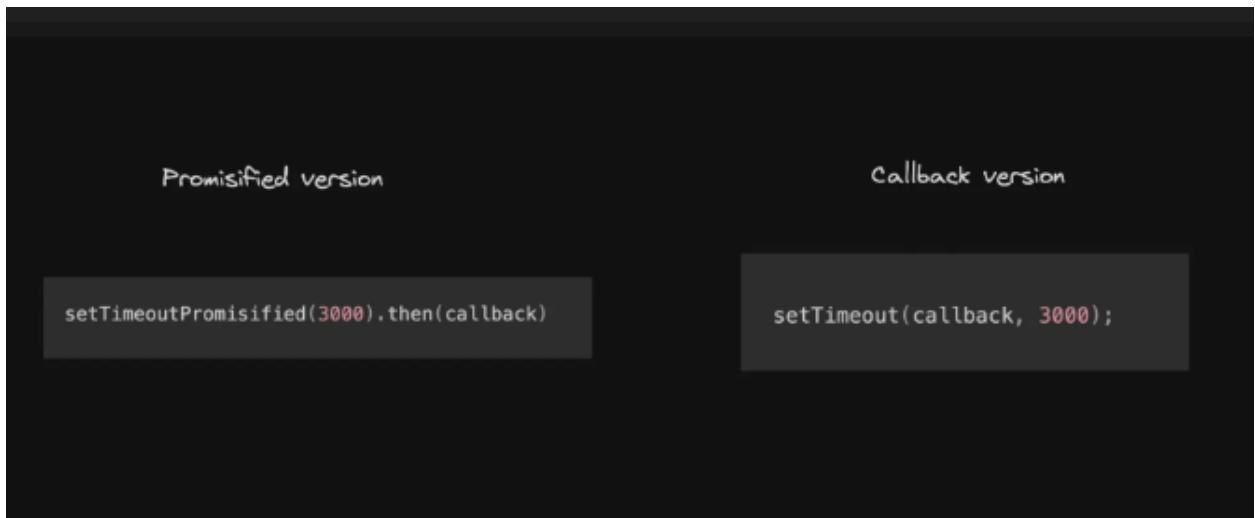
setTimeoutPromisified(3000).then(callback)
```

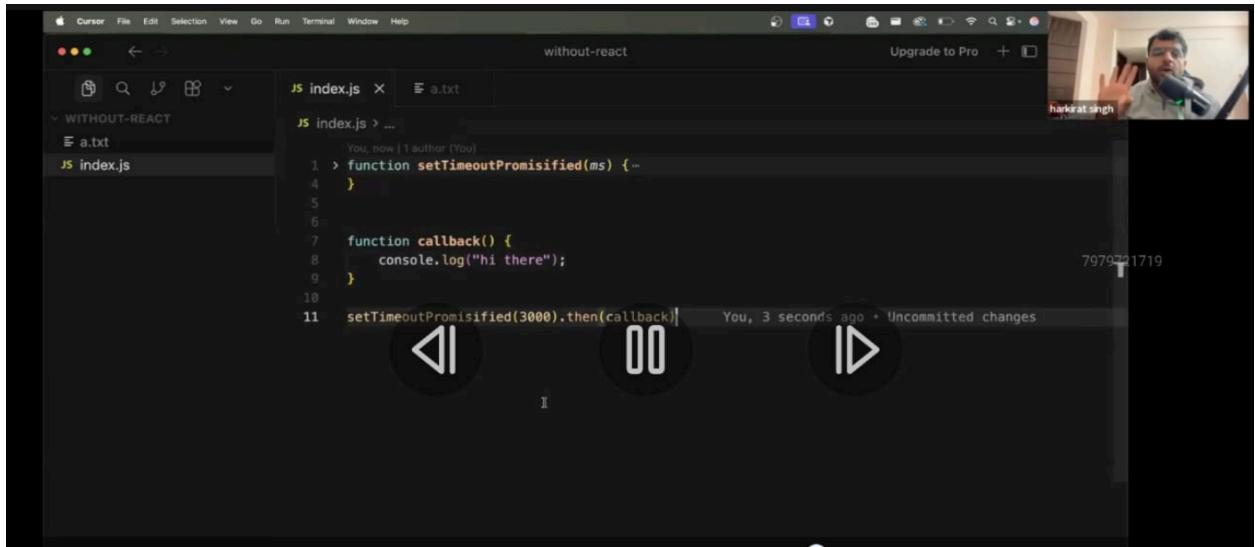
We use below now also

A screenshot of a Mac OS X desktop environment. At the top, there's a menu bar with options like Cursor, File, Edit, Selection, View, Go, Run, Terminal, Window, and Help. Below the menu bar is a toolbar with icons for file operations. The main window title is "without-react". Inside the window, there's a terminal pane showing the following code in index.js:

```
JS index.js x a.txt
JS index.js > ↗ callback
You, now | 1 author (You)
1
2
3
4  function callback() {
5    console.log("hi there");      You, 8 seconds ago * Uncommitted changes
6  }
7
8  setTimeout(callback, 5 * 1000);

The terminal also shows the output of the command: "hi there". Below the terminal are playback controls (rewind, stop, forward). At the bottom, there's a status bar with the number 7979721719. A video call overlay of a person named Harkirat Singh is visible in the top right corner.
```

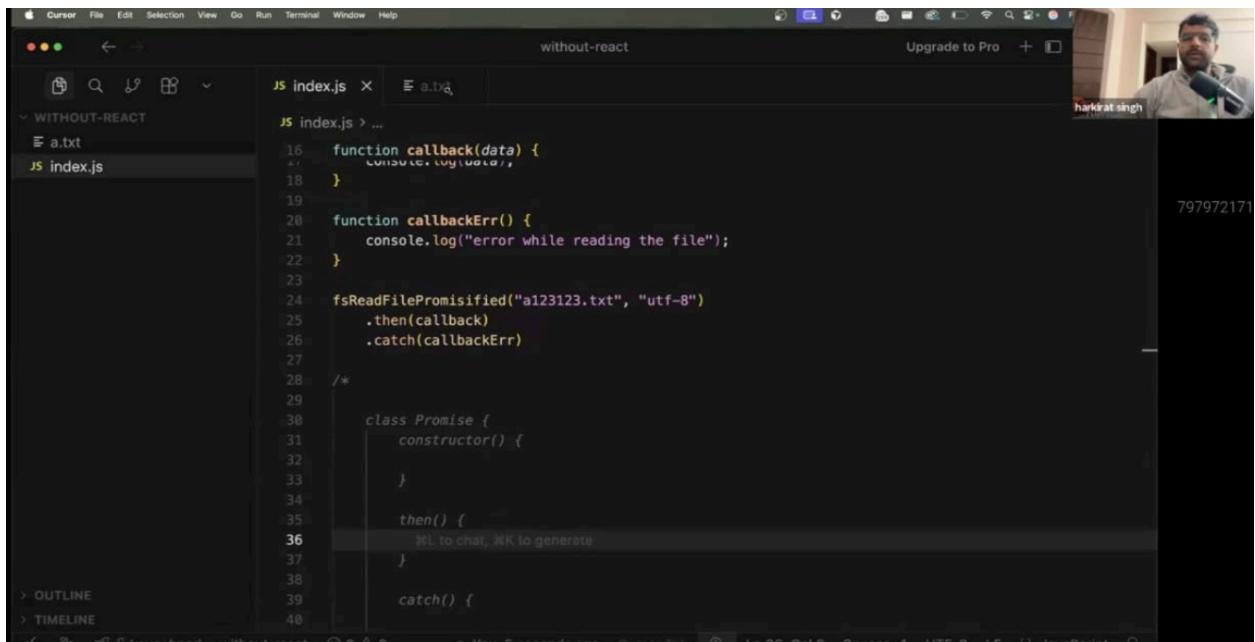




A screenshot of a video conference interface. On the right, a video feed of a man with a microphone is visible. The main window shows a code editor with two tabs: 'index.js' and 'a.txt'. The 'index.js' tab contains the following code:

```
You, now | 1 author (You)
1 > function setTimeoutPromisified(ms) { ...
2   }
3
4   function callback() {
5     console.log("hi there");
6   }
7
8   setTimeoutPromisified(3000).then(callback)
9
10
11 setTimeoutPromisified(3000).then(callback)| You, 3 seconds ago * Uncommitted changes
```

At bottom on the line there is promise class



A screenshot of a video conference interface. On the right, a video feed of a man with a microphone is visible. The main window shows a code editor with two tabs: 'index.js' and 'a.txt'. The 'index.js' tab contains the following code, with line 36 highlighted:

```
You, now | 1 author (You)
16   function callback(data) {
17     console.log(data);
18   }
19
20   function callbackErr() {
21     console.log("error while reading the file");
22   }
23
24   fsReadFilePromisified("a123123.txt", "utf-8")
25     .then(callback)
26     .catch(callbackErr)
27
28 /**
29
30   class Promise {
31     constructor() {
32
33     }
34
35     then() {
36       //L to chat, #K to generate
37     }
38
39     catch() {
40
41   }
```

A screenshot of a Mac OS X desktop environment. At the top, there's a menu bar with options like Cursor, File, Edit, Selection, View, Go, Run, Terminal, Window, Help. Below the menu bar is a toolbar with icons for file operations. The main window is a terminal application titled "without-react". It shows a file tree on the left with "a.txt" and "index.js". The "index.js" file content is:

```
You, 7 seconds ago | ↑ author '(You)
1 const fs = require("fs");
2
3 function callback(err, data) {
4     if (err) {
5         console.log("error while reading the file");
6     } else {
7         console.log(data);
8     }
9 }
10
11 fs.readFile("a.txt", "utf-8", callback);
```

The terminal output shows:

```
You, 5 seconds ago * Uncommitted changes
You, 7 seconds ago | ↑ author '(You)
You, 5 seconds ago | ↑ author '(You)
You, 5 seconds ago | ↑ author '(You)
You, 5 seconds ago | ↑ author '(You)
```

At the bottom of the terminal window, there's a status bar with information like "zsh", "Ln 11, Col 41", "Spaces: 4", "UTF-8", and "JavaScript". A video call interface is overlaid on the right side of the screen, showing a man with a microphone and a camera view.

```
function fsReadFilePromisified(filePath, encoding) {
    return new Promise((resolve, reject) => {
        fs.readFile(filePath, encoding, (err, data) => {
            if (err) {
                reject(err)
            } else {
                resolve(data);
            }
        })
    })
}
```

```
without-react
JS index.js x a.txt
WITHOUT-REACT
a.txt
JS index.js > ↗ callback
3 function fsReadFilePromisified(filePath, encoding) {
4     return new Promise((resolve, reject) => {
5         fs.readFile(filePath, encoding, (err, data) => {
6             if (err) {
7                 reject(err);
8             } else {
9                 resolve(data);
10            }
11        })
12    }
13}
14
15
16
17 function callback(data) { You, now + Uncommitted changes
18     console.log(data);
19 }
20
21 function callbackErr() {
22     console.log("error while reading the file");
23 }
24
25 fsReadFilePromisified("a.txt", "utf-8")
26     .then(callback)
27     .catch(callbackErr)
```

## .then, .catch, pending concept in promise

### Callback hell:

<https://petal-estimate-4e9.notion.site/Callback-hell-2ff7dfd10735801bbcb9c9b21dd4845d>

```
without-react
JS index.js x a.txt
WITHOUT-REACT
a.txt
JS index.js > ↗ setTimeout() callback > ↗ setTimeout() callback
You, now | 1 author (You)
1 setTimeout(function() {
2     console.log("hi there");
3     setTimeout(function() {
4         console.log("hello")
5     }, 3000);
6 }, 1000);|
```

The screenshot shows a code editor window titled "without-react". The main pane displays a file named "index.js" with the following content:

```
You, now \| author (You)
1  function setTimeoutPromisified(ms) {
2      return new Promise(resolve => setTimeout(resolve, ms));
3  }
4
5  setTimeoutPromisified(1000).then(function() {
6      console.log("hi");
7      return setTimeoutPromisified(3000)
8  }).then(function() {
9      console.log("hello");
10     return setTimeoutPromisified(5000)
11 }).then(function() {
12     console.log("Hello");
13 })
```

The status bar at the bottom right shows the commit ID: 7979721719.