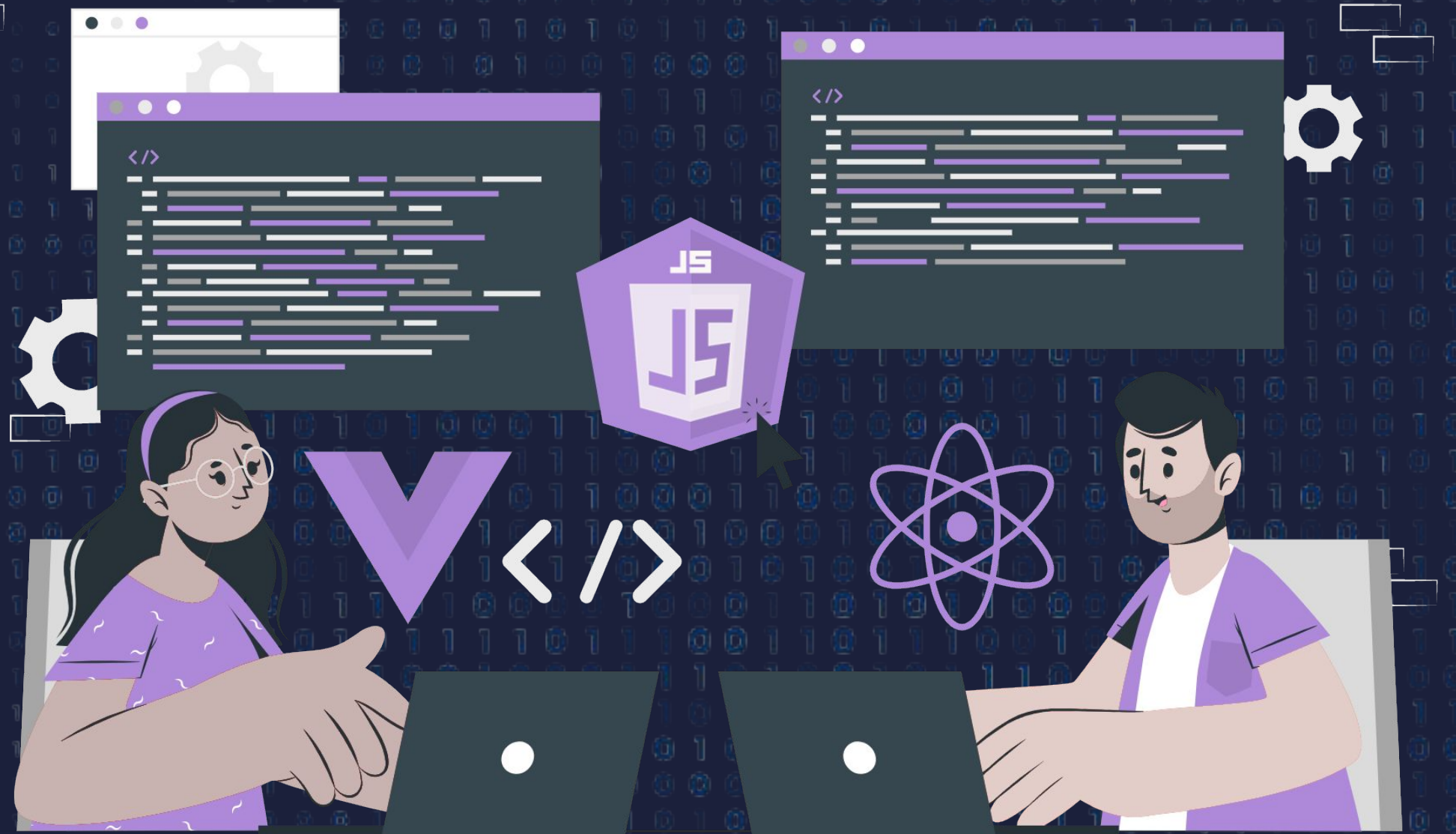# Lecture CheckList

1. Introduction to Arrow functions.

2. Arrow Function Syntax.

3. Why should developers use arrow functions?

Introduction to Arrow functions.

In the early 2010s javascript was rapidly evolving and developers were looking for ways to write more concise and expressive code. While working with functions, there was a need for some function declaration with concise syntax and extended capabilities.

At the time, function expressions were a common way to define functions, but they had some limitations. Around this time, a proposal for a new type of function called an "arrow function" was introduced. This proposal aimed to address some of the limitations of function expressions by introducing a new shorthand syntax for defining functions.

The proposal was first included in the 6th edition of ECMAScript (ECMAScript 2015), which was released in June 2015. This marked the official introduction of arrow functions to the JavaScript language.

Since then, arrow functions have become a popular feature of the language and are widely used by developers.

Arrow Function Syntax.

Arrow functions, also known as "fat arrow" functions, were introduced in JavaScript as a shorthand for writing function expressions.

Arrow functions can be written in different ways depending upon the number of parameters it accepts and the operation it performs.

```javascript
// 1. One parameter, and a single return statement
const square = (x) => x * x;

// 2. Multiple parameter, and a single return expression
const sumOftwoNumbers = (x, y) => x + y;

// 3. Multiple statements in function expression
const sum = (x, y) => {
  console.log(`Adding ${x} and ${y}`);
  return x + y;
};

// 4. Returning an object
const sumAndDifference = (x, y) => ({ sum: x + y, difference: x - y });
```

Arrow Function Syntax.

```
// CALLING A FUNCTION

let output1 = square(5); // OUTPUT: 25
let output2 = sumOftwoNumbers(1, 2); // OUTPUT: 3
let output3 = sum(1, 2);
/*
OUTPUT: Adding 1 and 2
3
*/
let output4 = sumAndDifference(5, 3); // OUTPUT: { sum: 8, difference: 2 }
```

Features of Arrow Function Syntax

- Parentheses are optional in the case of a single parameter.

- Must use parentheses in case of multiple parameters.

- The return keyword is not required for a single return expression in the function body.

- The return keyword is required in case of multiple statements in the function.

- To return an object notation, wrap it with parentheses.

Why should developers use arrow functions?

There are several reasons why developers opt for using arrow functions. Some of them include:

- Arrow functions have very concise code which is ideal for single-line functions, and for situations where conciseness is important.

- Arrow functions do not use the return keyword. Arrow functions automatically return the value of the expression that follows the =>, this makes it more readable.

- The arrow function comes with "this", so don't have its own bindings to this, arguments, or super, and should not be used as methods. We will be looking into this in further lectures.

- Arrow functions are often used with array methods like map(), filter(), and reduce(), because they make the code more readable, especially when used in a chain of array methods. We will be looking at these methods in further lectures.