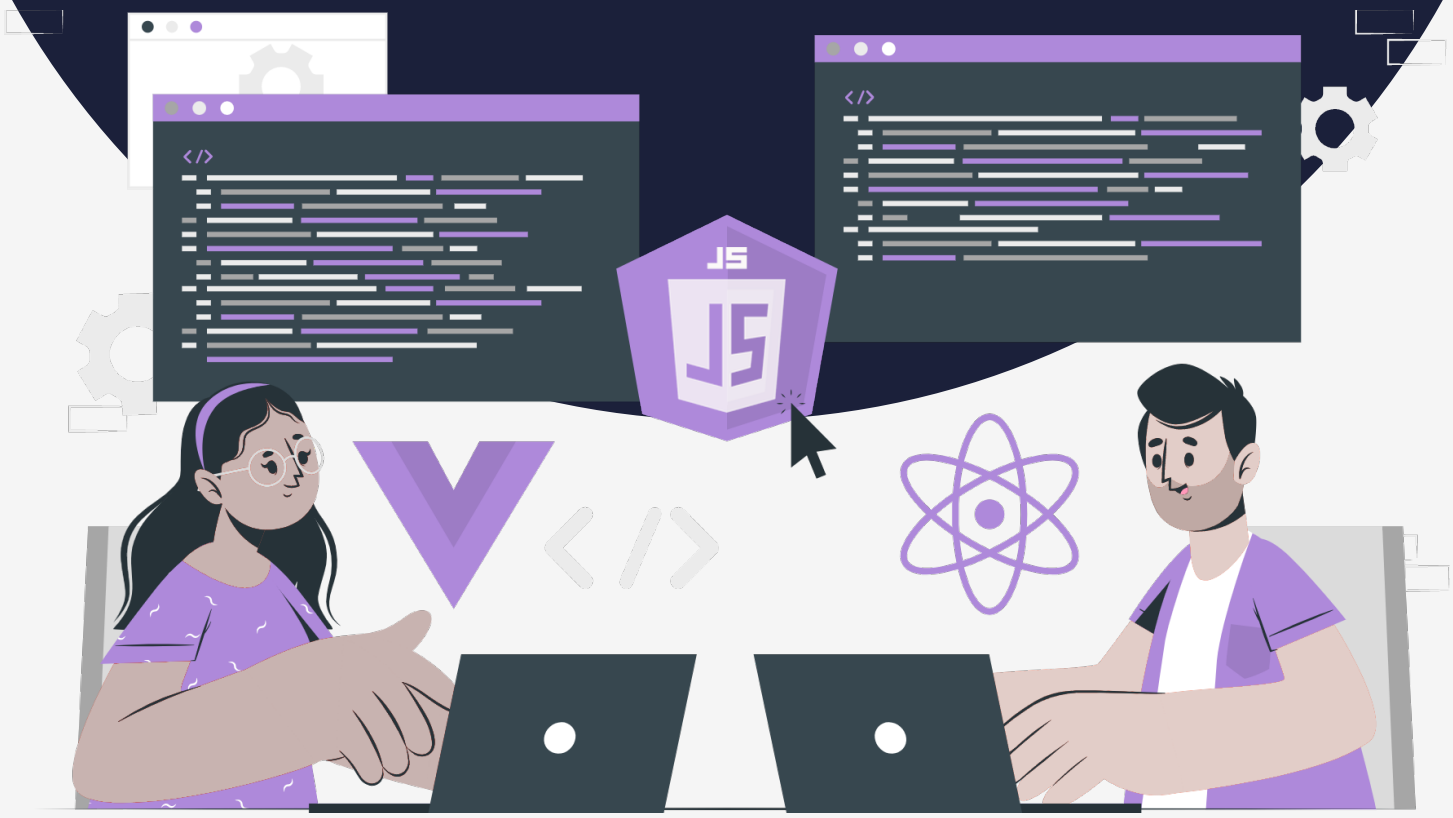
Lesson:

Destructuring Object



PW SKILLS

Topics Covered:

1. Introduction.
2. Destructuring object.
3. Random Accessing during object destructuring.
4. Object destructuring with default values.

In the previous lectures, we covered "Array Destructuring" which allows us to extract values from an array and assign them to variables in a concise way. In this lecture, we will focus on "Object Destructuring" which is a similar concept but applied to objects.

"Object Destructuring" allows us to extract properties from an object and assign them to variables. This can make our code more concise and readable. With "Object Destructuring", we can quickly access the properties of an object without having to write long chains of dot notation or bracket notation.

Destructuring Object.
Object destructuring makes our code more concise and readable by reducing the need for long chains of dot notation or bracket notation to access object properties. It can help prevent errors and bugs by allowing you to assign default values to variables in case the properties you're trying to extract are not defined. It also allows us to quickly extract only the properties you need from an object, which can improve performance and memory usage.

We can use key names when using "Object Destructuring" to extract properties from an object.

```
let student = {
  name: "Mithun S",
  course: "Full Stack Web Development",
  dashboardAccessGiven: true,
};

// Destructuring Object

let { name, course, dashboardAccessGiven } = student;
console.log(name);
console.log(course);
console.log(dashboardAccessGiven);

/*
OUTPUT:

Mithun S
Full Stack Web Development
true
*/
```

In the above code, we are declaring three variables (name, course, and dashboardAccessGiven) and then using object destructuring to assign the values of the corresponding properties from the student object to those variables.

After the destructuring is done, we are printing the values of those variables to the console, which will output the values of the corresponding properties from the student object.

Random Accessing during object destructuring.
Object destructuring also allows us to extract specific properties from an object and assign them to new variables using a shorthand syntax.

Using Random Property accessing, we can extract properties from an object based on their position or index, rather than their names. This technique is commonly used when dealing with objects that have a large number of properties.

```
let student = {
  name: "Mithun S",
  course: "Full Stack Web Development",
  dashboardAccessGiven: true,
};

// Random Accessing the properties: name & dashboardAccessGiven during destructuring.

let { name, dashboardAccessGiven } = student;
console.log(name);
console.log(dashboardAccessGiven);

/*
OUTPUT:
Mithun S
true
*/
```

Object destructuring with default values.

In situations where a property may not exist or may have a value of undefined, object destructuring with default values can be used to provide a fallback value.

This technique allows us to specify a default value for a variable if the value is not present in the object being destructured. Object destructuring with default values can be particularly useful in situations where it is necessary to handle optional properties, or when dealing with objects from external sources where the presence or absence of certain properties is not guaranteed.

```
let student = {
  name: "Mithun S",
  course: undefined,
  dashboardAccessGiven: true,
};

let {
  name,  dashboardAccessGiven,
  course = "Full Stack Web Development",
} = student;

console.log(name);
console.log(course);
console.log(dashboardAccessGiven);

/*
OUTPUT:
Mithun S
Full Stack Web Development
true
*/

let student = {
  name: "Mithun S",
  dashboardAccessGiven: true,
};

let {
  name,  dashboardAccessGiven,
  course = "Full Stack Web Development",
} = student;

console.log(name);
console.log(course);
console.log(dashboardAccessGiven);

/*
OUTPUT:
Mithun S
Full Stack Web Development
true
*/
```