

## Lab 2 Understanding

From these lessons, I understand how Vue uses data-driven logic to control both the behavior and appearance of an application. Vue allows the UI to automatically update whenever the underlying data changes, which helps keep everything consistent and reactive without manually manipulating the DOM.

Using v-for, I learned how to render lists from arrays stored in the app's data, such as product details and variant options. Each item in the list is generated dynamically, and the :key attribute plays an important role by giving Vue a way to uniquely identify each element. This improves performance and ensures Vue can properly track changes when items are updated or reordered.

I also learned how event handling works through v-on or the shorthand @. This makes it possible to respond to user actions like clicks and mouseovers. For example, clicking the "Add to Cart" button triggers a method that increases the cart value, while hovering over a product variant updates the displayed image. This interaction makes the app feel more engaging and intuitive for users.

Additionally, class and style binding allow the UI to visually react to data. Style binding lets me dynamically apply CSS styles, such as changing background colors based on variant data. Class binding allows CSS classes to be conditionally applied, like disabling and visually dimming a button when a product is out of stock. Overall, these concepts show how Vue ties together data, events, and styling to create a smooth and user-friendly experience.

Task of lesson 5 picture:

The screenshot shows a Vue.js application running in a browser. The code editor on the left contains the `index.html` and `main.js` files. The `index.html` file includes a template with a `v-for` loop for variants and a `v-for` loop for details. The `main.js` file defines a Vue instance with a data object containing a product named 'Socks' with an image, stock status, details, and variants (green and blue). The browser window on the right shows the final output: a blue pair of socks with a green logo, the title 'Socks', the status 'In Stock', and material details '50% cotton, 30% wool, 20% polyester'. Below the material details, it lists the color variants as 'green' and 'blue'.

Challenge of lesson 5 picture:

The screenshot shows a Vue.js application running in a browser. The code editor on the left contains the `index.html` and `main.js` files. The `index.html` file includes a template with a `v-bind:src` binding for the image and a `v-for` loop for details. The `main.js` file defines a Vue instance with a data object containing a product named 'Socks' with an image, stock status, details, and variants (green and blue). The browser window on the right shows the final output: a blue pair of socks with a green logo, the title 'Socks', the status 'In Stock', and material details '50% cotton, 30% wool, 20% polyester'. Below the material details, it lists the size variants as 'S', 'M', 'L', and 'XL'.

Task of lesson 6 picture:

index.html

```

1 <html>
2   <body>
3     <div id="app">
4       <div class="product-display">
5         <div class="product-container">
6           <div class="product-info">
7             <h1>{{ product }}</h1>
8             <p v-if="inStock">In Stock</p>
9             <p v-else>Out of Stock</p>
10            <ul>
11              <li v-for="detail in details">{{ detail }}</li>
12            </ul>
13            <div v-for="variant in variants" :key="variant.id" @mouseover="updateImage(variant.image)">
14              
15              <div>
16                <strong>{{ variant.color }}</strong>
17                {{ variant.id }}<br/>
18                {{ variant.materials }}<br/>
19                {{ variant.size }}<br/>
20                {{ variant.price }}<br/>
21                {{ variant.stock }}<br/>
22                {{ variant.description }}<br/>
23                {{ variant.id }}<br/>
24              </div>
25            </div>
26          </div>
27        </div>
28      </div>
29    </div>
30  </div>
31
32  <script src="./main.js"></script>
33
34  <script>
35    window.addEventListener('load', () => {
36      const app = Vue.createApp({
37        data() {
38          return {
39            cart: 0,
40            product: 'Socks',
41            image: './assets/images/socks_blue.jpg',
42            inStock: true,
43            details: ['50% cotton', '30% wool', '20% polyester'],
44            variants: [
45              { id: 2234, color: 'green', image: './assets/images/socks_green.jpg' },
46              { id: 2235, color: 'blue', image: './assets/images/socks_blue.jpg' }
47            ]
48          }
49        },
50        methods: {
51          addToCart() {
52            this.cart += 1
53          },
54          updateImage(variantImage) {
55            this.image = variantImage
56          },
57          removeFromCart() {
58            if (this.cart > 0) {
59              this.cart -= 1
60            }
61          }
62        }
63      }).mount('#app')
64    })
65  </script>
66
67  <style>
68    .product-display {
69      border: 1px solid #ccc;
70      padding: 10px;
71      width: fit-content;
72      margin: auto;
73    }
74    .product-container {
75      border: 1px solid #ccc;
76      padding: 10px;
77      width: fit-content;
78      margin: auto;
79    }
80    .product-info {
81      border: 1px solid #ccc;
82      padding: 10px;
83      width: fit-content;
84      margin: auto;
85    }
86    .product-image {
87      width: 100px;
88      height: 100px;
89      border: 1px solid #ccc;
90      border-radius: 50%;
91      margin-bottom: 10px;
92    }
93    .product-details {
94      border: 1px solid #ccc;
95      padding: 10px;
96      width: fit-content;
97      margin: auto;
98    }
99  </style>
100 </body>
101</html>

```

Challenge of lesson 6 picture:

main.js

```

1 <script>
2   const app = Vue.createApp({
3     data() {
4       return {
5         cart: 0,
6         product: 'Socks',
7         image: './assets/images/socks_blue.jpg',
8         inStock: true,
9         details: ['50% cotton', '30% wool', '20% polyester'],
10        variants: [
11          { id: 2234, color: 'green', image: './assets/images/socks_green.jpg' },
12          { id: 2235, color: 'blue', image: './assets/images/socks_blue.jpg' }
13        ]
14      }
15    },
16    methods: {
17      addToCart() {
18        this.cart += 1
19      },
20      updateImage(variantImage) {
21        this.image = variantImage
22      },
23      removeFromCart() {
24        if (this.cart > 0) {
25          this.cart -= 1
26        }
27      }
28    }
29  </script>
30
31  <style>
32    .product-display {
33      border: 1px solid #ccc;
34      padding: 10px;
35      width: fit-content;
36      margin: auto;
37    }
38    .product-container {
39      border: 1px solid #ccc;
40      padding: 10px;
41      width: fit-content;
42      margin: auto;
43    }
44    .product-info {
45      border: 1px solid #ccc;
46      padding: 10px;
47      width: fit-content;
48      margin: auto;
49    }
50    .product-image {
51      width: 100px;
52      height: 100px;
53      border: 1px solid #ccc;
54      border-radius: 50%;
55      margin-bottom: 10px;
56    }
57    .product-details {
58      border: 1px solid #ccc;
59      padding: 10px;
60      width: fit-content;
61      margin: auto;
62    }
63  </style>
64
65  </body>
66</html>

```

Task of lesson 7 picture:

The screenshot shows a Vue.js application running in a browser. On the left, the Visual Studio Code editor displays the `index.html` file with the following code:

```

<html lang="en">
  <body>
    <div id="app">
      <div class="product-display">
        <div class="product-container">
          <div v-for="variant in variants" :key="variant.id" @mouseover="updateImage(variant.image)" class="color-circle" :style="{ backgroundColor: variant.color }"></div>
          <button class="button" @click="addToCart" :disabled="!variant.inStock">Add to Cart</button>
        </div>
      </div>
    </div>
  </body>
</html>

```

The browser window shows the rendered product page for a pair of socks. The product image is a dark blue sock with a small green logo. The title is "Socks". Below it, a message says "Out of Stock". To the right, there is a description: "50% cotton, 30% wool, 20% polyester". Underneath the description are two circular color swatches, one green and one blue. At the bottom right is a grey "Add to Cart" button.

Challenge of lesson 7 picture:

The screenshot shows a Vue.js application running in a browser. On the left, the Visual Studio Code editor displays the `index.html` file with the following code:

```

<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Vue Mastery</title>
  </head>
  <body>
    <div id="app">
      <div class="cart">Cart (0)</div>
      <div class="product-display">
        <div class="product-item">
          <img alt="A grey sock with a small green logo." data-v-for="det" :class="`color-circle ${det.color}`" />
          <div class="product-details">
            <h1>Socks</h1>
            <p>Out of Stock</p>
            <ul>
              <li>50% cotton</li>
              <li>30% wool</li>
              <li>20% polyester</li>
            </ul>
            <div>
              <span>Green</span>
              <span>Blue</span>
            </div>
            <button>Add to Cart</button>
          </div>
        </div>
      </div>
    </div>
  </body>
</html>

```

The browser window shows the rendered product page for a pair of socks. The product image is a grey sock with a small green logo. The title is "Socks". Below it, a message says "Out of Stock". To the right, there is a description: "50% cotton, 30% wool, 20% polyester". Underneath the description are two circular color swatches, one green and one blue. At the bottom right is a grey "Add to Cart" button. In the bottom left corner of the browser window, there is a debug console message:

```

You are running a development build of Vue.
Make sure to use the production build (*.prod.js) when deploying.

```