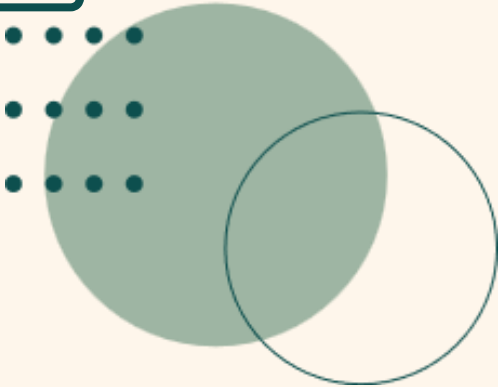




VUE-JS DASAR

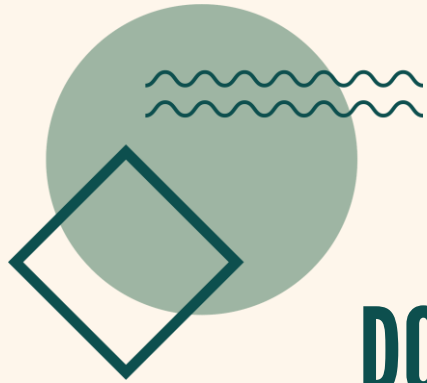


SRI ADINDA (233510515)

PENDAHULUAN

Sebuah Framework JavaScript yang digunakan untuk membangun antarmuka web yang interaktif. Fokus utama Vue.JS menyediakan mekanisme Reactive Data Binding dan menyediakan fungsi untuk mengatur komponen View dengan API yang sederhana mungkin.





DOWNLOAD DAN INSTALASI VUE-JS

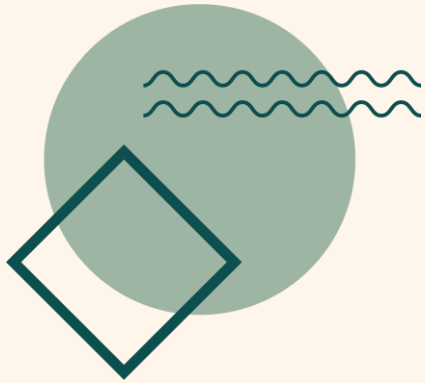




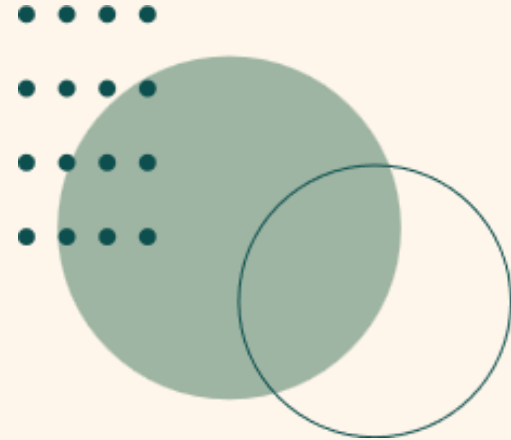
Bisa langsung mengakses situs Vue.js disini :

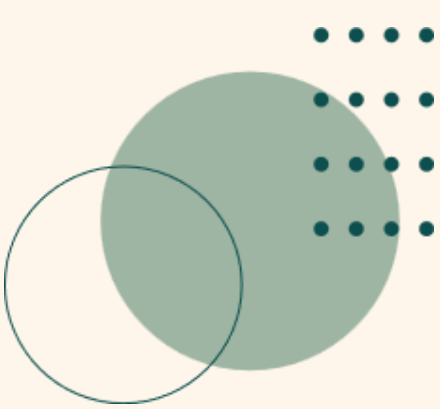
<https://vuejs.org/>





MEMBUAT PROJECT



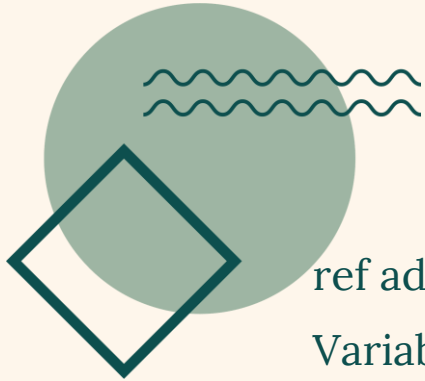


- Untuk membuat project Vue, gunakan **Vite** sebagai build tool.
- Untuk membuat project Vue menggunakan JavaScript :

```
npm create vite@latest
```

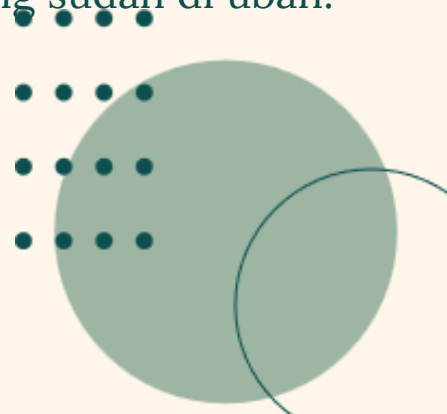


REACTIVE STATE



ref adalah cara Vue mendeklarasikan variable reaktif.

Variabel yang dideklarasikan menggunakan ref akan dipantau oleh Vue, maka Vue akan melakukan render ulang Componentnya, sehingga Component akan ditampilkan ulang dengan data State yang baru yang sudah di ubah.

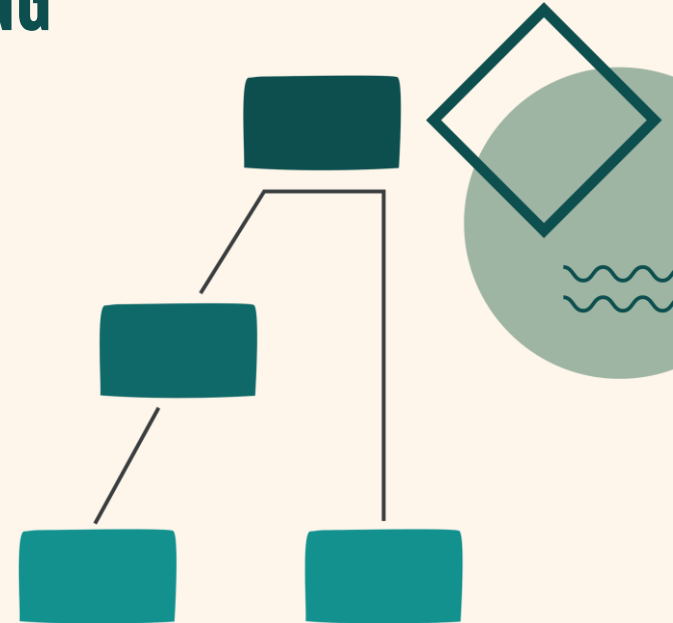


REACTIVE DATA BINDING

Berfungsi agar data dan DOM tetap terikat bersama. Jika biasanya menggunakan jQuery secara manual dalam memanipulasi DOM, kode yang ditulis pasti akan berulang dan rawan kesalahan.

Menganut data-driven yang artinya menggunakan sintaks khusus dalam template HTML untuk “**mengikat**” DOM.

Setelah binding diciptakan, DOM akan bersinkronisasi dengan data dan setiap kali mengubah data, Dom juga akan memperbaruinya.





TEXT INTERPOLATION

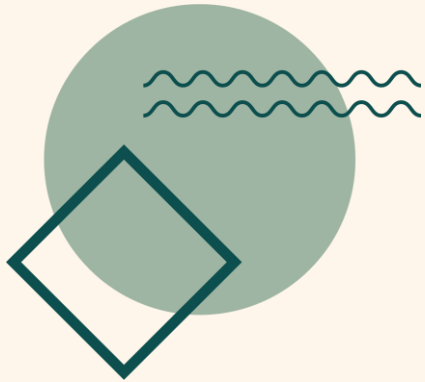
Text interpolation memungkinkan untuk menyisipkan nilai variable JavaScript ke dalam HTML menggunakan sintaks `{{ }}` atau kurawal double untuk mengakses data dari Component.

Memudahkan untuk menampilkan data dinamis pada elemen HTML

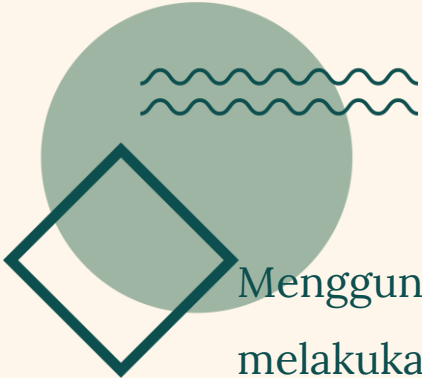


TEXT INTERPOLATION

```
1  <script setup>
2  import { ref } from 'vue'
3
4  const greeting = ref('Welcome to Vue!')
5  const name = ref('John Doe')
6  </script>
7
8  <template>
9    <h1>{{ greeting }} {{ name }}</h1>
10 </template>
11
```



FORM BINDING



Menggunakan direktif **v-model** untuk mengikat data ke elemen form atau melakukan sinkronisasi data sesuai inputan (textarea,select,input) .

Fitur ini memungkinkan kita untuk melakukan two way data binding, yang berarti setiap perubahan pada input akan memperbarui variable di Vue dan sebaliknya.



FORM BINDING

- Directive **v-model** mendukung banyak input Binding.
- Element `<input>` dan `<textarea>` akan menggunakan attribute value dan menggunakan `@input` event
- Element `<input>` type checkbox dan radio, menggunakan attribute checked dan menggunakan `@change` event
- Element `<select>` menggunakan attribute value dan menggunakan `@change` event

✓ Contact.vue ×

```
1  <script setup>
2  import {reactive} from "vue";
3
4  const contact = reactive({
5    name: "",
6    email: "",
7    age: 0,
8    type: "Regular",
9    complain: false,
10   message: ""
11  })
12  </script>
13
```

```
<template>
  <h1>Contact Us</h1>
  <form>
    <div>
      <label for="name">Name:
      <input type="text" id="name" v-model="contact.name">
    </label>
    </div>
    <div>
      <label for="email">Email:
      <input type="email" id="email" v-model="contact.email">
    </label>
    </div>
    <div>
      <label for="age">Age:
      <input type="number" id="age" v-model.number="contact.age">
    </label>
    </div>
  </form>
</template>
```

```
<div>
  <label for="type">Type:
    <select id="type" v-model="contact.type">
      <option value="Regular">Regular</option>
      <option value="VIP">VIP</option>
    </select>
  </label>
</div>
<div>
  <label for="complain">Complain:
    <input type="checkbox" id="complain" v-model="contact.complain">
  </label>
</div>
<div>
  <label for="message">Message:
    <textarea id="message" v-model="contact.message"></textarea>
  </label>
</div>
</form>
```

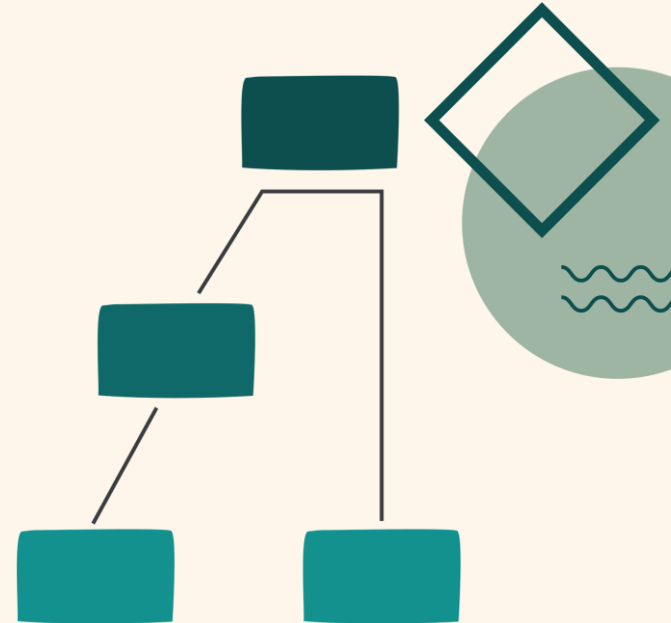
```
<h1>Preview</h1>

<div>
  <p>Name: {{ contact.name }}</p>
  <p>Email: {{ contact.email }}</p>
  <p>Age: {{ contact.age }}</p>
  <p>Type: {{ contact.type }}</p>
  <p>Complain: {{ contact.complain }}</p>
  <p>Message: {{ contact.message }}</p>
</div>

</template>
```

EVENT LISTENER

Event listener memungkinkan kita untuk menangani aksi yang dilakukan pengguna, seperti klik, ketikan, submit, dll. Vue mempermudah hal ini dengan menggunakan Directive **v-on** (shortcut **@**) untuk menghubungkan event dengan fungsi JavaScript.



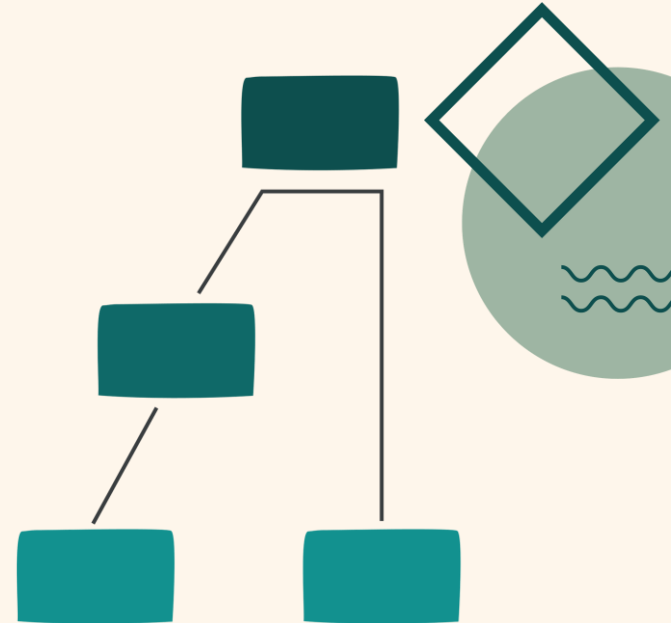
EVENT LISTENER

```
1  <script setup>
2  import { ref } from 'vue'
3  // Variabel reaktif untuk counter
4  const count = ref(0)
5  // Fungsi untuk increment counter
6  function increment() {
7    count.value++
8  }
9  </script>
10 <template>
11   <!-- Menggunakan v-on untuk mendengarkan event 'click' -->
12   <button v-on:click="increment">Click me</button> <!-- Menggunakan event listener -->
13   <p>Counter: {{ count }}</p>
14 </template>
```

- **V-on:click="increment"** menghubungkan event klik dengan fungsi increment, Setiap kali tombol diklik, maka nilai count akan bertambah satu dan UI akan diperbarui secara otomatis.
- Fungsi increment berfungsi untuk memodifikasi nilai dari variable reaktif count, yang secara otomatis diperbarui pada tampilan.

ATTRIBUTE BINDING

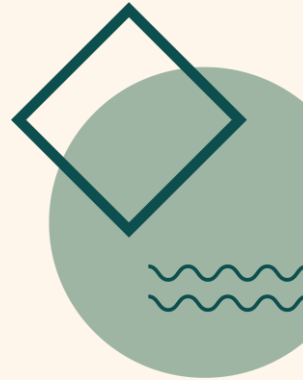
- Mendukung direktif **v-bind** untuk mengikat data Vue ke atribut HTML. Ini memungkinkan kita untuk mengubah atribut elemen DOM, seperti class, style, atau href secara dinamis berdasar data yang ada.
- Berguna untuk memperbarui tampilan seperti mengubah warna tombol saat diklik ataua memperbarui URL secara dinamis.



ATTRIBUTE BINDING

```
1 <script setup>
2 import { ref } from 'vue'
3 // Mendeklarasikan variabel reaktif untuk class tombol
4 const buttonClass = ref('normal-btn')
5 // Fungsi untuk mengubah class tombol ketika tombol diklik
6 const toggleButtonClass = () => {
7   // Mengubah kelas tombol antara 'normal-btn' dan 'active-btn'
8   buttonClass.value = buttonClass.value === 'normal-btn' ? 'active-btn' : 'normal-btn'
9 }
10 </script>
11 <template>
12   <!-- Menggunakan v-bind untuk mengubah class tombol secara dinamis -->
13   <button v-bind:class="buttonClass" @click="toggleButtonClass">Click Me</button>
14 </template>
```

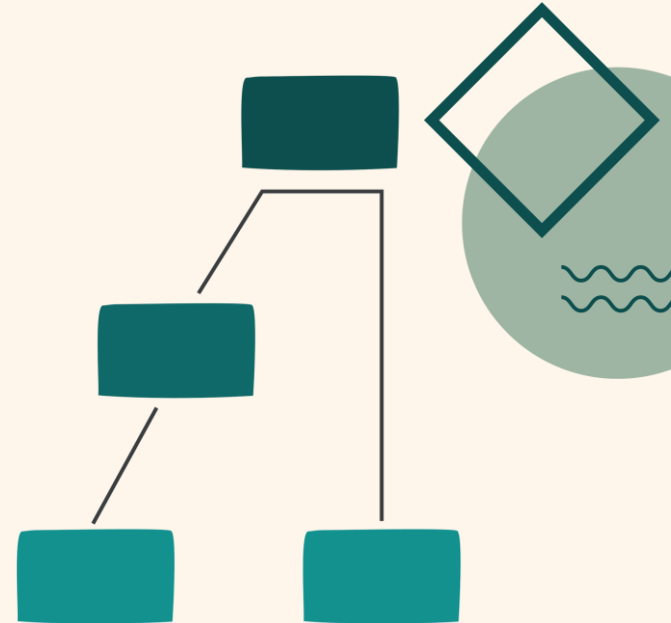
Ketika tombol di klik, nilai dari buttonClass berubah antara **normal-btn** dan **active-btn** , yang menyebabkan perubahan gaya tombol sesuai dengan definisi CSS.



CONDITIONAL RENDERING

Memungkinkan kita untuk menampilkan atau menyembunyikan elemen DOM berdasarkan kondisi tertentu. Vue menyediakan banyak directive untuk melakukan Conditional Rendering.

Ada **v-if** untuk kondisi if, **v-else-if** untuk kondisi else if, dan **v-else** untuk kondisi else.





TERIMA KASIH

