MODUL 1 – DASAR Vue.js

Ikuti seluruh instruksi yang diberikan lengkap dengan penjelasannya untuk memahami materi dasar-dasar Vue.js dengan baik

Sebelum memulai, buat folder baru dengan nama MODUL1. Semua file dalam modul ini akan disimpan dalam folder tersebut.

# Environment Setup (CDN)

1. Buat file baru dengan nama lat1.html
2. Di dalam lat1.html, buat struktur dasar HTML
3. Di dalam tag <head>, tambahkan tag script untuk menambahkan library Vue.js, seperti berikut.

<script src="https://cdn.jsdelivr.net/npm/vue"></script>

1. Simpan.

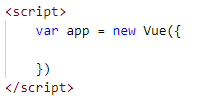
# Introduction

1. Di dalam body, tambahkan tag div dengan id=”app” dan tag <script> tepat di bawahnya.



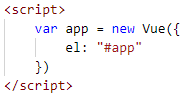
Elemen div#app akan dikontrol melalui instance dari Vue yang dideklarasi di dalam tag <script>

1. Di dalam tag <script> ketikkan baris kode berikut untuk membuat inisialisasi Vue object.



Vue object ini terdiri atas berbagai macam property yang sudah ditentukan oleh Vue. Inisialisasi diperlukan untuk mengontrol elemen div#app, misalnya untuk menampilkan teks tertentu.

1. Property pertama yang akan ditambahkan adalah el, dimana value atau nilainya merupakan id dari elemen yang menjadi target untuk dikontrol. Untuk menentukan div#app sebagai target, ketikkan baris kode berikut.



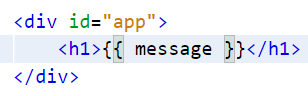
1. Setelah div#app ditetapkan menjadi target, kontrol dapat dilakukan melalui object. Misal, untuk menampilkan pesan “Hello World”. Tambahkan baris kode berikut.



Property data merupakan object sehingga juga memiliki property dan value, dimana message merupakan property dan ‘Hello World’ sebagai value

*n.b: Perhatikan penambahan koma di antara property el dan data*

1. Teks “Hello World” telah ditambahkan melalui data. Berikutnya kembali ke div#app untuk menempatkan teks tersebut di dalam <h1> menggunakan ekspresi {{ }}, sebagai berikut.



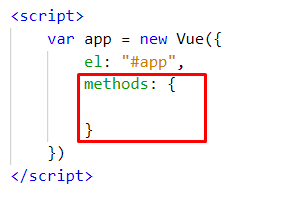
1. Simpan dan jalankan di browser.

# Methods

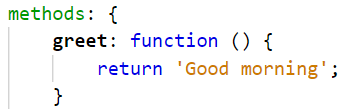
1. Buat file baru dengan nama lat2.html. Untuk latihan ini, kode Javascript ditambahkan di dalam lat2.html menggunakan tag script.
2. Di dalam lat2.html tambahkan kode berikut.



1. Di dalam tag script, tambahkan baris kode untuk inisialisasi Vue object seperti yang dilakukan di latihan sebelumnya.
2. Setelah property el, tambahkan property methods untuk mendeklarasi method baru.

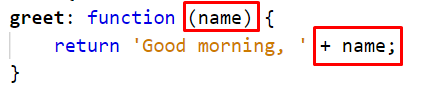


Lengkapi dengan menambahkan nama method serta barisan kode yang dieksekusi ketika method dipanggil



Ketika method ‘greet’ dipanggil di dalam div#app, akan dikembalikan string ‘Good morning’.

1. Kembali ke lat2.html, kemudian tambahkan elemen <p> di dalam div#app untuk tempat memanggil method greet.
2. Baris kode yang ditambahkan untuk memanggil method greet di dalam <p> adalah \_\_\_\_\_\_\_\_
3. Simpan dan jalankan
4. Untuk melewatkan nilai ke dalam method greet, tambahkan parameter sebagai berikut.



1. Baris kode yang ditambahkan untuk memanggil method greet yang sudah dimodifikasi adalah \_\_\_\_\_\_\_\_\_\_
2. Simpan dan jalankan di browser.



# Data Binding

Data binding adalah cara atau teknik untuk mengikat (*binding*) data pada view. Terdapat beberapa directive yang digunakan, salah satunya adalah v-bind. Directive ini dapat digunakan untuk binding value pada atribut HTML.

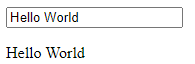
1. Buat file baru dengan nama lat3.html
2. Di dalamnya tambahkan CDN Vue.js, elemen div#app, dan tag script untuk menambahkan barisan kode Javascript (Vue.js)
3. Inisialisasi object Vue.js baru, kemudian target elemen div#app dengan property el (silahkan mengacu pada lat1.html)
4. Di dalam div#app, tambahkan elemen-elemen berikut.



1. Tambahkan object data di dalam instance Vue, seperti berikut.



1. Perhatikan tag <p> yang berisi ekspresi {{ message }} dan v-bind yang mengikat atribut value dari tag <input> juga dengan nilai message di dalamnya. Jika dieksekusi, harusnya elemen p dan input menampilkan teks yang sama, yaitu “Hello World”, sesuai dengan nilai dari property message yang ditambahkan pada data. Untuk mengujinya, simpan dan jalankan di browser.



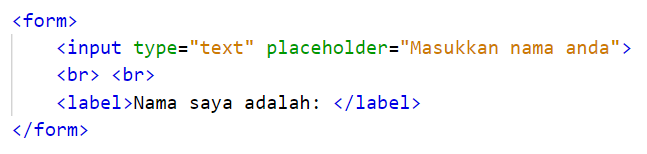
Sekarang, coba hapus teks di dalam <input>, ganti dengan teks lain. Apakah <p> masih menampilkan teks yang sama dengan isi dari <input>?

Jawabannya tidak, karena <input> dan <p> telah terikat satu arah dengan property message melalui v-bind. Ikat satu arah atau one way binding adalah teknik di mana dalam contoh di atas, nilai dari property “message” dapat mengontrol <input> dan <p>, tetapi tidak sebaliknya. Oleh karena itu, berapa kali pun isi dari <input> diubah, nilai dari property “message” tetap, dan hasilnya teks yang ditampilkan oleh <p> juga tetap sama.

# Two Way Data Binding

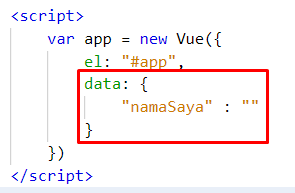
Teknik data binding berikutnya disebut dengan two way data binding atau ikat dua arah.

1. Buat file baru dengan nama lat4.html
2. Tambahkan CDN Vue.js, div#app, dan tag script
3. Inisialisasi Vue.js object dan target elemen div#app menggunakan property el
4. Tambahkan satu kolom isian dan satu label di dalam div#app



Dengan teknik two way data binding, nilai yang diketikkan di kolom isian dapat ditampilkan secara langsung pada user. Directive yang dibutuhkan adalah v-model

1. Tambahkan property namaSaya dalam data. Nilainya dibuat kosong terlebih dulu.



1. Berikutnya, pada tag input, tambahkan directive v-model , seperti berikut.



1. Untuk menampilkan value pada label, panggil property namaSaya seperti berikut.

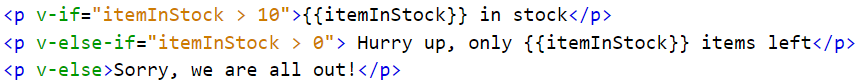


Simpan dan jalankan pada browser. Isi kolom isian dan lihat apakah nilainya langsung ditampilkan pada label.

# Condition

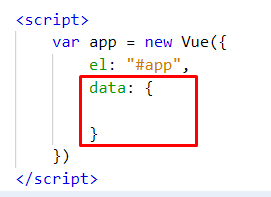
Pada bagian ini, akan ditampilkan item dengan kondisi-kondisi tertentu menggunakan directive v-if, v-else-if, dan v-else.

1. Buat file baru dengan nama lat5.html
2. Tambahkan CDN Vue.js, div#app, dan tag script untuk barisan kode Vue.js
3. Inisialisasi object Vue.js, kemudian target div#app dengan property el
4. Di dalam div#app, tambahkan tiga komponen paragraf yang akan ditampilkan dalam tiga kondisi berbeda.

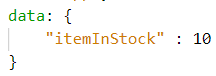


* Perhatikan bagaimana nilai dari masing-masing directive merupakan sebuah ekspresi yang menghasilkan nilai logika TRUE atau FALSE. Hasil dari pengujian ekspresi itulah yang akan menentukan paragraf mana yang akan ditampilkan
* Paragraf pertama ditampilkan jika nilai dari itemInStock > 10
* Paragraf ke-2 ditampilkan jika kondisi sebelumnya tidak terpenuhi dan nilai dari itemInStock > 0
* Paragraf ke-3 ditampilkan jika dua kondisi sebelumnya tidak terpenuhi

1. Setelah property el, tambahkan property data dengan value berupa object seperti berikut.



1. Tambahkan property itemInStock dengan value 10



1. Simpan dan jalankan hasilnya pada browser.
2. Ubah nilai itemInStock untuk menampilkan paragraf lainnya.

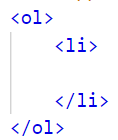
# Loop

Pada bagian ini, akan diterapkan perulangan array menggunakan directive v-for

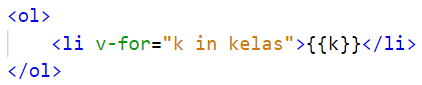
1. Buat file baru dengan nama lat6.html
2. Tambahkan CDN Vue.js, div#app, dan tag script
3. Inisialisasi Vue.js object dan target elemen div#app menggunakan property el
4. Tambahkan property data dengan value berupa object, dimana property-nya adalah ‘kelas’ dengan value berupa array, seperti berikut.



1. Di dalam div#app, tambahkan tag <ol> dan <li> untuk menampilkan semua elemen dari array kelas.



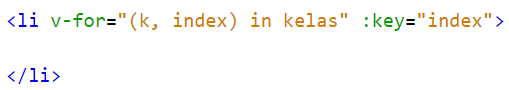
1. Di dalam tag <li> tambahkan directive v-for untuk melakukan loop terhadap array kelas dan menampilkan semua isinya.



k adalah variabel yang menampung item-item kelas satu per satu. Oleh karena itu, untuk menampilkan setiap item, dipanggil variabel k menggunakan {{}}

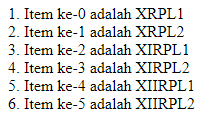
1. Simpan dan jalankan hasilnya di browser

**Tambahan: untuk menampilkan index dari perulangan, dapat digunakan kode v-for seperti berikut.**



Dimana index adalah counter dari struktur looping, dan menyimpan nilai 0

1. Untuk menampilkan hasil seperti berikut, silahkan memodifikasi struktur loop yang sudah ada.

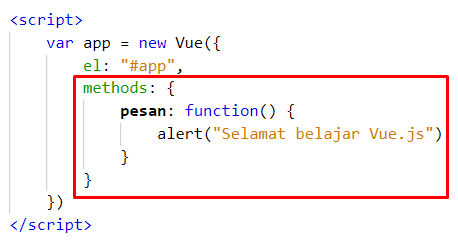


Petunjuk: manfaatkan index dari setiap item di dalam array kelas untuk mencapai hasil seperti yang diminta.

# Event Handling

Menangani event dalam Vue.js dapat menggunakan directive v-on.

1. Buat file baru dengan nama lat7.html
2. Tambahkan CDN Vue.js, elemen div#app, dan tag script untuk baris kode Javascript (Vue.js)
3. Inisialisasi object Vue.js, kemudian target div#app dengan property el
4. Setelah property el, tambahkan property methods. Di dalamnya, tambahkan method baru untuk memanggil method pesan(). Method ini akan menampilkan pesan “Selamat belajar Vue.js” ketika dipanggil.



1. Di dalam div#app, tambahkan button untuk memanggil method pesan()



v-on:click digunakan untuk menangani event pemanggilan fungsi melalui klik tombol. Perhatikan bagaimana nilai dari v-on:click adalah nama dari fungsi yang hendak dipanggil ketika event klik tombol terjadi.

1. Simpan dan jalankan aplikasi di browser.
2. Vue.js juga menyediakan shorthand untuk penggunaan v-on, sehingga ketika ditulis lengkap adalah

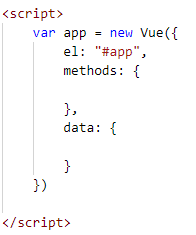
v-on:click=”pesan”

dan ketika dipersingkat menjadi

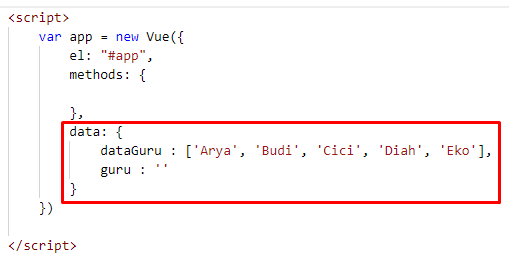
@click=”pesan”

Vue.js menyediakan key modifiers untuk penanganan event keyboard menggunakan keyup. Contohnya penanganan event ketika menekan tombol Enter. Untuk contoh penggunaan key modifier, akan dibuat form dimana data disimpan dengan menekan tombol Enter atau menekan tombol.

1. Buat file baru dengan nama lat8.html
2. Tambahkan struktur dasar HTML, dan CDN untuk Vue.js
3. Inisialisasi Vue.js object dan target elemen div#app menggunakan property el

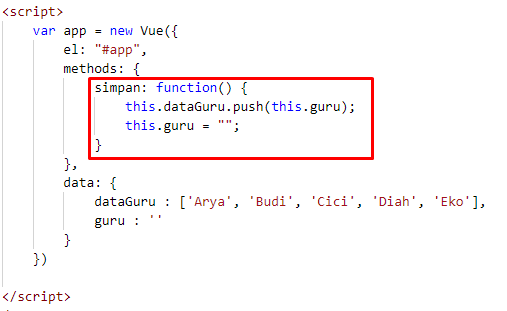


1. Di dalam data, tambahkan property ‘dataGuru’ dan ‘guru’



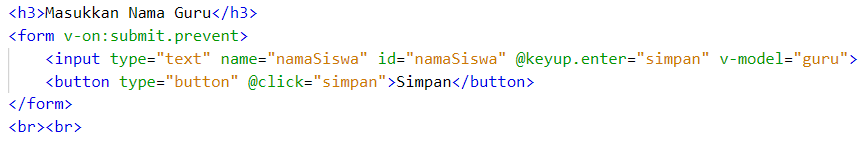
Perhatikan bahwa value dari ‘dataGuru’ adalah array dengan item berupa enam nilai string sedangkan property ‘guru’ adalah string kosong.

1. Tambahkan method baru untuk menyimpan data dari form



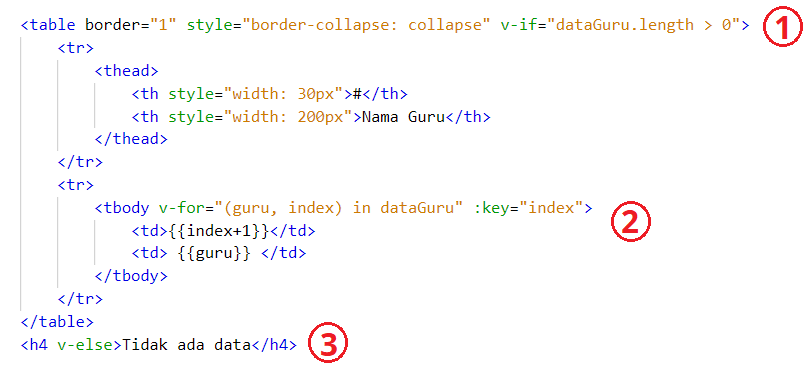
guru nantinya akan di-bind dengan kolom isian form menggunakan two-way data binding. Input user disimpan di dalam guru kemudian ditambahkan sebagai elemen dari array dataGuru.

1. Di div#app, tambahkan form yang berisi kolom isian dan tombol submit (hapus elemen tombol sebelumnya).



* v-on:submit.prevent ditambahkan agar halaman tidak dimuat ulang ketika data disubmit.
* @keyup.enter (shorthand dari v-on:keyup.enter) ditambahkan untuk memanggil method simpan ketika tombol Enter ditekan setelah input data di kolom isian
* v-model ditambahkan untuk two-way data binding dengan property guru yang akan ditambahkan di object data nanti
* @click akan memanggil method simpan

1. Setelah tag <br> tambahkan baris kode berikut untuk menambahkan <table>. Seluruh isian dari form akan ditampilkan dalam tabel ini.



1. v-if menguji kondisi jumlah elemen yang disimpan di dalam array dataGuru. Jika jumlah lebih dari 0 maka tabel ditampilkan
2. Menampilkan semua isi dari array dataGuru. Index adalah counter dari v-for dan nilainya ditampilkan sebagai nomor tabel.
3. Ditampilkan jika kondisi nomor 1 tidak terpenuhi
4. Simpan dan jalankan di browser.

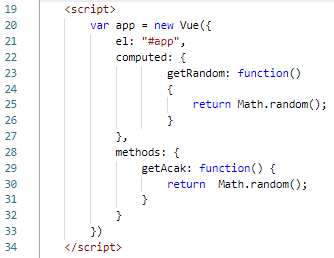
# Computed Properties

Bagian ini akan ditunjukkan bagaimana menampilkan angka hasil pengacakan melalui dua cara, yaitu dengan computed property dan method, kemudian hasil keduanya akan dibandingkan. Dari hasil perbandingan, akan dapat dipahami dengan mudah apa fungsi dari computed property.

1. Buat file dengan nama lat9.html.
2. Di dalam latihan.html, buat struktur dasar HTML
3. Tambahkan CDN Vue.js, div#app, dan tag script untuk barisan kode Vue.js
4. Inisialisasi object Vue.js, kemudian target div#app dengan property el

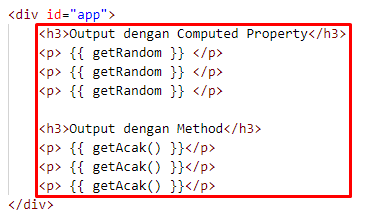


1. Setelah el, tambahkan kode berikut seperti berikut.



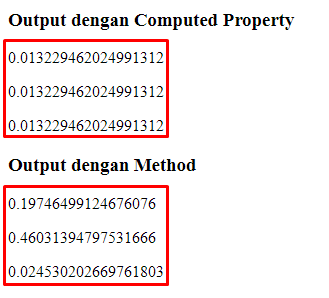
* Baris 22-27 adalah deklarasi dari computed property getRandom
* Baris 28-32 adalah deklarasi dari method getAcak()
* Dari script di atas, diketahui bahwa jika getRandom dan getAcak dipanggil, maka keduanya akan menghasilkan nilai acak dari Math.random(), yaitu angka desimal 0-1.

1. Untuk menampilkan hasil, tambahkan kode berikut di dalam div#app



* Perhatikan bagaimana computed property getRandom dipanggil seperti data property pada umumnya
* Sedangkan method getAcak() dipanggil sebagaimana method dipanggil, yaitu diakhiri dengan tanda kurung ()

Simpan dan jalankan pada browser, bandingkan output keduanya



Pada getRandom, fungsi dijalankan untuk menghasilkan angka acak 0-1, kemudian hasilnya disimpan sebagai suatu computed property. Seperti tampak pada hasil di atas, nilainya sama meskipun dipanggil 3 kali. Nilai dari getRandom berubah hanya ketika dibutuhkan, misalnya ketika halaman dimuat ulang.

Berbeda dengan computed property, getAcak() akan menampilkan hasil berbeda setiap kali dipanggil kembali. Oleh karena itu, ketiga hasil dari pemanggilan getAcak() di atas menampilkan nilai yang berbeda-beda.

Dengan demikian, dapat disimpulkan bahwa computed property adalah property yang menyimpan nilai hasil komputasi. Proses komputasi ini dideklarasi sebagai suatu fungsi yang mengembalikan suatu nilai.

# Lampiran Kode

**Latihan 1**

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>Latihan 1</title>      <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>  </head>  <body>      <div id="app">          <h1>{{ message }}</h1>      </div>      <script>          var app = new Vue({              el: "#app",              data: {                  message: "Hello world"              }          })      </script>  </body>  </html> |

**Latihan 2**

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>Latihan 2</title>      <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>  </head>  <body>      <div id="app">          <p>{{ greet('Bias Damiasa') }}</p>      </div>      <script>          var app = new Vue({              el: "#app",              methods: {                  greet: function(name){                      return 'Good morning, ' + name;                  }              }            })      </script>  </body>  </html> |

**Latihan 3**

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>Latihan 3</title>      <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>  </head>  <body>      <div id="app">          <input type="text" v-bind:value="message">          <p>{{ message }}</p>      </div>      <script>          var app = new Vue({              el: "#app",              data: {                  message: "Hello World"              }          })      </script>  </body>  </html> |

**Latihan 4**

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>Latihan </title>      <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>  </head>  <body>      <div id="app">          <form>              <input type="text" placeholder="Masukkan nama Anda" v-model="namaSaya">              <br><br>              <label>Nama saya adalah: {{namaSaya}} </label>          </form>      </div>      <script>          var app = new Vue({              el: "#app",              data: {                  "namaSaya" : ""              }            })      </script>  </body>  </html> |

**Latihan 5**

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>Latihan 5</title>      <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>  </head>  <body>      <div id="app">          <p v-if="itemInStock > 10">{{itemInStock}} in stock</p>          <p v-id="itemInStock > 0">Hurry up, only {{itemInStock}} items left</p>          <p v-else>Sorry, we are all out</p>      </div>      <script>          var app = new Vue({              el: "#app",              data: {                  itemInStock: 10              }          })      </script>  </body>  </html> |

**Latihan 6**

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>Latihan 6</title>      <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>  </head>  <body>      <div id="app">          <ol>              <li v-for="k in kelas">{{k}}</li>          </ol>          <ol>              <li v-for="(k, index) in kelas" :key="index">                  Item ke-{{index}} adalah {{k}}              </li>          </ol>      </div>      <script>          var app = new Vue({              el: "#app",              data: {                  "kelas" : ["XRPL1", "XRPL2", "XIRPL1", "XIRPL2", "XIIRPL1", "XIIRPL2"]              }            })      </script>  </body>  </html> |

**Latihan 7**

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>Latihan 7</title>      <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>  </head>  <body>      <div id="app">          <button v-on:click="pesan">Tampil Pesan</button>      </div>      <script>          var app = new Vue({              el: "#app",              methods: {                  pesan: function() {                      alert("Selamat belajar Vue.js")                  }              }          })      </script>  </body>  </html> |

**Latihan 8**

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>Latihan </title>      <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>  </head>  <body>      <div id="app">          <h3>Masukkan nama guru: </h3>          <form v-on:submit.prevent>              <input type="text" @keyup.enter="simpan" v-model="guru">              <button type="button" @click="simpan">Simpan</button>          </form>          <br><br>          <table border="1" style="border-collapse: collapse;" v-if="dataGuru.length > 0">              <tr>                  <thead>                      <th style="width:30px">#</th>                      <th style="width:200px">Nama Guru</th>                  </thead>              </tr>              <tr>                  <tbody v-for="(data, index) in dataGuru" :key="index">                      <td>{{ index + 1 }}</td>                      <td>{{ data }}</td>                  </tbody>              </tr>          </table>      </div>      <script>          var app = new Vue({              el: "#app",              methods: {                  simpan: function() {                      this.dataGuru.push(this.guru);                      this.guru = "";                  }              },              data: {                  dataGuru : ['Arya', 'Budi', 'Cici', 'Diah', 'Eko'],                  guru : ''              }          })        </script>  </body>  </html> |

**Latihan 9**

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>Computed Property</title>      <script src="https://cdn.jsdelivr.net/npm/vue"></script>    </head>  <body>        <div id="app">          <h3>Output dengan Computed Property</h3>          <p> {{ getRandom }} </p>          <p> {{ getRandom }} </p>          <p> {{ getRandom }} </p>          <h3>Output dengan Method</h3>          <p> {{ getAcak() }}</p>          <p> {{ getAcak() }}</p>          <p> {{ getAcak() }}</p>      </div>      <script>          var app = new Vue({              el: "#app",              computed: {                      getRandom: function()                      {                          return Math.random();                      }              },              methods: {                  getAcak: function() {                      return  Math.random();                  }              }          })      </script>  </body>  </html> |