Overview

Pada Mata kuliah kali ini kita akan mempelajari tentang bagaimana cara membuat sebuah aplikasi CRUD Management Produk sederhana. Disini anda akan membuat setiap component dari awal.

Anda akan mempelajari bagaimana untuk :

- Membuat aplikasi web dari awal menggunakan Vue.js 2.x.x
- Menggunakan vue-router utnuk melakukan routing pada setiap bagian dari aplikasi anda.
- Membuat unidirectional flow atau flux pattern menggunakan vuex untuk memisahkan data dan tampilan
- Menghubungkan aplikasi web kita dengan backend REST API menggunakan axios sebagai http request library

---

Key to perfection

Selama proses workshop akan ada beberap bagian "**Key to perfection**". Yang merupakan tugas dengan beberapa petunjuk yang berkaitan dengan permasalahan dunia nyata. Tugas-tugas tersebut dapat anda kerjakan apabila anda menginginkan tantangan lebih.

Persiapan

Sebelum kita memulai membuat aplikasi kita mengguanakan Vue.js, berikut ini adalah beberapa hal yang kita perlukan untuk mulai bekerja dengan baik :

- Node.js versi 6 ke atas.
- Code editor pilihan masing-masing, (disarankan menggunakan Atom dengan package Vue-component languange yang terinstall).
- Web Browser Modern (Disarankan mengguakan Google Chrome dengan Vue Dev Tools yang sudah terinstall)

Hello World!

Salah satu kelebihan paling utama dari Vue.js adalah flexibilitas dimana kita dapat membuat sebuah web application dengan berbagai cara tergantung pada kebutuhan dan skala dari web application kita.

Kita akan memulai dengan membuat sebuah aplikasi "Hello World!" yang sangat sederhana. Untuk memulai buatlah sebuah file html dengan code sebagai berikut :

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Hello world!</title>
  <!-- memasukkan Vue.js dari CDN unpkg-->
  <script src="https://unpkg.com/vue"></script>
  <script>
    // menunggu hingga element <body> selesai dimuat
    // pada jQuery sama seperti $(document).ready()
    document.addEventListener('DOMContentLoaded', function () {
      // membuat instance vue yang baru dengan element template serta
      // data atau state yang perlu ditampilkan
      new Vue({
        el: '#app',
        data: {
          message: 'Hello World!'
        }
      })
    })
  </script>
</head>
<body>
  <div id="app">
    {{ message }}
  </div>
</body>
</html>
```

Bukan file tersebut di web browser anda, dan lihat tampilan yang dimunculkan pada web browser tersebut.

Itu semua merupakan hal yang kita perlukan untuk mulai membuat sebuah aplikasi Vue.js sederhana dengan memisahkan bagian data atau state dengan bagian view atau presentation.

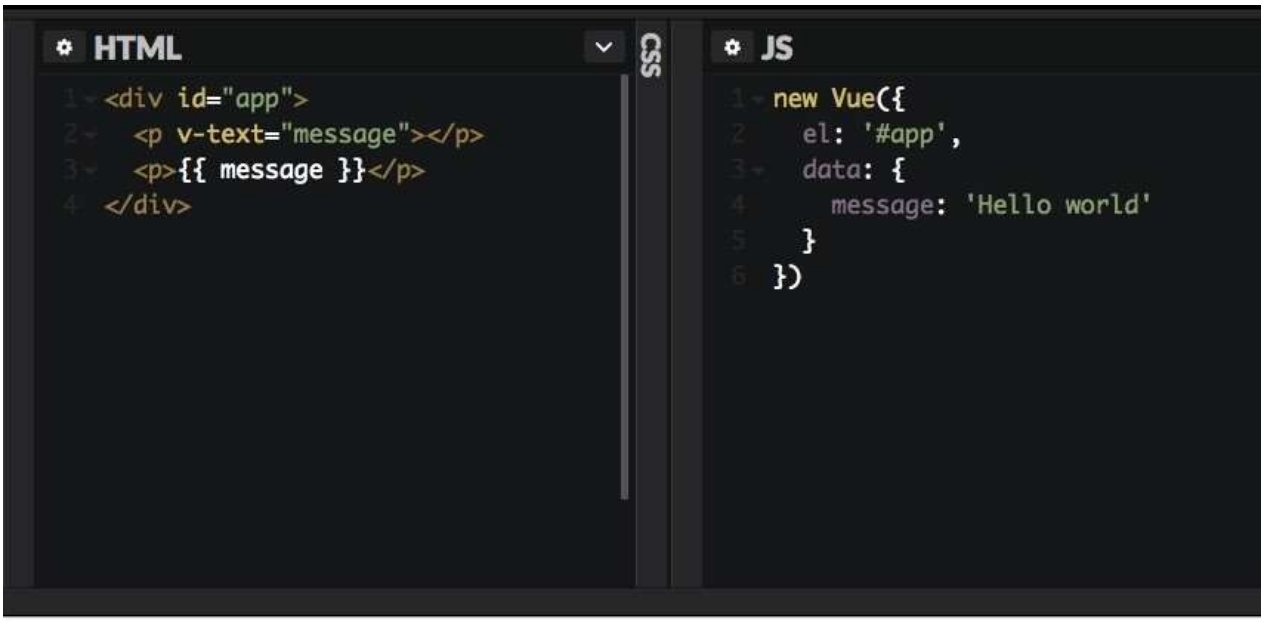
Sekalipun kedepannya kita akan mencoba untuk membagun web application yang lebih kompleks, dengan contoh "Hello World!" saya ingin menunjukkan bahwa sebenarnya sangat mudah untuk memulai membuat aplikasi dengan Vue.js dibandingkan dengan menggunakna framework yang lain. Seperti yang sudah diterangkan pada bagian sebelumnya, salah satu kelebihan dari Vue.js dibanding framework lain adalah fleksibilitas dimana kita bisa memulai project dari yang paling sederhana dan menambahkan beberapa add-ons official seperti vue-router dan vuex.

Mengenal Directive (v-*)

Salah satu fitur yang dimiliki oleh Vue.js adalah penggunaan **_directive.** _Directive sendiri merupakan sebuah fitur yang memungkinkan kita untuk menambahkan sebuah special attribute pada tag HTML dari web application kita. Special attribute tersebut kemudian akan memberitahu library atau framework yang kita gunakan untuk melakukan manupilasi terhadap DOM. Pada Vue.js sendiri setiap attribute directive yang biasa digunakan memiliki awalan atau prefix `v-` .

Contoh Sederhana :

```
<p v-text="message"></p>
```



Hello world

Hello world

Pada contoh script di atas terdapat sebuah tag `<p>` yang memiliki attributes `v-text` . Bisa kita lihat bahwa tag <p> tersebut tidak memiliki isi content. Akan tetapi karena memiliki attribute `v-text` yang value-nya merujuk ke state `message` , maka secara otomatis Vue.js akan melakukan manipulasi DOM dengan menambahkan isi content pada tag `<p>` tersebut sesuai dengan value dari state `message` script Vue.js kita.

Secara teknis attribute `v-text` akan mengupdate value `textContent` setiap saat value message di `ViewModel` atau `state` kita berubah.

Untuk passing sebuah value ke content dari sebuah element html kita juga dapat melakukan interpolation dengan menggunakan script double curly braces `{{ }}` seperti yang terlihat di screenshot di atas.

Daftar Lengkap Directive pada Vue.js

- `v-bind`
- `v-model`
- `v-text`
- `v-html`
- `v-on`
- `v-if`
- `v-else-if`
- `v-else`
- `v-show`
- `v-for`

v-bind

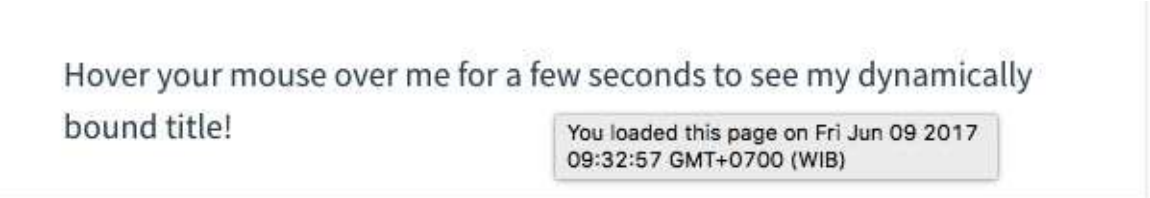
v-bind berfungsi untuk melakukan data binding dengan sebuah attribute dari sebuah tag HTML. Contoh penggunaannya sebegai berikut :

```
<div id="app">
  <span v-bind:title="message">
    Hover your mouse over me for a few seconds
    to see my dynamically bound title!
  </span>
</div>
```

```
new Vue({
  el: '#app',
  data: {
    message: 'You loaded this page on ' + new Date()
  }
})
```

Script di atas berfungsi untuk melakukan binding data dari attribute title dari tag span tersebut dengan value dari state message dari instance Vue. Jadi setiap kali terjadi perubahan value pada state `message` value pada attribute `title` tersebut juga akan ikut berubah menyesuaikan.

Hasil dari script di atas kurang lebih akan seperti pada gambar di bawah ini. DImana ketika kita melakukan mose hove di atas text tersebut akan muncul text sesuai dengan value di state `message` .



v-model

v-model berfungsi untuk melakukan two-way data binding yang umumnya dilakukan pada form input. DImana value pada input yang memiliki attribute directive v-model akan merujuk pada state yang menjadi reference-nya, dan juga setiap kali terjadi perubahan value pada form input tersebut, value dari state juga akan turut berubah.

```
<div id="app">
  <p>{{ message }}</p>
  <input v-model="message">
</div>
```

```
new Vue({
  el: '#app',
  data: {
    message: 'Hello Vue!'
  }
})
```

v-text

v-text berfungsi untuk melakukan binding sebuah value dari state yang berupa text ke dalam innerHTML dari sebuah element. Attribute directive ini memiliki fungsi yang mirip dengan penggunaan interpolation dengan double curly braces `{{ }}` .Contoh dari penggunaan v-text adalah sebagai berikut.

```
<p v-text="message"></p>
```

```
new Vue({
 el: '#app',
 data: {
  message: 'Hello Vue!'
 }
})
```

v-html

v-html memiliki fungsi yang hampir sama dengan v-text. Namun berbeda dengan v-text yang akan merender sebuah value dari sebuah state yang berupa plain text, v-html akan merender value dari state yang berupa kode HTML. Contoh penggunaan v-html adalah sebagai berikut :

```
<div v-html="message"></div>
```

```
new Vue({
 el: '#app',
 data: {
  message: '<p>Hello <strong>Vue!</strong></p>'
 }
})
```

v-on

Berbeda dengan attribute directive sebelumnya yang lebih fokus pada operasi binding data. Attribute v-on digunakan untuk melakukan event-handling. Attribute v-on akan untuk mengeksekusi atau menjalankan sebuah fungsi atau methods dari instance Vue yang diterima sebagai value dari directive atrribute ini. Sebelum menerima value, attribute ini juga menerima keypath yang akan menentukan jenis event yang akan dihandle seperti click, keyup, submit, dsb.

Contoh penggunaan-nya sebagai berikut :

```
<button type="button" v-on:click="showAlert">Click Me!</button>
```

```
new Vue({
 el: '#app',
 data: {
  message: 'Hello Vue!'
 },
 methods: {
  showAlert () {
   alert(this.message)
  }
 }
})
```

v-if

Attribute directive v-if digunakan untuk melakukan conditional rendering. Dimana tag atau element HTML yang memiliki attribute ini akan ditampilkan bedasarkan kondisi value pada state, atau sama sekali tidak direnderer ketika kondisi value pada state tersebut false. Attribute ini dapat dikombinasikan dengan attribute v-else-if dan juga v-else.

Contoh penggunaan-nya sebagai berikut :

```
<h1 v-if="status !== true && status !== false">The status is ambiguous</h1>
<h1 v-else-if="status">The status is true</h1>
<h1 v-else>The status is false</h1>
```

```
new Vue({
 el: '#app',
 data: {
  status: 'I make it ambigous'
 }
})
```

v-show

v-show memiliki fungsi yang mirip dengan v-if, akan tetapi dengan sedikit perbedaan, dimana apabila kondisi yang menjadi parameter atau value dari v-if adalah false maka element tersebut tidak akan direnderer sama sekali. Berbeda dengan v-show apabila kondisi parameter-nya tidak terpenuhi elemen tersebut akan tetap di render, hanya saja akan di hide dengan property display pada styling-nya menjadi none.

v-for

Attribute directive ini digunakan untuk melakukan list rendering dari list yang berdasarkan data yang memiliki bentuk array.

`v-for` membutuhkan syntax special yang berbentuk `item in items` , dimana `items` adalah sumber data array dan `item` adalah alias untuk setiap element dari array yang di iterasi.

Contoh penggunaannya sebagai berikut :

```
<ul>
  <li v-for="student in students">{{ student.name }}</li>
</ul>
```

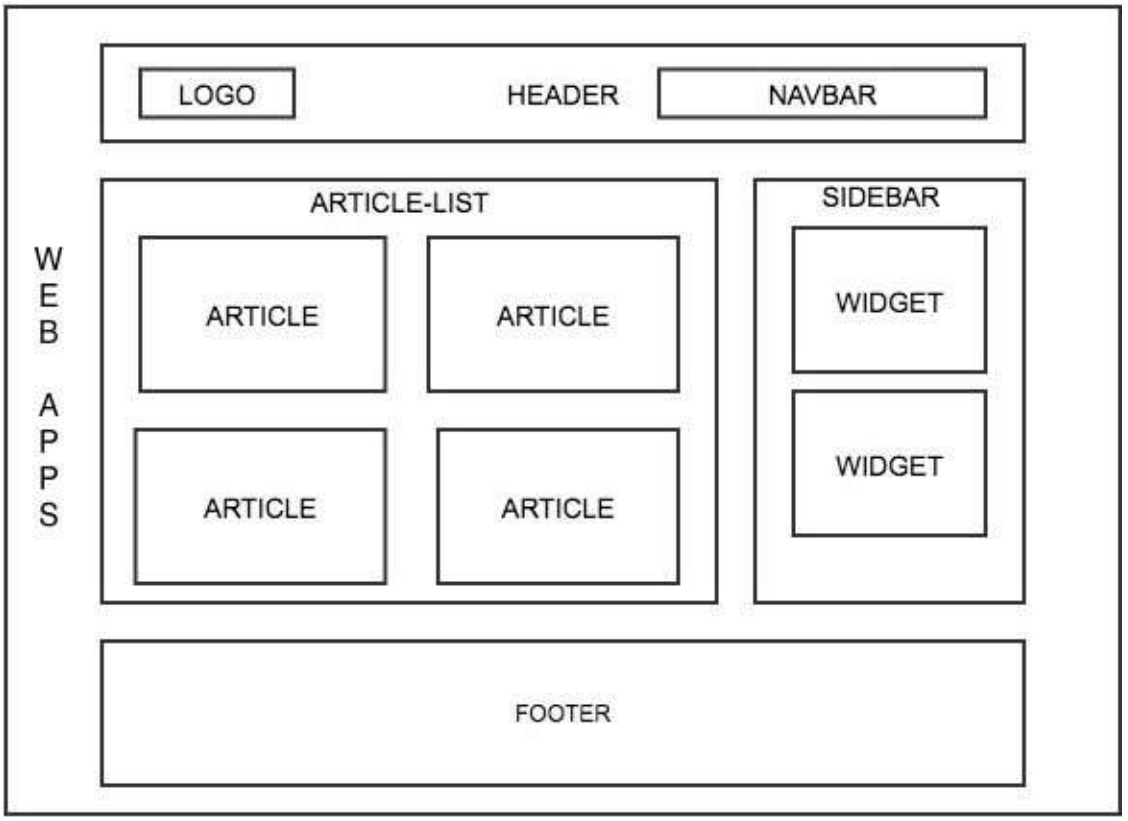
```
new Vue({
  el: '#app',
  data: {
    students: [
      {
        name: 'Peter Parker',
        gender: 'male'
      },
      {
        name: 'Steve Rogers',
        gender: 'male'
      },
      {
        name: 'Natasha Romanov',
        gender: 'female'
      },
      {
        name: 'Tony Stark',
        gender: 'male'
      }
    ]
  }
})
```

Mengenal Component

Salah satu fitur yang paling powerful dari Vue.js adalah Component. Component sendiri merupakan sebuah sistem yang memungkinkan kita untuk menambahkan atau meningkatkan fungsionalitas dari HTML. Dengan component kita dapat membungkus atau mengenkapsulasi beberapa tag HTML menjadi sebuah tag baru yang dapat digunakan berulang-ulan (reusable) pada sebuah web application.

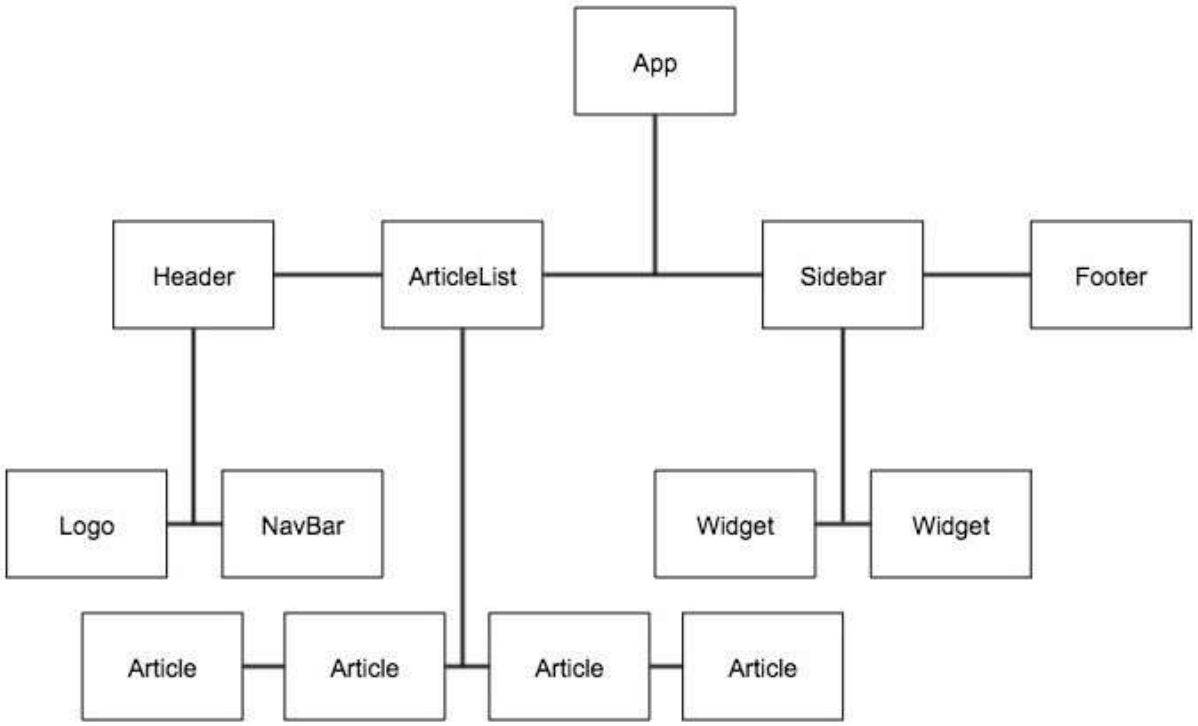
Pada umum nya di Vue.js setiap component merupakan sebuah custom element yang memiliki behavior atau sifat dari Vue compiler. Di mana pada setiap component memiliki `data/state` , `props` , `methods` , serta `lifecycle hooks` masing-masing.

Konsep Component



Pada dasarnya di dalam konsep component, setiap bagian dari sebuah aplikasi web merupakan sebuah component. Sebuah aplikasi web sendiri merupakan sebuah satu component besar yang menaungi bayak component lain. Dan sebuah component juga dapat terdiri dari banyak component lainnya. Seperti misal-nya pada gambar mock-up

halaman web di atas yang memiliki beberapa bagian seperti header, article-list, sidebar, dan footer. Di mana setiap bagian tersebut merupakan component, yang juga bisa saja terdiri dari berbagai component lain.



Terdapat konsep parent-child para web applications yang dibangun berdasarkan konsep component itu sendiri. Di mana sebuah component besar yang menaungi dan terdiri atau tersusun dari banyak component lain merupakan component parent, dan juga component kecil yang dinaungi dan merupakan bagian dari penyusun sebuah component yang lebih besar merupakan component children. Dari konsep ini dapat digambarkan sebuah component-tree yang memperlihatkan hierarki atau kedudukan masing-masing component