



Project Based Internship

Dynamic Components in Vue JS

- Memahami konsep dynamic component
- Menerapkan konsep dynamic component

Daftar Isi

A. Memahami Konsep Dynamic Component	3
B. Menerapkan Konsep Dynamic Component	3
References	10

A. Memahami Konsep Dynamic Component

Dynamic component pada Vue.js memungkinkan anda untuk menentukan mounting point dimana anda dapat beralih secara dinamis antar component dalam aplikasi anda. kita akan mencoba dan melihat beberapa contoh tentang bagaimana dynamic component bekerja.

Aplikasi memuat sekumpulan data yang mewakili array bidang. Setiap bidang memiliki jenis tertentu, opsi yang mungkin, kemungkinan default, dan banyak lagi.

Pada dasarnya, alih-alih menambahkan ke tata letak, Anda akan menambahkan . Dan ya, itu cukup banyak. Ada satu aspek penting. Sekarang setelah Anda tahu cara membuat komponen dinamis, bagaimana Anda juga akan secara dinamis meneruskan data ke dalamnya?

```
<my-component><component :is="my-component">
```

Ternyata, fitur model binding mendukung meneruskan objek dan memperluas Array kunci/nilai sebagai atribut dan nilai. sederhananya ditulis: `.v-bind="object"`

B. Menerapkan Konsep Dynamic Component

Case yang akan kita angkat kali ini untuk menerapkan dynamic component adalah sebuah dashboard sederhana yang memiliki dua buah halaman, dimana dynamic component tersebut akan diterapkan saat peralihan antar halaman. Masing masing halaman tersebut akan memiliki sebuah

component, maka untuk membuatnya, silahkan buat sebuah file, misalnya: index.html dan masukkan kerangka berikut:

```
<!DOCTYPE html>
<html>
<head>
<title>Dynamic Component - Daengweb</title>
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdFjh4u" crossorigin="anonymous" />
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap-theme.min.css" integrity="sha384-rHyoN1iRrsV4V4nD09utlInGAsiCJ3u7uwjdsM95VrLvYooPP2bMYpgeqJ0IXel/Sp" crossorigin="anonymous" />
<link rel="stylesheet" type="text/css" href="https://maxcdn.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css" />
</head>
<body>
<div id="app"></div>

<script type="text/javascript" src="https://unpkg.com/vue@2.5.6/dist/vue.js"></script>
<script type="text/javascript">
  new Vue({
    el: '#app'
  });
</script>
</body>
</html>
```

Selanjutnya kita buat navbar sederhana yang nantinya berfungsi untuk menempatkan link kedua page tersebut. Masukkan code berikut:

```
<nav class="navbar navbar-default">
  <div class="container-fluid">
    <div class="navbar-header">
      <a class="navbar-brand" href="#">BTPN Syariah</a>
    </div>
    <ul class="nav navbar-nav">
      <li>
        <a href="#">Postingan</a>
      </li>
      <li>
        <a href="#">Buat Postingan</a>
      </li>
    </ul>
  </div>
</nav>
```

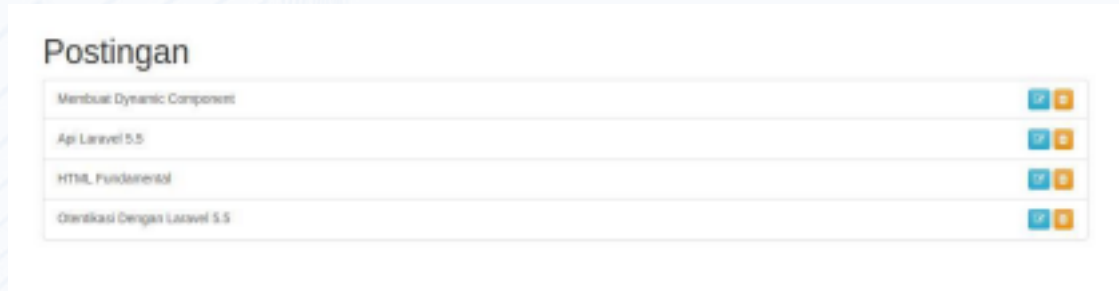
Pada tahap ini kita akan membuat component untuk kedua page component pertama berisi list postingan dan component kedua berisi form untuk membuat postingan. Mari kita buat secara bertahap.

```
<template id="postingan">
  <div>
    <h1>Postingan</h1>
    <div class="list-group">
      <a v-for="post in posts" href="#" class="list-group-
item clearfix">
        {{ post }}
        <span class="pull-right">
          <button class="btn btn-xs btn-info">
            <span class="fa fa-edit"></span>
          </button>
          <button class="btn btn-xs btn-warning">
            <span class="fa fa-trash"></span>
          </button>
        </span>
      </a>
    </div>
  </div>
</template>
```

Lalu definisikan template diatas menggunakan Vue.component:

```
Vue.component('postingan', {
  template: '#postingan',
  data: function() {
    return {
      posts: [
        'Membuat Dynamic Component',
        'Api Laravel 5.5',
        'HTML Fundamental',
        'Otentikasi Dengan Laravel 5.5'
      ]
    }
  }
});
```

Hasil yang keluar di browser akan menjadi seperti berikut:



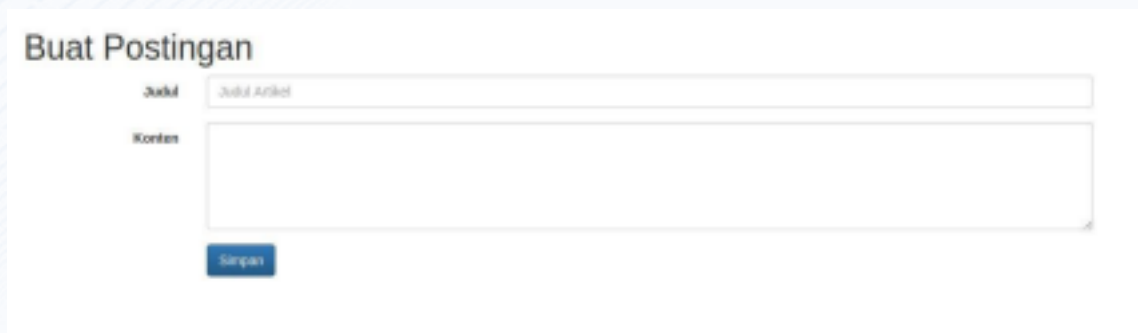
Component selanjutnya adalah buat postingan, buat template terlebih dahulu:

```
<template id="buat-postingan">
  <div>
    <h1>Buat Postingan</h1>
    <form class="form-horizontal">
      <div class="form-group">
        <label class="col-sm-2 control-label">Judul</label>
        <div class="col-sm-10">
          <input type="text" class="form-control" placeholder="Judul Artikel">
        </div>
      </div>
      <div class="form-group">
        <label class="col-sm-2 control-label">Konten</label>
        <div class="col-sm-10">
          <textarea class="form-control" rows="5"></textarea>
        </div>
      </div>
      <div class="form-group">
        <div class="col-sm-offset-2 col-sm-10">
          <button type="submit" class="btn btn-primary">Simpan</button>
        </div>
      </div>
    </form>
  </div>
</template>
```

Jangan lupa untuk mendefinisikan template diatas:

```
Vue.component('buat-postingan', {
  template: '#buat-postingan'
});
```

Maka hasil yang tertera yaitu:



Setelah kedua component tersebut dibuat, langkah selanjutnya adalah mengaitkan antar component agar bertindak secara dinamis tergantung halaman yang sedang diakses oleh user. Ketika user mengklik postingan maka kita akan memberitahu Vue untuk me-render component postingan. Demikian pula, ketika user mengklik Buat postingan maka Vue akan me-render component buat-postingan.

Untuk melakukannya, kita dapat menggunakan elemen `<component>` khusus sebagai mounting point pada root Vue instance dan menentukan component mana yang akan kita render dengan menggunakan atribut `is`. Maka tambahkan elemen berikut tepat berada markup navigasi:


```
<div class="container">
  <component :is="postingan"></component>
</div>
```

Dan untuk me-render component buat-postingan kita hanya perlu mengganti value dari atribut is

```
<component :is="buat-postingan"></component>
```

Akan tetapi kita tidak akan mungkin membuat user untuk mengganti source code ketika ingin me-render halaman lain, maka kita akan membuatnya benar-benar dinamis sehingga user hanya perlu mengklik link yang terdapat di navigasi maka secara otomatis component yang akan di-render akan menyesuaikan tergantung permintaan user.

Solusinya cukup sederhana yakni kita akan menggunakan directive v-bind. Maka pada root Vue instance kita akan membuat sebuah variable pada object data, yang akan menyimpan component aktif saat ini.

```
new Vue({
  el: '#app',
  data: {
    dilihat: 'postingan'
  }
});
```

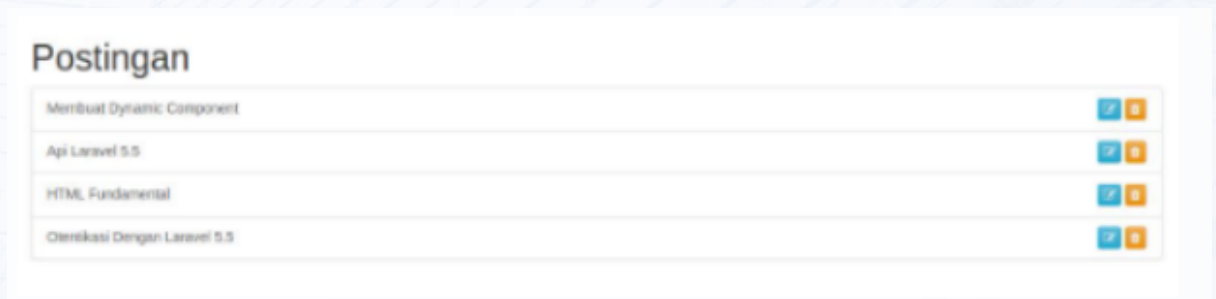

Selanjutnya pada elemen component kita akan menggunakan directive v-bind pada atribut is :

```
<component :is="dilihat"></component>
```

Langkah terakhir kita perlu menambahkan event pada link navigasi, dimana ketika user mengklik link salah satu diantara keduanya maka secara otomatis variable dilihat akan diperbaharui datanya, sehingga secara otomatis Vue akan me-render component yang dituju.

```
<li>
  <a href="#" @click="dilihat='postingan'">Postingan</a>
</li>
<li>
  <a href="#" @click="dilihat='buat-postingan'">Buat Postingan</a>
</li>
```

Adapun tampilannya secara keseluruhan akan tampak seperti ini:



References

<https://daengweb.id/vuejs-dynamic-component>

<https://www.raymondcamden.com/2018/10/31/working-with-dynamiccomponents-in-vuejs>