



Hafid Mukhlisin  
Muhammad Azamuddin

2.6

FRONTEND WEB SERIES

# Vue JS

The progressive Javascript framework

---

KUNGFU KODING

# Lisensi & Informasi Versi

---

## Lisensi

Ebook ini hanya boleh digunakan oleh pemilik email yang tertera di header buku ini. Penggunaan buku oleh selain pemilik email tersebut merupakan tindakan yang tidak diperbolehkan.

Siapapun (termasuk pemilik buku) tidak memiliki hak untuk menyalin dan atau menyebarkan buku ini tanpa seizin penulis.

## Versi ebook

versi	tanggal	author	keterangan
1.0.0	10 Oktober 2018	Hafid Mukhlisin	Rilis pertama
1.0.1	10 November 2018	Hafid Mukhlisin	Rilis kedua
1.0.2	18 November 2018	Hafid Mukhlisin	Rilis ketiga
2.0.0	4 Agustus 2019	Hafid Mukhlisin	Rilis keempat

# Persembahan

---

Bismillahirrahmanirrahim. Ucapan tanpa batas untuk Yang Maha Kuasa, Allah SWT atas setiap nafasku dan keberkahanNya. Shalawat serta salam bagi junjunganku, Nabi Muhammad SAW atas teladannya.

Terima kasih kepada Bapak dan Ibu penulis, atas cinta dan doa tulus yang tak pernah putus. Terima kasih juga kepada penulis sampaikan kepada istri tercinta, Hari Dwipanjayani, yang telah sabar menemani penulis dalam menghabiskan sisa umur ini. Tentu saja kepada keempat anak-anak penulis yang telah menjadi penyejuk hati, Ammar, Faqih, Syamil, dan Hilyah.

Buku ini juga saya persembahkan kepada mereka yang telah menginspirasi penulis yaitu Irfan Maulana, Peter Jack Kambey, Mulia Nasution, Yohan Totting, Adib Firman serta seluruh komunitas Vue JS Indonesia, Laravel Indonesia, Yii Framework Indonesia dan WWWID yang tidak bisa penulis sebutkan satu persatu.

Dan juga tentu saja buku ini saya persembahkan tidak lain tidak bukan untuk Anda para pembaca dan komunitas TI Indonesia, semoga ilmu yang sedikit ini dapat bermanfaat bagi kemajuan ilmu pengetahuan dan teknologi di Indonesia. Amiin.

# Kata Pengantar

---

Puji syukur kehadiran Allah Subhanahu Wata'ala atas limpahan nikmatnya sehingga buku panduan belajar Vue JS ini dapat diselesaikan dengan baik. Buku ini merupakan bagian dari paket buku "Be Fullstack Developer" yang ditulis bersama dengan rekan penulis yaitu Muhammad Azamuddin.

Tidak lupa penulis ucapkan terima kasih kepada berbagai pihak baik yang telah membantu kami secara langsung atau tidak langsung dalam proses penyusunan buku ini, diantaranya om Peter Jack Kambey (PHP Indonesia), om Irfan Maulana (Vue JS Indonesia) dan om Fachruzi Ramadhan (Laravel Indonesia).

Buku ini membahas Vue JS mulai dari dasar hingga membuat aplikasi berbasis Vue dan interaksinya dengan backend (Laravel web service). Untuk memudahkan dalam memahami materi dalam buku ini maka penulis juga melengkapinya dengan studi kasus yaitu membuat toko online berbasis mobile web.

Akhirnya penulis menyadari bahwa dalam penyusunan buku ini tak lepas dari kekurangan disana sini, oleh karenanya penulis memohon kritik saran dan masukan demi perbaikan pada edisi berikutnya.

Jakarta 3 Oktober 2018

Hafid Mukhlisin

# Daftar Isi

---

<b>Lisensi &amp; Informasi Versi</b>	2
Lisensi	2
Versi ebook	2
<b>Persembahan</b>	3
<b>Kata Pengantar</b>	4
<b>Mengenal Vue</b>	12
Intro	12
Apa itu Vue?	12
Sejarah Vue	12
Mengapa Memilih Vue?	13
Framework Javascript Populer	13
Didukung Banyak Pustaka	15
Bukan One Man Show	15
Digunakan Perusahaan Besar	15
Mudah Dipelajari	15
Mudah Diintegrasikan dengan Pustaka Lain	15
Dukungan Official untuk Pengembangan Aplikasi Enterprise	15
Fitur Utama	15
Virtual DOM	15
Component Base	16
Template	16
Modularity	17
Reactivity	17
Routing	17
State Management	17
Development Tools	17
Instalasi & Konfigurasi	18
Hello World	18
Menguji Reaktifitas	22
Kesimpulan	23
<b>Dasar-Dasar Vue</b>	24
Intro	24
Objek Vue	24
Inisiasi Objek Vue	24
Properti el	25
Properti Data	25
Siklus Objek Vue	27
create	29
mount	32
update	33
destroy	35
Penulisan Template	36
Data Teks	36
Data Raw HTML	37
Data Attribute	37
JS Expression	38
Properti Template	39
Properti Methods, Computed, & Filters	41
Properti Methods	41
Properti Computed	41

Properti Filters .....	43
Argumen Pada Filters .....	45
Chaining Filters .....	46
Deklarasi Filters Secara Terpisah .....	47
Kesimpulan .....	47
<b>Directive</b> .....	49
Intro .....	49
Mengenal Directive .....	49
v-html .....	49
v-once .....	49
v-text .....	49
v-show .....	49
v-if .....	50
v-on .....	52
Modifier .exact .....	59
Modifier Mouse Button .....	59
v-bind .....	59
Dynamic Argument .....	61
Kesimpulan .....	62
<b>List</b> .....	63
Intro .....	63
Menampilkan Data Array .....	63
v-for Menggunakan Tag Template .....	64
v-for Menggunakan Index .....	64
Menampilkan Data Objek .....	65
Menampilkan Data Collection .....	67
Atribut Key .....	69
Membatasi v-for menggunakan v-if .....	70
Perubahan (mutation) Data Pada Array .....	71
push() & pop() .....	72
unshift() & shift() .....	72
sort() & reverse() .....	73
splice() .....	73
fungsi set pada Vue .....	76
Perubahan Data Pada Objek .....	77
Kesimpulan .....	78
<b>Form</b> .....	80
Intro .....	80
Input Binding .....	80
Text .....	82
Boolean .....	84
Array .....	85
Filtering Data List .....	87
Handling Submit Form & Validation .....	88
Validasi Data .....	91
Prepare Data Submit .....	96
Send Data To Server .....	97
Handling File Upload .....	101
Kesimpulan .....	103
<b>Component</b> .....	105
Intro .....	105
Component Dasar .....	105
Penamaan Component .....	106
Component Registration .....	106

Global Component .....	107
Local Component .....	108
Deklarasi Properti Data .....	110
Reusable Component .....	110
<b>Component Lanjutan .....</b>	<b>111</b>
Kirim Data ke Component .....	111
Directive Pada Component .....	113
Update Data Parent From Component .....	117
Two Way Data Binding on Component .....	119
Content Distribution with Slots .....	120
Fallback Slots .....	122
Penamaan Slot .....	122
Scoped Slot .....	123
Dynamic Slot Names .....	125
Named Slots Shorthand .....	125
Single File Component .....	125
Dynamic Components .....	128
Transition Effect .....	131
Mixins .....	131
Plugins .....	133
Deklarasi Plugins .....	133
Menggunakan Plugin .....	134
Kesimpulan .....	134
<b>Routing .....</b>	<b>135</b>
Intro .....	135
Features .....	135
Installation .....	135
Getting Started .....	135
Dynamic Routing .....	138
Membuat BooksComponent .....	138
Membuat BookComponent .....	140
Programmatic Navigation .....	143
Penamaan Routing .....	145
Mengirimkan Props ke Component Routing .....	146
Transitions Effect .....	147
Navigation Guards .....	147
Global .....	148
Per Route .....	148
Dalam Component .....	148
Prevent Leave Accident .....	149
Authentication Route .....	150
Kesimpulan .....	151
Intro .....	152
Mengenal State Management .....	152
Kapan Menggunakan State Management? .....	153
Pustaka State Management .....	153
Arsitektur Vuex .....	154
Instalasi Vuex .....	154
Instalasi Dev Tools .....	155
Getting Started .....	157
Mengakses Store Via Component .....	162
Getters .....	164
Mutations .....	165
Actions .....	167

Asynchronous Actions .....	168
Menangani Two Way Data Binding .....	173
Kesimpulan .....	176
<b>Scaffolding Application .....</b>	<b>178</b>
Intro .....	178
Briefing Projek .....	178
Fitur Utama Aplikasi .....	178
Unified Model Language .....	179
Use Case Diagram .....	179
Activity Diagram .....	180
Class Diagram .....	181
Desain Database .....	181
Preparing Project .....	183
Command Line Tools .....	183
Package Manager for JS .....	184
Instalasi NodeJS & NPM .....	184
Instalasi Vue melalui NPM .....	186
Apa itu Bundler .....	188
Instalasi Vue menggunakan JS Bundler .....	189
Browserify .....	189
Webpack .....	192
Single File Component .....	196
Web Server Development .....	199
Hot Reload .....	201
Vue Command Line Interface (CLI) .....	203
Create New Project .....	204
Create New Project on Web Base .....	210
Menambahkan Plugin Baru .....	220
Kesimpulan .....	232
<b>Web Service .....</b>	<b>233</b>
Intro .....	233
Mengenal Web Service .....	233
Definisi Web Service .....	233
Standard Web Service .....	233
Method Web Service .....	233
Cara Kerja Web Service .....	233
HTTP Response Code .....	234
Stateless pada Web Service .....	235
Persiapan Tools Pengembangan .....	235
Bahasa Pemrograman: PHP .....	235
Database Server: MySQL, MariaDB .....	236
Web Server: Nginx, Apache .....	236
Git .....	236
Package Tools .....	237
Docker .....	237
XAMPP .....	246
Homestead .....	248
Composer .....	248
Instalasi Composer .....	249
Postman .....	250
Penggunaan .....	250
Generate Dokumentasi .....	252
Laravel .....	255
Mengenal Laravel .....	256



Instalasi .....	256
Konfigurasi .....	258
Variabel Konfigurasi .....	258
Virtual Domain & Pretty URL .....	259
Struktur Direktori Aplikasi .....	261
Routing .....	262
Routing Web .....	262
Routing API (Web Service) .....	264
HTTP Verbs Method .....	264
Routing Parameter .....	266
Routing Name .....	268
Routing Group .....	268
Controller .....	271
Middleware .....	272
Rate Limiting .....	272
CORS .....	274
Multiple Middleware .....	278
Database .....	278
Konfigurasi .....	279
Migration .....	280
Seeding .....	291
Interact with Database .....	302
API Resources .....	315
Handling Error .....	323
Authentication .....	325
Konfigurasi .....	326
Get Authenticate User .....	326
Check Authenticate User .....	326
Protect Routing .....	327
Authentication Mechanism .....	327
Kesimpulan .....	337
<b>Finishing Project .....</b>	<b>338</b>
Intro .....	338
Konstanta Global .....	338
Layout Aplikasi .....	339
Layout Header .....	342
Layout Main Content .....	344
Layout Footer .....	346
Layout Sidebar .....	346
Membuat Halaman Home .....	350
Layout Halaman Home .....	351
Endpoint Random Category .....	355
Endpoint Top Book .....	358
Menghubungkan Layout Home dengan Endpoint Random Categories & Top Books .....	361
Membuat Halaman Kategori Buku .....	369
Endpoint Categories .....	369
Template & Script .....	371
Mendaftarkan Route Categories .....	374
Membuat Header Pada Parent vs Child .....	375
Membuat Halaman Buku .....	377
Endpoint Book .....	377
Template & Script .....	379
Mendaftarkan Route Books .....	381
Membuat Halaman Detail Kategori .....	382

Endpoint Detail Category .....	383
Template & Script .....	386
Mendaftarkan Route Detail Category .....	388
Membuat Halaman Detail Buku .....	389
Endpoint Detail Book .....	390
Template & Script .....	393
Mendaftarkan Route Book .....	395
Membuat Reusable Component .....	396
Membuat Component BookItem .....	398
Menggunakan Component BookItem .....	399
Implementasi State Management .....	401
Mendefinisikan State .....	401
Mendefinisikan Mutation & Action Pada State .....	402
Memanggil Action State Pada Component .....	404
Memecah State Menjadi Module Tersendiri .....	408
Membuat Component Alert .....	411
Component SnackBar .....	411
Module State Alert .....	412
Component Alert .....	413
Menampilkan Alert .....	415
Membuat Indikator Keranjang Belanja .....	416
Menambahkah Getters Count .....	416
Menggunakan Getters Count Pada Header .....	417
Membuat Halaman Pencarian Buku .....	418
Membuat Endpoint Search .....	419
Membuat Component Search .....	421
Menerapkan Component Search .....	423
Trigger Halaman Pencarian .....	424
Membuat Fitur Login .....	425
Endpoint Login .....	425
State Login / Auth .....	426
Memetakan State User Pada Component .....	427
Membuat Dynamic Component .....	427
Membuat Component Login .....	431
Menampilkan Data User Login .....	435
Membuat Fitur Logout .....	437
Endpoint Logout .....	437
Membuat Link Logout .....	438
Halaman Register .....	440
Endpoint Register .....	440
Membuat Component Register .....	442
Menampilkan Halaman Register .....	444
Halaman Keranjang Belanja .....	446
Link Keranjang Belanja .....	446
Mengupdate State Cart .....	446
Membuat Component Keranjang Belanja .....	448
Halaman Checkout .....	452
Endpoint Province & City .....	452
Endpoint Update Alamat Pengiriman .....	455
Membuat Halaman Checkout Part 1 .....	456
Routing Checkout .....	460
Menampilkan Cart Pada Component Checkout .....	464
Endpoint Couriers .....	466
Endpoint Courier Services .....	467

Menampilkan Form Courier Pada Component Checkout .....	474
Endpoint Payment .....	478
Update Tombol Pay Pada Component Checkout .....	484
Halaman Pembayaran .....	488
Integrasi Payment Gateway (Experimental) .....	491
Apa itu Payment Gateway? .....	491
Persiapan Integrasi .....	492
Registrasi Midtrans .....	496
Integrasi Midtrans .....	499
Membuat Halaman Profile .....	503
Link Profile di SideBar .....	503
Layout Halaman Profile .....	504
Membuat Halaman Histori Belanja .....	506
Endpoint My Order .....	506
Link My Order di SideBar .....	507
Layout Halaman My Order .....	508
Source Code .....	511
Kesimpulan .....	511
<b>Deployment</b> .....	512
Intro .....	512
Diskon Hosting & VPS .....	512
Persiapan .....	512
Persiapan Aplikasi Web Frontend .....	513
Persiapan Aplikasi Web Service .....	513
Konfigurasi .....	514
Matikan Mode Debug .....	514
Proses Deployment .....	516
Deployment Aplikasi Web Frontend .....	516
Deployment Aplikasi Web Service .....	523
Membuat Sub Domain .....	523
Membuat & Mengimport Database .....	525
Mengunggah File Aplikasi .....	530
Kesimpulan .....	538

# Mengenal Vue

---

## Intro

Pada bab ini, kamu akan diajak mengenal Vue, mengapa memilih Vue, tools apa saja yang diperlukan dan bagaimana cara menggunakannya secara mudah dan sederhana.

## Apa itu Vue?

Vue (dibaca: view) merupakan salah satu dari sekian banyak pustaka (library) pada bahasa pemrograman Javascript yang digunakan untuk membangun tampilan antarmuka pengguna (user interface) dari suatu aplikasi berbasis web khususnya untuk aplikasi berbasis halaman tunggal atau *single page application* (SPA).

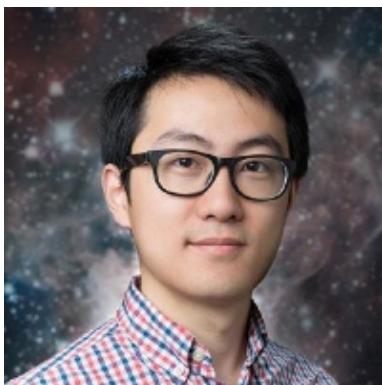


Vue sebagaimana Javascript (JS) memang awalnya didesain untuk kebutuhan web, namun seiring perkembangan teknologi yang mendukung JS, maka saat ini Vue juga mulai dapat digunakan untuk mengembangkan aplikasi berbasis desktop dan mobile.

Situs resmi Vue bisa kita jumpai pada alamat <http://vuejs.org>, adapun link githubnya pada alamat <https://github.com/vuejs>.

## Sejarah Vue

Awalnya, Vue merupakan proyek pribadi Evan You (<http://evanyou.me>) ketika masih bekerja di Google Creative Labs pada tahun 2013. Di sana, Evan terlibat dalam pembuatan berbagai prototipe tampilan antarmuka pengguna menggunakan pustaka AngularJs (versi 1).



Hal inilah yang kemudian menginspirasi Evan untuk membuat suatu pustaka sendiri dengan gaya Angular namun menggunakan pendekatan API (Application Programming Interface) yang lebih sederhana.

Vue mengusung konsep web component dan virtual DOM sebagaimana React namun dengan pendekatan yang lebih natural, siapapun yang telah mengenal dasar HTML, Javascript & CSS akan dengan mudah dan

cepat dalam menguasai serta mengadopsi Vue.

Pada Februari 2014, Vue pertama kali dipublikasikan dan langsung mendapatkan sambutan yang luar biasa pada minggu pertamanya, hingga membuat Evan terpacu untuk lebih serius lagi dalam mengembangkannya.

Oktober 2015, Vue versi 1.0 dipublikasikan yang menandakan Vue siap digunakan untuk production. Diawal tahun 2016, Evan mulai bekerja penuh waktu untuk mengelola Vue berkat banyaknya dukungan atau sponsor yang ia dapat melalui situs Patreon (<https://www.patreon.com/evanyou>).

Salah satu sponsor utama yang sekaligus meningkatkan branding dari Vue adalah Taylor Otwell (<http://taylorotwell.com>), founder Laravel PHP Framework (<https://laravel.io>) dimana kemudian menjadikan Vue sebagai pustaka Javascript untuk Laravel.

Saat buku ini diupdate, Vue telah mencapai versi 2 dengan berbagai perbaikan dan penambahan fitur baru. Versi ini menggunakan engine berbeda untuk menangani Virtual DOM namun tetap ringan dan cepat, kita bisa menggunakan template HTML atau JSX sebagaimana yang umum dipakai di React. Manajemen state didukung secara official melalui Vuex, dan secara natural telah mendukung server side rendering.

Meskipun demikian, secara umum Vue masih tetap menjaga kompatibilitas dengan versi sebelumnya.

## Mengapa Memilih Vue?

Jika dilihat dari karakteristik penggunaanya, React menarik bagi mereka yang menyukai functional programming. Angular menarik bagi developer yang terbiasa bermain di bahasa pemrograman Java atau C#. Sedangkan Vue menarik bagi mereka yang menyukai classic HTML, CSS & JavaScript. Hal inilah yang menjadikan alasan mengapa Vue banyak mengambil hati para web developer (termasuk penulis 😊).

Berikut ini, penulis akan mencoba merangkum tentang beberapa alasan mengapa banyak developer memilih Vue.

### Framework Javascript Populer

Tidak diragukan lagi, Vue merupakan framework Javascript modern yang cukup populer disamping React & Angular.

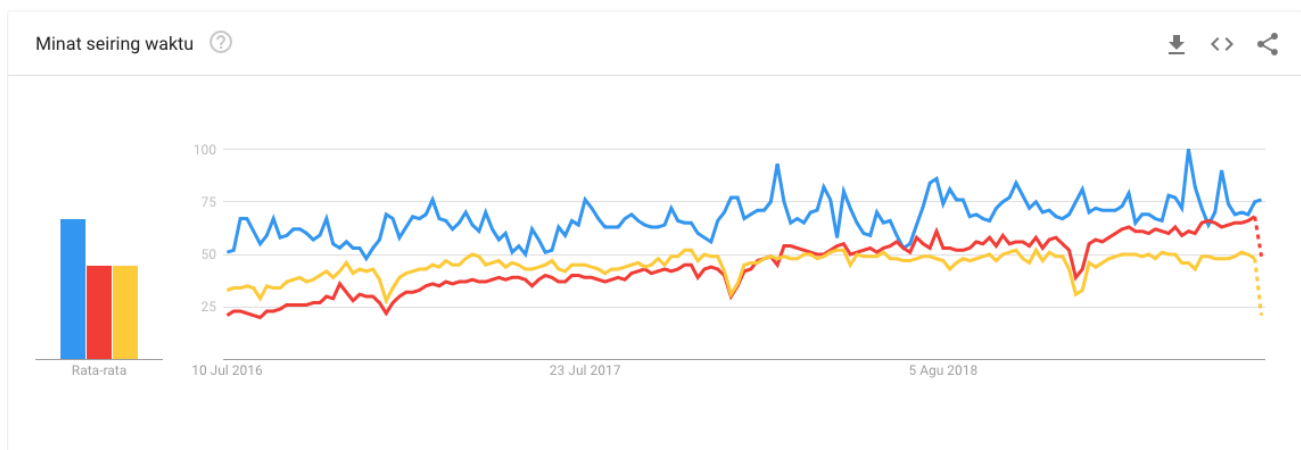


Berdasarkan data dari Github (Juli 2019), jumlah star (bintang) di akun Githubnya (<https://github.com/vuejs/vue>) mencapai lebih dari 140 ribu user dan di-fork oleh sekitar 20 ribu user, meski jumlah ini sebelas duabelas dengan perolehan React dan jauh di atas Angular (js & io)

The image displays three screenshots of GitHub repository pages, stacked vertically. Each screenshot shows the repository name, navigation links (Sponsor, Watch, Star, Fork), and a list of repository statistics (Issues, Pull requests, Projects, Wiki, Security, Insights). The first screenshot is for 'vuejs / vue', the second for 'facebook / react', and the third for 'angular / angular.js'. Each page also features a search bar and 'Sign in' / 'Sign up' buttons.

Repository	Watch	Star	Fork
vuejs / vue	5,939	143,253	20,723
facebook / react	6,650	132,333	24,516
angular / angular	3,269	49,489	13,556
angular / angular.js	4,194	59,568	28,878

Namun, berdasarkan data dari Google Trends, pencarian terkait Vue selama tiga tahun terakhir jauh melampaui dua rivalnya tersebut.



Sumber: <https://trends.google.com/trends/explore?date=2016-07-10%202019-07-10&q=vue,react,angular>

Tentu saja hal ini hanyalah salah satu parameter atau tolok ukur dari kepopuleran suatu pustaka atau framework.

## Didukung Banyak Pustaka

Salah satu kelebihan dari Vue ini adalah didukung oleh banyak pustaka, sehingga cukup memudahkan bagi developer untuk bekerja dengan Vue. Berbagai pustaka yang mendukung dan menggunakan Vue bisa kita jumpai pada tautan ini <https://github.com/vuejs/awesome-vue>.

## Bukan One Man Show

Saat ini Vue sudah mencapai versi 2, bukan lagi proyek pribadi, sebab core developer-nya sudah terdiri dari belasan orang (<https://vuejs.org/v2/guide/team.html>), belum lagi kontributornya di Github yang cukup banyak.

## Digunakan Perusahaan Besar

Tidak hanya digunakan oleh perorangan, beberapa perusahaan atau web besar juga telah menggunakan Vue diantaranya: Adobe, Alibaba, Xiaomi, Line, Nintendo, Gitlab, Laravel dsb, selengkapnya di <https://madewithvuejs.com>.

## Mudah Dipelajari

Pendekatan yang ditawarkan Vue cukup sederhana dan tidak banyak memperkenalkan konsep baru, sehingga siapapun dengan latar belakang pengetahuan web dasar (HTML, CSS, Javascript) akan mudah menggunakan dan mengadopsi Vue.

Yap, jika kamu masih baru dalam dunia web programming, maka memang butuh usaha lebih karena tidak akan dibahas secara detail pada buku ini. Penulis berasumsi bahwa kamu sudah memiliki pengetahuan tentang itu, dan jika belum maka gunakan referensi lain terkait web dasar.

## Mudah Diintegrasikan dengan Pustaka Lain

Jika kamu sudah menggunakan pustaka lain pada aplikasi saat ini maka kamu tidak perlu khawatir untuk mengintegrasikannya dengan Vue. Apakah kamu tetap ingin menggunakan JQuery misalnya, maka itu tidak menjadi masalah berarti.

## Dukungan Official untuk Pengembangan Aplikasi Enterprise

Berbeda dengan React, Vue mendukung dan mengembangkan sendiri secara resmi pustaka-pustaka yang digunakan untuk membangun aplikasi skala besar, seperti routing (Vue Router), state management (Vuex), server side rendering, dsb. Namun hal ini tidak membuat kita sulit untuk menggunakan pustaka lain yang mungkin biasa kita gunakan, seperti Redux, Mobx, dsb.

## Fitur Utama

Berikut ini beberapa fitur utama yang dimiliki Vue.

### Virtual DOM

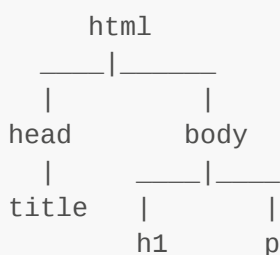
DOM singkatan dari Document Object Model merupakan model yang menggambarkan halaman HTML atau XML. DOM berbentuk struktur hirarki pohon yang menghubungkan masing-masing elemen HTML/XML (node). Contoh.

```

<html>
<head>
  <title>Contoh</title>
</head>
<body>
  <h1> Halo </h1>
  <p> Test </p>
</body>
</html>

```

Kode HTML di atas jika dilihat dari sudut pandang DOM memiliki root node `html`, node `html` memiliki dua child node yaitu `head` dan `body`. Node `head` memiliki satu child yaitu `title`, sedangkan node `body` memiliki dua child yaitu `h1` dan `p`.



Javascript memiliki kemampuan untuk mengakses dan memanipulasi semua DOM tersebut secara langsung.

```

const h1s = Array.from(document.getElementsByTagName('h1'))

console.log(h1s[0]); // <h1> Halo </h1>

```

Namun alih-alih memanipulasinya secara langsung, Vue memilih pendekatan lain yaitu membuat abstraksi objek virtual dari DOM kemudian memanipulasinya baru kemudian merender atau menampilkan hasilnya. Pendekatan ini lebih efektif dan cepat dibandingkan langsung memanipulasi DOM-nya sebagaimana yang dilakukan pustaka lain semisal JQuery.

## Component Base

Vue menggunakan pendekatan berbasis komponen, dimana setiap tampilan atau bagian dari tampilan merupakan komponen. Melalui pendekatan ini, tampilan yang kompleks dapat dipecah menjadi beberapa bagian dan setiap bagian itu bisa digunakan kembali pada bagian lainnya. Hal ini akan membuat kode lebih efisien dan bersih. Kode komponen pada Vue ditulis menggunakan kode Javascript sebagai sebuah object.

## Template

Berkaitan dengan poin sebelumnya, template merupakan kode yang dijadikan dasar dari suatu komponen dan biasanya berupa kode-kode HTML biasa. Penulisan template pada Vue bisa sangat fleksibel dan out of the box. Kita bisa tuliskan suatu template menjadi satu dengan kode komponennya seperti React, atau



dipisahkan menggunakan tag template tag HTML yang id-nya telah didaftarkan, bisa juga dipisahkan pada file tersendiri yang umumnya menggunakan ekstensi Vue, dsb.

## Modularity

Komponen pada Vue bisa dipecah menjadi modul-modul kecil. Hal ini akan memudahkan developer dalam pengembangan atau pengelolaan kodenya terutama pada proyek aplikasi skala besar.

## Reactivity

Secara default, Vue mendukung reactivity yaitu perubahan data pada suatu bagian tertentu akan secara interaktif mempengaruhi bagian yang lain. Fitur ini akan memudahkan developer dalam mengembangkan aplikasi karena cukup dengan fokus pada flow data dan template.

## Routing

Routing merupakan kebutuhan untuk pembuatan aplikasi enterprise karena menyangkut bagaimana suatu halaman pada aplikasi tersebut diakses oleh pengguna melalui web browser. Meski bukan pada core-nya, namun Vue menyediakan pustaka yang didukung secara resmi untuk menangani routing aplikasi, yaitu Vue router <https://router.vuejs.org>.

## State Management

Oleh karena vue berbasis komponen, maka diperlukan pendekatan terpusat untuk menyimpan state atau data aplikasi yang bisa dibaca dan dimodifikasi oleh semua komponen yang membutuhkannya. State management juga bukan core pada Vue seperti halnya routing, namun pustaka yang menangani state ini juga didukung secara resmi yaitu vuex <https://vuex.vuejs.org>.

## Develompent Tools

Sebagai sebuah pustaka Javascript biasa, untuk mengembangkan aplikasi berbasis Vue sebenarnya developer hanya membutuhkan code editor untuk menulis kode programnya, serta web browser untuk menampilkan hasilnya.

Tidak ada pilihan spesifik, silahkan gunakan code editor favoritmu, misalnya: Visual Studio Code (penulis menggunakan ini), Sublime, Netbeans, Notepad++, Intelij Idea, dsb.

Adapun untuk web browser pun juga bebas, bisa Google Chrome (penulis menggunakan ini), Mozilla Firefox, Safari, bahkan Microsoft IE (versi 9 atau later) 😊.

Android	Firefox	Chrome	IE	iPhone	Edge	Safari
6.0 🐙 ✓	58 🦊 7 ✓	65 🦊 7 ✓	9 🦊 7 ✓	10 🍏 10.12 ✓	16 🦊 10 ✓	8 🍏 10.10 ✓
			10 🦊 8 ✓			
			11 🦊 8.1 ✓			

Ya, untuk fase awal ini, dua tools ini dulu yang harus kamu siapkan dan penulis yakin semua itu sudah tersedia di komputermu. Sebenarnya banyak tools lain yang perlu juga digunakan namun secara bertahap saja ya 😊, sebab penulis tidak ingin kamu pusing di awal, khawatir kalah sebelum berperang.

Penulis juga ingin menunjukkan kepadamu tentang seberapa sederhana Vue, awalnya sih 😊.  
Bagaimana? sepakat?

## Instalasi & Konfigurasi

Sebagai sebuah pustaka JS, maka kita perlu menambahkannya ke dalam halaman HTML kita sebelum kita menggunakannya. Saat buku ini ditulis, versi terbaru Vue adalah 2.6.10 (Juli 2019). Untuk melihat versi terbaru dan sebelumnya, silahkan kunjungi tautan berikut <https://github.com/vuejs/vue/releases>.

Pustaka Vue terbagi menjadi dua, yaitu mode development (filenya tidak dikompres) dan mode production (file dikompres). Sangat disarankan menggunakan mode development saat mengembangkan aplikasi menggunakan Vue sebab semua informasi umum (warning) jika terjadi kesalahan kode akan dimunculkan.

File Vue yang akan kita tambahkan ke dalam halaman HTML bisa kita unduh ke lokal (sehingga tidak membutuhkan koneksi internet lagi) atau ditautkan langsung dengan server pustaka Vue (CDN). Kita bisa mengunduh pustaka Vue versi development pada tautan berikut <https://vuejs.org/js/vue.js>, Adapun versi production bisa kita jumpai pada tautan ini <https://vuejs.org/js/vue.min.js>.

Sebagaimana umumnya pustaka javascript, untuk menambahkan ke halaman HTML kita maka cukup dengan kode berikut.

```
<script src="lib/vue.js"></script>
```

Jika kita memilih menautkan langsung ke server, maka kita bisa gunakan tautan ini <https://cdn.jsdelivr.net/npm/vue@2.6.10/dist/vue.js>

Catatan: sesuaikan dengan versi saat ini. Silahkan cek versi yang tersedia pada tautan ini <https://cdn.jsdelivr.net/npm/vue>

```
<script src="https://cdn.jsdelivr.net/npm/vue@2.6.10/dist/vue.js"></script>
```

Catatan: pastikan bahwa ketika aplikasi akan dilaunching (production) maka ubah vue.js menjadi vue.min.js untuk tujuan keamanan dan performa. Namun untuk pengembangan, tetap disarankan menggunakan mode development.

Untuk mengembangkan aplikasi skala besar, maka instalasi Vue disarankan dengan menggunakan package manager seperti NPM (penulis menggunakan ini) atau YARN. Disamping itu, Vue membuat tools CLI <https://cli.vuejs.org> yang akan memudahkan kita dalam membuat *scaffolding* proyek aplikasi (manajemen kode & tools serta konfigurasi saat pengembangan aplikasi). Topik ini akan dibahas tuntas pada bagian berikutnya.

## Hello World

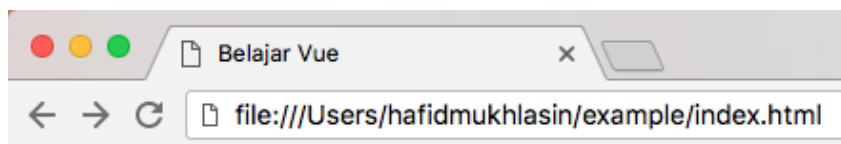
Cara klasik belajar untuk mulai belajar suatu bahasa pemrograman atau pustaka baru adalah dengan cara membuat kode untuk menampilkan teks "hello world" menggunakan bahasa atau pustaka tersebut. Apabila

kita bisa membuatnya maka konon selanjutnya akan lebih mudah. Cara ini akan kita gunakan untuk mengawali belajar Vue pada buku ini.

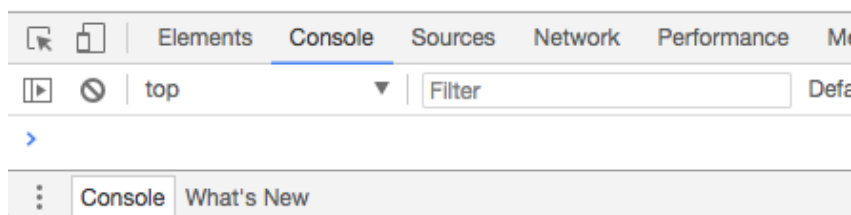
Disini, kita akan buktikan seberapa natural Vue bagi kamu yang sudah terbiasa dengan HTML, CSS & JS. Mari kita mulai dengan membuat file HTML dengan nama index.html (tentu saja kamu boleh menggunakan nama lain) sebagaimana kode HTML pada umumnya.

```
<!DOCTYPE html>
<html>
<head>
  <title>Belajar Vue</title>
</head>
<body>
  <div id="app">
    <h1>Hello world</h1>
  </div>
</body>
</html>
```

Kemudian jalankan file ini pada browser, maka hasilnya sebagai berikut.



# Hello world



Maka pada browser akan muncul teks "Hello world". Apakah ini cukup natural? Mudah sekali bukan? Oh bukan, penulis hanya bercanda, itu bukan Vue, itu hanya HTML biasa 😊. Karenanya, mari kita ubah kode di atas (di dalam tag HTML body) menjadi sebagai berikut.

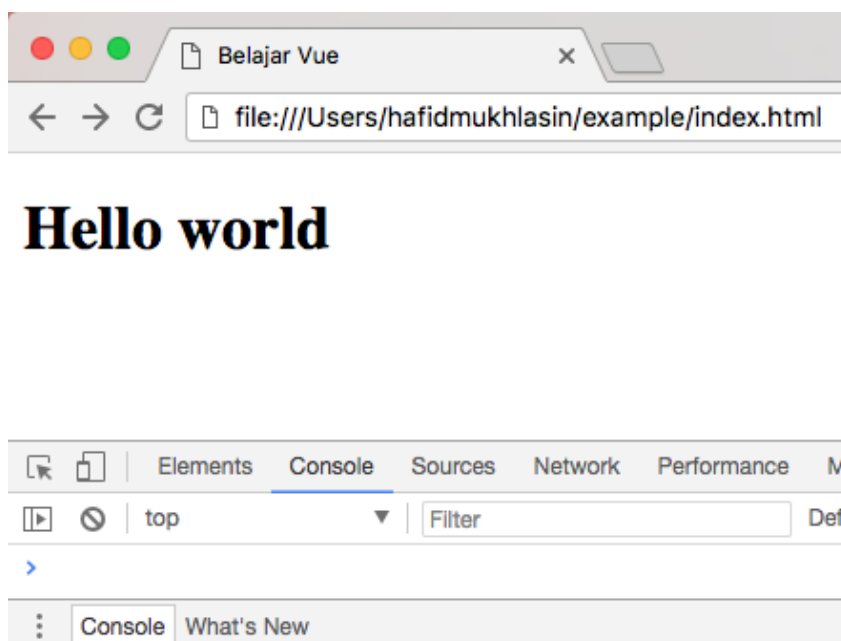
```
<!DOCTYPE html>
<html>
<head>
  <title>Belajar Vue</title>
  <script src="lib/vue.js"></script>
</head>
```

```

<body>
  <div id="app">
    <h1>{{ message }}</h1>
  </div>
  <script type="text/javascript">
    var vm = new Vue({
      el: '#app',
      data: {
        message: 'Hello world!'
      }
    })
  </script>
</body>
</html>

```

Kode di atas akan menghasilkan tampilan sebagai berikut.

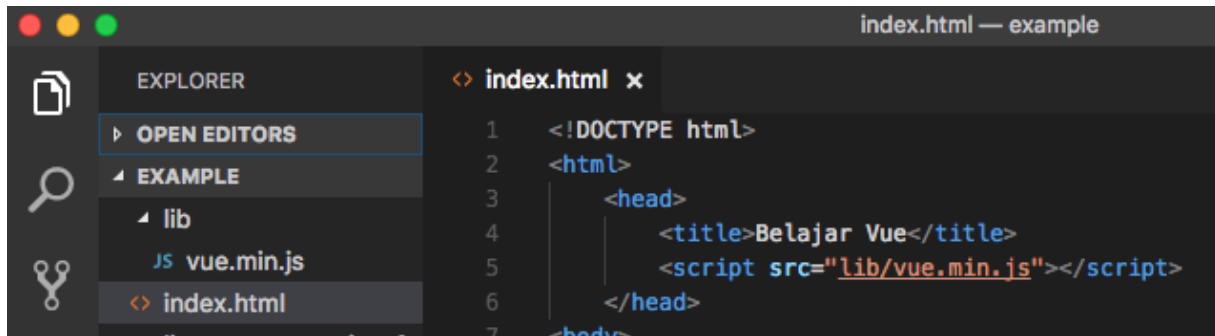


Wah kok ribet sekali? Hanya untuk menuliskan HTML di web browser kodenya sepanjang itu! Apa kelebihannya?

Baik, pada contoh ini memang tidak ada kelebihannya, bahkan tidak disarankan untuk menggunakan kode ini jika hanya untuk menampilkan teks statis pada browser. Namun dari kode sederhana ini, kita akan belajar tentang bagaimana Vue tersebut bekerja.

- Pertama Kita butuh HTML untuk menjalankan kode-kode Vue, karena kita tahu bahwa Vue hanyalah sebuah pustaka Javascript yang tugasnya memanipulasi tampilan HTML.
- Kedua Kita perlu menambahkan (include) pustaka Vue ke HTML sebagaimana yang telah dijelaskan pada bagian **Instalasi** karena Vue merupakan pustaka Javascript

```
<script src="lib/vue.js"></script>
```



Catatan: pustaka Vue tidak harus diletakkan di dalam elemen head, bisa juga di dalam body.

- Ketiga Kita perlu membuat kontainer (mount point) berupa elemen HTML, untuk menandai bahwa di dalam elemen tersebut nantinya hasil kompilasi Vue akan ditampilkan atau dimuat. Sebagai penanda, kita perlu tambahkan atribut id pada tag tersebut yang nantinya akan didefinisikan pada saat inisiasi objek Vue.

```
<div id="app">
  ...
</div>
```

Catatan: Nilai dari atribut id tidak harus `app`, bebas saja, tergantung definisi di saat inisiasi objek Vue.

- Keempat Kita perlu menggunakan double kurung kurawal (mustache) untuk menandai bahwa teks tersebut merupakan variabel yang akan dimanipulasi oleh Vue, model seperti ini telah umum digunakan diberbagai template engine.

```
<div id="app">
  <h1>{{ message }}</h1>
</div>
```

Di samping itu, kita juga bisa menggabungkannya dengan teks statis.

```
<h1>Pesan: {{ message }}</h1>
```

atau menggunakan operasi Javascript untuk menggabungkan dua teks (string) tersebut.

```
<h1>{{ 'Pesan: ' + message }}</h1>
```

- Kelima Kita perlu membuat instance/objek baru untuk class Vue, yang tentunya ditulis dengan menggunakan Javascript.

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

Objek Vue yang dibuat ini disimpan ke dalam variabel bernama `vm` (nama bebas) untuk memudahkan kita nantinya dalam mengakses objek ini. Objek Vue pada kode di atas menggunakan dua properti yaitu `el` dan `data`. Properti `el` menunjukkan id dari elemen HTML yang akan dijadikan sebagai target atau tempat ditampilkannya hasil manipulasi data dan template.

Di samping itu bisa juga kita gunakan perintah `vm.$mount('#app')` untuk mengarahkan mount point Vue pada saat runtime.

```
var vm = new Vue({
  data: {
    message: 'Hello world!'
  }
})

vm.$mount('#app')
```

Properti berikutnya adalah `data` yang berbentuk objek, dimana di dalamnya terdapat key `message` dengan nilai 'Hello world!' yang merupakan representasi dari variabel. Key atau variabel dalam properti `data` inilah yang akan mengubah kode template `{{ message }}` (lihat poin keempat) menjadi teks "Hello world!". Dengan kata lain, jika kita mengubah nilai dari variabel `message` ini menjadi misalnya "Hello Vuejs!" maka tentu tampilan yang kita lihat pada browser juga akan berubah sesuai teks tersebut. Sederhana sekali bukan?

Mungkin kamu melihatnya sederhana, namun Vue telah melakukan hal yang mungkin lebih dari yang kamu bayangkan. Vue diam-diam telah menghubungkan antara DOM dengan data atau variabel, dimana sekarang keduanya menjadi reaktif.

Catatan: kita bisa mengakses properti dan variabel dalam objek `vm` tersebut, misalnya untuk mengakses properti `el` atau `data` maka kita bisa gunakan perintah `vm.$el` atau `vm.$data`. Adapun untuk mengakses variabel `message` dalam properti `data` kita bisa gunakan perintah `vm.$data.message` atau langsung `vm.message` (tanpa tanda dollar).

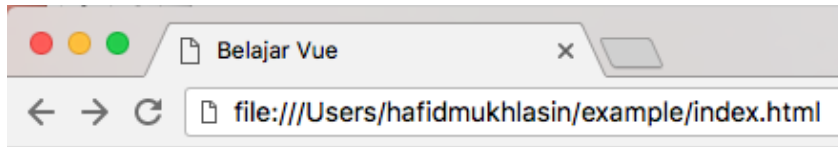
## Menguji Reaktifitas

Pada bagian awal ini, mari kita bereksperimen untuk menguji sifat reaktif dari Vue dengan cara yang sederhana. Kita akan menggunakan console pada browser (penulis menggunakan Chrome).

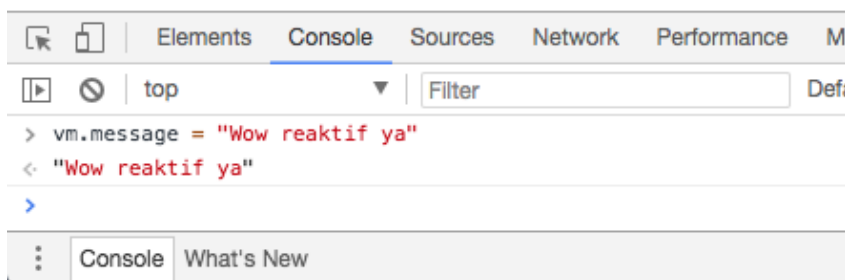
Pada console, mari kita ubah variabel `message`, dengan menggunakan perintah berikut berikut:

```
vm.message = "Wow reaktif ya"
```

Tekan enter dan lihat apa yang terjadi?



# Wow reaktif ya



Yap, tanpa refresh maka seketika itu juga teks yang muncul di halaman browser berubah sesuai dengan teks yang kita sematkan pada variabel message di console.

Pada kondisi nyata, tentu saja perubahan data tidak dilakukan dengan menggunakan console pada browser melainkan melalui cara-cara yang natural yaitu menggunakan perintah JS yang dijalankan misalnya melalui event onclick button, input dari user, dsb.

## Kesimpulan

Vue merupakan pustaka JS yang bekerja memanipulasi elemen HTML menggunakan teknik virtual DOM HTML sehingga lebih cepat prosesnya dibandingkan langsung memanipulasi DOM-nya. Untuk menggunakannya pada aplikasi, maka pustaka Vue cukup diincludekan menggunakan element HTML script sebagaimana umumnya pustaka JS.

Sekarang kita telah mengenal Vue dan bagaimana cara kerja Vue. Penulis berharap, kamu benar-benar memahami apa yang telah kita bahas pada bab ini, jika belum maka sebaiknya kamu mengulangnya lagi dan sekali lagi hingga benar-benar faham.

Untuk memperkuat pemahaman kamu, maka pada bab selanjutnya, akan dibahas mengenai dasar-dasar Vue yang tentunya lebih dari sekedar hello world 😊.

Bagaimana? Penasarankan?

# Form

---

## Intro

Pada bab ini kita akan belajar tentang bagaimana menangani input data dari user melalui form serta bagaimana memanipulasi tampilan datanya.

Catatan: untuk latihan pada bab ini, save as file html yang berisi kode Vue kita menjadi form.html

## Input Binding

Form HTML memiliki berbagai jenis field input seperti text, password, radio, dsb yang peruntukannya tentu berbeda tergantung dari data yang ingin ditangkap.

```
<input name="username" type="text">
<input name="password" type="password">
<input name="gender" type="radio">
```

Terkait dengan input binding ini, yang kita butuhkan adalah two way data binding, di mana nilai dari field input terhubung dengan data secara dua arah. Artinya perubahan field input yang dilakukan oleh user akan menyebabkan perubahan variabel data, sebaliknya perubahan variabel data akan menyebabkan perubahan pada field input.

Berdasarkan, bahasan tentang directive, mungkin kita segera bisa menemukan triknya. Ya, yang kita butuhkan dua directive sekaligus v-on sebagai listener perubahan field input, dan v-bind yang bertugas mem-bind perubahan variabel data untuk diterapkan pada field input.

```
<div id="app">
  <form>
    <input type="text" name="title" :value="title" @input="title =
    $event.target.value" />

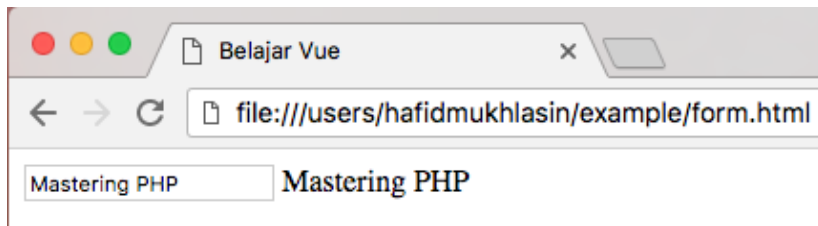
    {{ title }}
  </form>
</div>

<script>
var vm = new Vue({
  el: '#app',
  data: {
    title: "Mastering "
  },
})
</script>
```

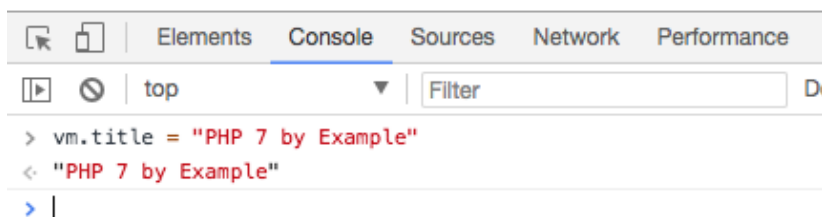
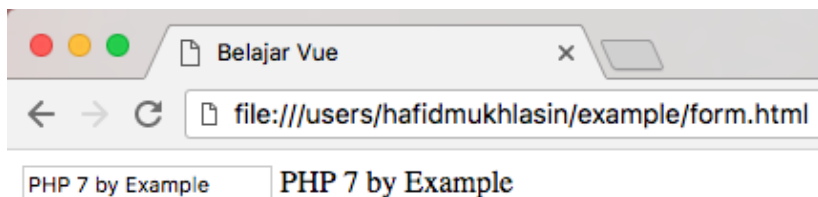


Atribut value di-*bind* dan event oninput di-*listen*. Kode `:value="title"` menunjukkan bahwa nilai dari field input ini di-*bind* dengan variabel `title`, sedangkan kode `@input="title = $event.target.value"` artinya ketika field diinput maka nilai variabel `title` akan diubah sesuai isian user.

Berikut ini hasilnya.



Ketika field input diubah maka variabel title juga berubah.



Ketika variabel title diubah via console `vm.title = "PHP 7 by Example"` maka nilai dari field input juga akan mengikuti.

Catatan: metode ini akan dipakai ketika kita bermain dengan komponen.

Karenanya, untuk mengatasi "kerumitan" ini, Vue memperkenalkan directive `v-model` yang bertugas melakukan two way data binding tersebut. Berikut ini contohnya.

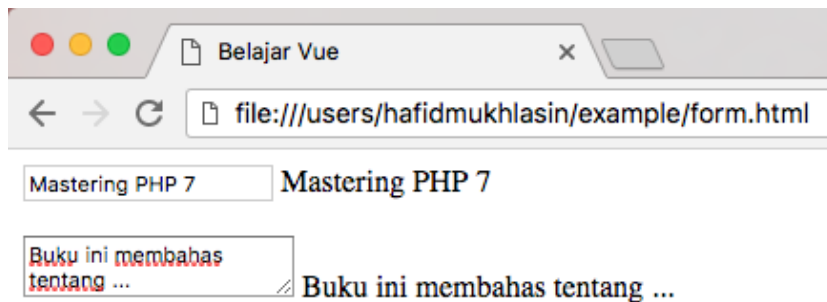
```
<form>
  <input type="text" name="title" v-model="title" placeholder="masukkan
  judul">
    {{ title }}
  <br><br>
  <textarea name="description" v-model="description"
  placeholder="masukkan deskripsi"></textarea>
    {{ description }}
</form>
```

Catatan: pastikan selalu menggunakan atribut name pada setiap field yang digunakan, disamping merupakan best practice juga akan berguna nanti pada bahasan selanjutnya. Nilai atribut name tidak harus sama dengan nilai dari atribut v-model, hanya saja karena keduanya identik maka sebaiknya disamakan saja.

Tentu saja kita perlu tambahkan dua variabel yaitu title dan description.

```
data: {  
  title: "",  
  description: ""  
},
```

Hasilnya sebagai berikut.



## Text

Pada contoh di atas, elemen field input text, password dan textarea menerima data dalam bentuk teks atau string, sehingga tipe data pada variabel datanya adalah string juga.

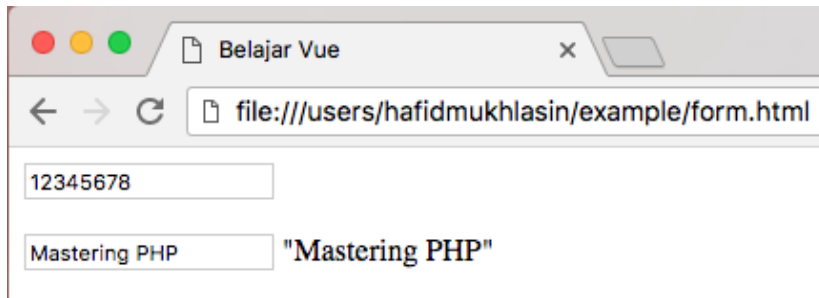
Directive v-model juga memiliki modifier, diantaranya adalah `number` dan `trim`.

Modifier `number` digunakan untuk memastikan input data dari user bertipe numeric. Sedangkan modifier `trim` digunakan untuk menghapus spasi putih diawal atau akhir dari string.

```
<input type="number" name="price" v-model.number="price">  
<br><br>  
<input type="text" name="title" v-model.trim="title" placeholder="masukkan  
judul"> "{{ title }}"
```

Catatan: tambahkan variabel price.

Mari kita lihat hasilnya.



Belajar Vue

file:///users/hafidmukhlasin/example/form.html

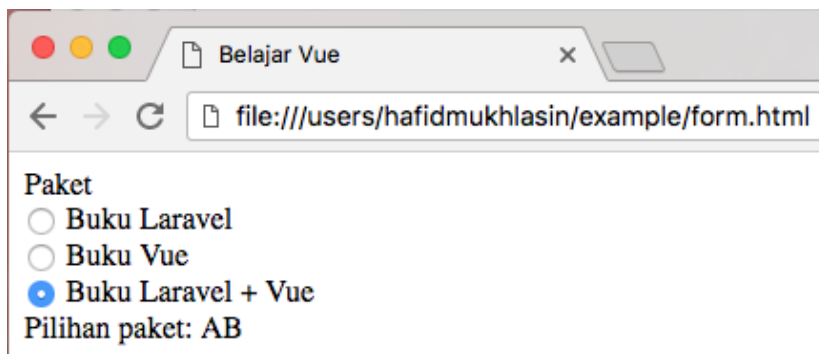
12345678

Mastering PHP "Mastering PHP"

Field input radion bertipe data teks.

```
Paket <br>
<input type="radio" name="bukuA" value="A" v-model="paket">
<label for="bukuA">Buku Laravel</label>
<br>
<input type="radio" name="bukuB" value="B" v-model="paket">
<label for="bukuB">Buku Vue</label>
<br>
<input type="radio" name="bukuAB" value="AB" v-model="paket">
<label for="bukuAB">Buku Laravel + Vue</label>
<br>
<span>Pilihan paket: {{ paket }}</span>
```

```
data: {
  paket: ''
}
```



Belajar Vue

file:///users/hafidmukhlasin/example/form.html

Paket

☐ Buku Laravel

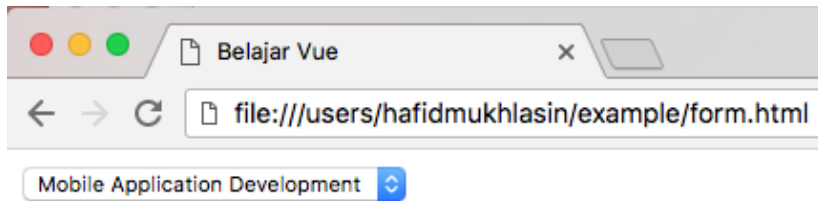
☐ Buku Vue

☒ Buku Laravel + Vue

Pilihan paket: AB

Field input select (single select) juga bertipe data teks. (tambahkan variabel category)

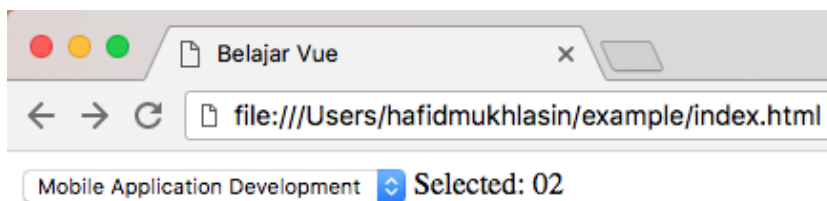
```
<select name="category" v-model="category">
  <option disabled value="">Please select one</option>
  <option>Graphics Programming</option>
  <option>Mobile Application Development</option>
  <option>Virtual and Augmented Reality</option>
</select>
```



Catatan: Jika menggunakan field select, sebaiknya menambahkan elemen option disabled dengan nilai kosong sebagaimana contoh di atas untuk mengatasi masalah pada IOS.

Pada contoh di atas, atribut value pada elemen option tidak didefinisikan sehingga nilai dari variabel category mengikuti text diantara elemen option. Namun apabila value didefinisikan maka yang digunakan adalah value tersebut.

```
<select name="category" v-model="category">
  <option disabled value="">Please select one</option>
  <option value="01">Graphics Programming</option>
  <option value="02">Mobile Application Development</option>
  <option value="03">Virtual and Augmented Reality</option>
</select>
<span>Selected: {{ category }}</span>
```

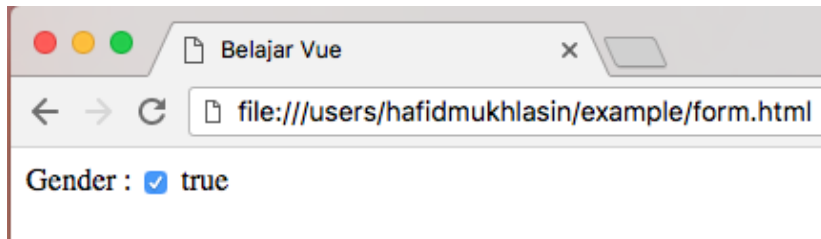


## Boolean

Input data bertipe data boolean (true atau false) biasanya terjadi pada field input checkbox tunggal. Contoh pada input data gender.

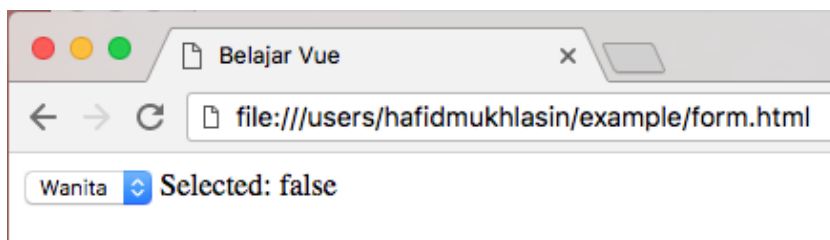
```
Gender :
<input type="checkbox" name="checkbox" v-model="gender">
<label for="checkbox">{{ gender }}</label>
```

```
data: {
  gender: true
}
```



Field input select tunggal dengan dua option juga bisa bertipe boolean. Perhatikan contoh berikut.

```
<select name="gender" v-model="gender">
  <option value="false">Wanita</option>
  <option value="true">Pria</option>
</select>
<span>Selected: {{ gender }}</span>
```



## Array

Input data bertipe array terjadi pada field input dengan kemungkinan pilihan lebih dari satu seperti checkbox multiple dan select multiple.

```
Hobi <br>
<input type="checkbox" name="hobby1" value="nonton" v-model="hobbies">
<label for="hobby1">Nonton</label>
<input type="checkbox" name="hobby2" value="jalan" v-model="hobbies">
<label for="hobby2">Jalan</label>
<input type="checkbox" name="hobby3" value="makan" v-model="hobbies">
<label for="hobby3">Makan</label>
<br>
<span>Pilihan hobi: {{ hobbies }}</span>

<hr>

<select v-model="categories" multiple>
  <option value="01">Graphics Programming</option>
  <option value="02">Mobile Application Development</option>
  <option value="03">Virtual and Augmented Reality</option>
</select>
<span>Selected: {{ categories }}</span>
```

Adapun definisi dari variabel data harus sebagai array.

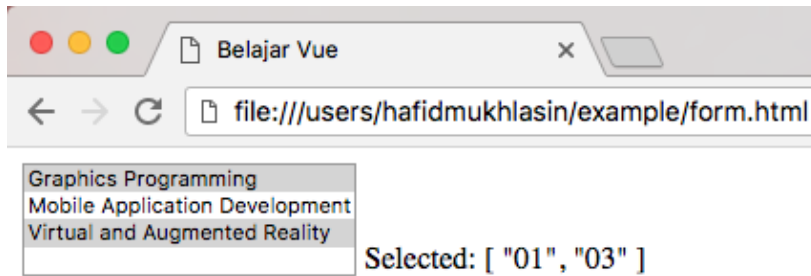
```
var vm = new Vue({
  el: '#app',
  data: {
    hobbies: [],
    categories: []
  }
})
```

Tampilan yang berulang dengan pola tertentu seperti option pada field input select, tentu saja bisa kita *generate* menggunakan *directive v-for* sebagaimana yang telah dibahas pada bagian terdahulu.

```
data: {
  categories: [],
  options: [
    { text: 'Graphics Programming', value: '01' },
    { text: 'Mobile Application Development', value: '02' },
    { text: 'Virtual and Augmented Reality', value: '03' }
  ]
}
```

```
<select name="categories" v-model="categories" multiple>
  <option v-for="option in options" :value="option.value">
    {{ option.text }}
  </option>
</select>
<span>Selected: {{ categories }}</span>
```

Atribut value *dibind*.



## Filtering Data List

Sebelum kita gunakan form untuk submit data, ada satu materi terkait dengan list namun sengaja dibahas pada bab ini karena terkait dengan form input. Sederhana sekali sebenarnya karena secara umum konsepnya telah dibahas pada bab list tersebut.

Materi ini adalah membuat field pencarian atau filtering data buku dalam bentuk list. Pada template kita perlu satu filed input dan list.

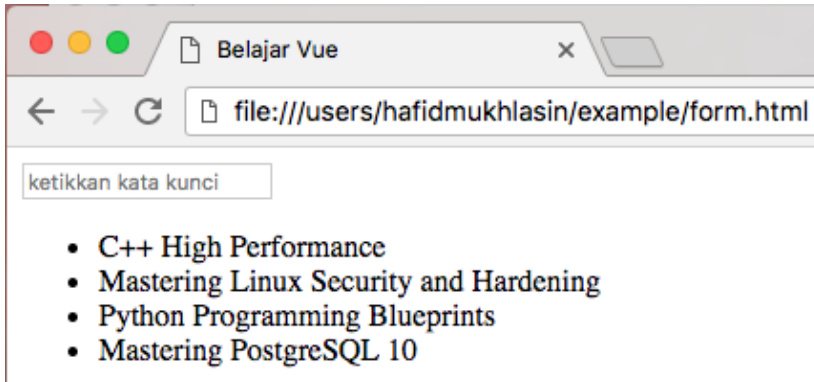
```
<div id="app">
  <input type="text" v-model="keyword" placeholder="ketikkan kata kunci">
  <ul>
    <li v-for="(book, index) of filterBooks" :key="index">
      {{ book }}
    </li>
  </ul>
</div>
```

Adapun kode Javascript-nya sebagai berikut.

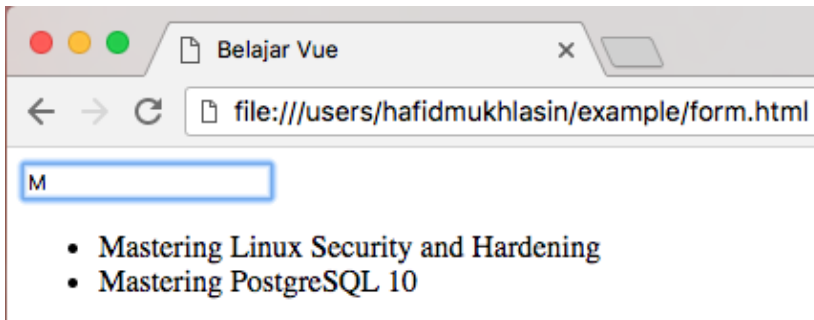
```
var vm = new Vue({
  el: '#app',
  data: {
    keyword: '',
    books : [
      'C++ High Performance',
      'Mastering Linux Security and Hardening', 'Python Programming
Blueprints',
      'Mastering PostgreSQL 10'
    ]
  },
  computed: {
    filterBooks() {
      return this.books.filter((book)=>{
        return book.includes(this.keyword)
      })
    }
  }
})
```

Ada dua variabel yaitu keyword untuk menampung kata kunci dan books untuk menampung data buku. Disamping itu ada satu fungsi pada properti computed yaitu filterBooks. Fungsi filterBook akan melakukan pengecekan adakah item di variabel books yang berisi variabel keyword menggunakan method bawaan Javascript yaitu `includes`.

Hasilnya sebagai berikut.



Ketika diinput teks



## Handling Submit Form & Validation

Pertanyaan: Apa fungsi elemen HTML form jika field bisa lewat begitu saja melalui v-model tanpa perlu submit?

Sebagaimana kita ketahui bahwa form memiliki event submit yang umumnya digunakan untuk mengirimkan data (yang berasal dari field input user) ke server atau ke bagian lain dari aplikasi.

Sudah menjadi konsensus bahwa sebuah form yang berisi beberapa field input data membutuhkan button submit yang menandai bahwa semua data yang diisi melalui field input tersebut siap dikirimkan.

Berikut ini contoh template form input data buku.

```
<form @submit="submitForm($event)" action="http://example.com/add-product"
method="post">
  <label>Title:</label>
  <input type="text" v-model="title" />

  <label>Description:</label>
  <textarea v-model="description"></textarea>
```



```

<label>Authors:</label>
<input type="text" v-model="authors">

<label>Price:</label>
<input type="number" v-model.number="price">

<label>Categories:</label>
<select v-model="categories" multiple>
  <option v-for="option in options" :value="option.value">
    {{ option.text }}
  </option>
</select>

<label></label>
<input type="submit" value="Submit">
</form>

```

Pada elemen form `<form @submit="submitForm" action="http://example.com/add-product" method="post">`, kita menggunakan 3 atribut yaitu directive `v-on:submit` atau `@submit` (yang akan dijalankan ketika form disubmit), `action` (endpoint pengiriman data), dan `method` (metode pengiriman data apakah get atau post).

Jika kita menggunakan Vue untuk membuat aplikasi SPA (Single Page Application) maka dua atribut terakhir yaitu `action` dan `method` menjadi *unfaedah* 😊. Oleh karena itu, atribut sekaligus directive `@submit` yang kita jadikan tumpuan untuk dibahas pada bagian ini.

Directive ini pada contoh diatas memanggil fungsi `submitForm` yang mana pada fungsi ini kita bisa gunakan untuk misalnya: memvalidasi isian dari user, melakukan kalkulasi jika diperlukan, mengirimkan data isian tersebut ke server melalui http client, dsb.

```

var vm = new Vue({
  el: '#app',
  data: {
    title: '',
    description: '',
    authors: '',
    price: 0,
    categories: [],
    options: [
      { text: 'Graphics Programming', value: '01' },
      { text: 'Mobile Application Development', value: '02' },
      { text: 'Virtual and Augmented Reality', value: '03' }
    ]
  },
  methods: {
    submitForm(event){
      console.log(event)

      // kode validasi
    }
  }
})

```

```

        // kode kirim ke server

        // kode status informasi
        alert('Terima kasih')

        // block redirect ke action
        event.preventDefault()
    }
}
})

```

Kode `event.preventDefault()` pada methods `submitForm` berfungsi untuk mencegah agar form tidak diredirect ke alamat yang didefinisikan di atribut `action`, hal ini dikarenakan pengiriman data tidak melalui cara normal HTML melainkan melalui Javascript (http client). Disamping cara ini, kita bisa juga menggunakan modifier `prevent` pada directive `@submit`

```
<form @submit.prevent="submit"...
```

Supaya tampilan form lebih rapi, untuk sementara kita menggunakan style CSS berikut (letakkan pada elemen head HTML).

```

<style>
form {
    border: 1px solid #ddd;
    padding: 5px;
    width: 225px;
    background: #efefef;
}
label{
    display: block;
    margin-top: 5px;
}

input, textarea, select, option {
    min-width: 200px;
}
</style>

```

Mari kita lihat hasilnya.

The screenshot shows a web browser window with the title 'Belajar Vue'. The address bar shows the file path: `file:///users/hafidmukhlisin/example/form.html`. The form contains the following fields:

- Title:**
- Description:**
- Authors:**
- Price:**
- Categories:**

A blue 'Submit' button is at the bottom of the form. A modal dialog box is open on the right with the title 'From This Page' and the text 'Terima kasih' (Thank you). It has an 'OK' button.

Selanjutnya, kita akan fokus pada bagian methods `submitForm()`.

## Validasi Data

Proses validasi adalah proses memastikan setiap isian yang diinput oleh user melalui form tersebut sesuai dengan persyaratan minimal yang kita tentukan. Misalnya: title harus diisi dan minimal 3 karakter, description boleh tidak diisi namun kalau diisi maka tidak boleh lebih dari 500 karakter, price tidak boleh minus, dsb. Jika ditemukan terdapat isian yang tidak memenuhi syarat minimal maka akan dimunculkan pesan peringatan berupa alert serta dicatat sebagai error (counter). Pada bagian akhir kemudian dicek secara total apakah terdapat error (error lebih dari 0) ataupun tidak (error = 0), jika tidak ada error maka ditampilkan pesan terima kasih dan kode untuk mengirim data tersebut ke server.

```
submitForm(event){
  let error = 0

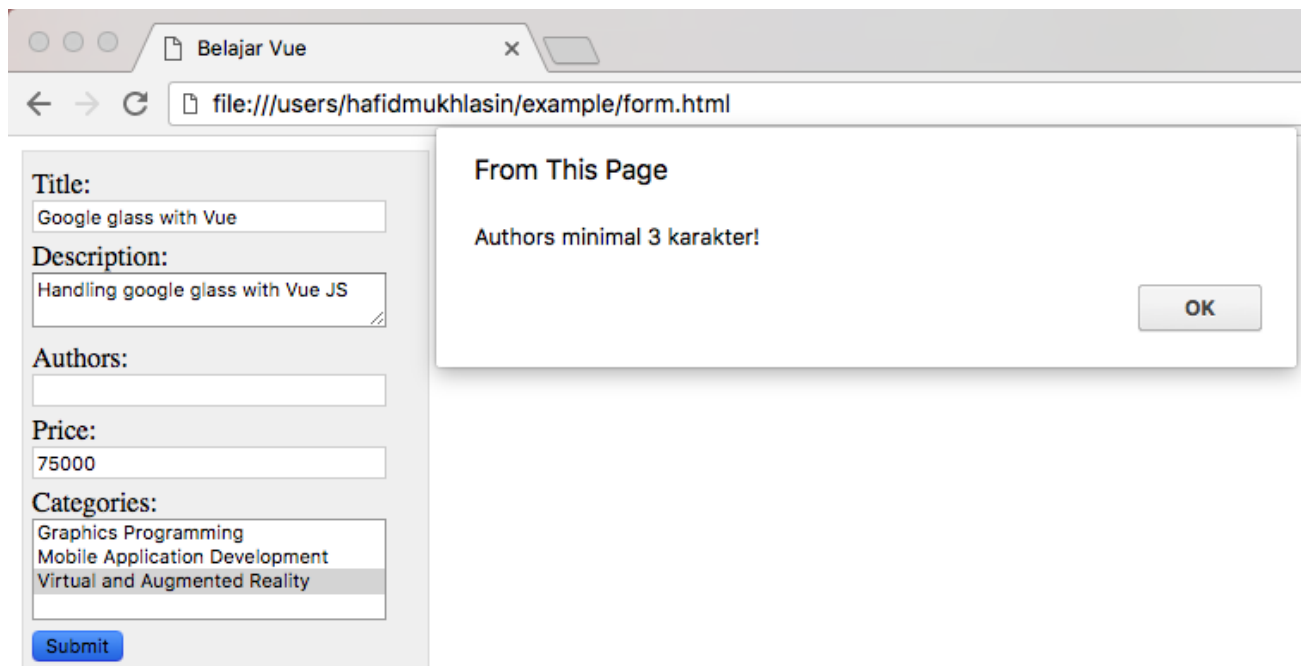
  if(this.title.length < 3){
    error++
    alert('Title minimal 3 karakter!')
  }
  else if(this.description.length > 500){
    error++
    alert('Description maximal 500 karakter!')
  }
  else if(this.authors.length < 3){
    error++
    alert('Authors minimal 3 karakter!')
  }
  else if(this.price < 0){
    error++
    alert('Price tidak boleh minus!')
  }
  else if(this.categories.length === 0){
```

```
error++
alert('Pilih minimal 1 category!')
}

if( error === 0 ){
  alert('Terima kasih telah mengisi data dengan benar!')
  // kirim data ke server
}

event.preventDefault()
}
```

Berikut hasilnya jika ada field yang tidak memenuhi minimal persyaratan

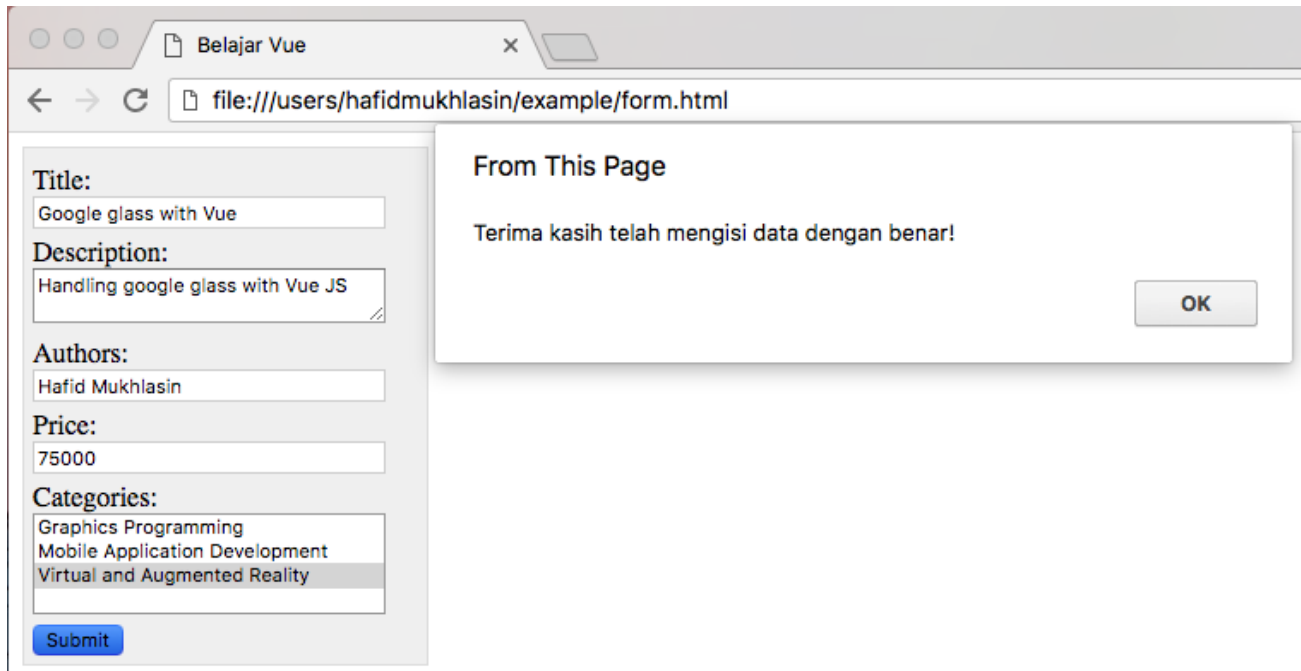


The screenshot shows a web browser window with the title "Belajar Vue". The address bar displays the file path "file:///users/hafidmukhlasi/example/form.html". The form contains the following fields:

- Title:** Google glass with Vue
- Description:** Handling google glass with Vue JS
- Authors:** (empty field)
- Price:** 75000
- Categories:** Graphics Programming, Mobile Application Development, Virtual and Augmented Reality

A blue "Submit" button is at the bottom of the form. An alert dialog box is open, displaying the message "From This Page" and "Authors minimal 3 karakter!". The "OK" button is on the right side of the alert box.

Dan berikut ini jika semua semua field telah memenuhi persyaratan.



Kita bisa juga menambahkan method `setFocus` pada input yang belum memenuhi syarat. Sehingga akan memudahkan dari sisi user. Caranya dengan menambahkan directive `ref` sebagai penanda unik pada field input.

```
<input type="text" name="title" ref="title" v-model="title">
```

Melalui directive tersebut kemudian bisa kita akses dengan kode berikut.

```
if(this.title.length < 3){
  error++
  this.$refs.title.focus()
  alert('Title minimal 3 karakter!')
}
```

Bisa juga dengan kode ini `this.$refs.title.select()`.

Catatan: tambahkan juga directive `ref` pada form supaya lebih mudah nantinya menangani elemen form tersebut.

Supaya lebih user friendly, pesan *error* tidak kita munculkan dalam bentuk *alert*, melainkan dalam bentuk teks di browser. Kita bisa kombinasikan dengan teknik list untuk menampilkan *error* yang akan kita tampung dalam bentuk array.

Mari kita ubah kode konstruktor Vue menjadi sebagai berikut.

```
var vm = new Vue({
  el: '#app',
  data: {
```

```

    title: 'Google Glass with VueJS',
    description: 'Control Google Glass with VueJS',
    authors: 'Hafid Mukhlasin',
    price: 75000,
    categories: [],
    options: [
      { text: 'Graphics Programming', value: '01' },
      { text: 'Mobile Application Development', value: '02' },
      { text: 'Virtual and Augmented Reality', value: '03' }
    ],
    errors: []
  },
  methods: {
    submitForm(event){
      this.errors = []
      if(this.title.length < 3){
        this.errors.push('Title minimal 3 karakter!')
        this.$refs.title.select()
      }
      if(this.description.length > 500){
        this.errors.push('Description maximal 500 karakter!')
        this.$refs.description.select()
      }
      if(this.authors.length < 3){
        this.errors.push('Authors minimal 3 karakter!')
        this.$refs.authors.select()
      }
      if(this.price < 0){
        this.errors.push('Price tidak boleh minus!')
        this.$refs.price.select()
      }
      if(this.categories.length === 0){
        this.errors.push('Pilih minimal 1 category!')
        this.$refs.categories.focus()
      }

      if( this.errors.length === 0 ){
        alert('Terima kasih telah mengisi data dengan benar!')
        // kirim data ke server
      }
    }
  }
})

```

Kemudian pada template, kita ubah menjadi sebagai berikut.

```

<form ref="formBook" @submit.prevent="submitForm($event)"
action="http://example.com/add-product" method="post">

  <p v-if="errors.length">
    <b>Please correct the following error(s):</b>
    <ul>

```

```
        <li v-for="error in errors">{{ error }}</li>
    </ul>
</p>

<label>Title:</label>
<input name="title" ref="title" type="text" v-model="title">

<label>Description:</label>
<textarea name="description" ref="description" v-model="description">
</textarea>

<label>Authors:</label>
<input name="authors" ref="authors" type="text" v-model="authors">

<label>Price:</label>
<input name="price" ref="price" type="number" v-model.number="price">

<label>Categories:</label>
<select name="categories" ref="categories" v-model="categories"
multiple>
    <option v-for="option in options" :value="option.value">
        {{ option.text }}
    </option>
</select>

<label></label>
<input type="submit" value="Submit">
</form>
```

Kode di atas akan menghasilkan tampilan berikut.

**Please correct the following error(s):**

- Authors minimal 3 karakter!
- Pilih minimal 1 category!

**Title:**

**Description:**

**Authors:**

**Price:**

**Categories:**

Catatan: tutorial ini hanya menunjukkan tentang bagaimana validasi itu bekerja, selanjutnya tentu kamu bisa gunakan berbagai cara untuk memastikan bahwa input data user sesuai dengan yang diharapkan. Disamping cara manual ini, ada beberapa pustaka Vue yang bisa kita gunakan untuk memudahkan kita melakukan validasi data form, diantaranya:

- <https://github.com/monterail/vuelidate>
- <http://vee-validate.logaretm.com>

Peringatan: jangan lupa bahwa validasi ini hanya dari sisi client yang sangat rawan untuk dimanipulasi oleh user "nakal". Oleh karena itu validasi dari sisi server, merupakan hal mutlak yang harus kita lakukan untuk memastikan bahwa apa yang diinput oleh user itu sesuai dengan harapan kita.

## Prepare Data Submit

Setelah validasi form dilakukan, langkah berikutnya adalah mempersiapkan data sebelum dikirimkan ke server. Kita bisa menggunakan object FormData (<https://developer.mozilla.org/en-US/docs/Web/API/FormData/FormData>) untuk mem-packing data hasil isian form menjadi sebuah object.

```
let formData = new FormData();
// tambahkan satu persatu field
formData.append('username', 'Chris');
```



Berikut ini implementasi pada kasus kita.

```
if( this.errors.length === 0 ){
  alert('Terima kasih telah mengisi data dengan benar!')
  // persiapan data
  let formData = new FormData()
  formData.append('title', this.title)
  formData.append('description', this.description)
  formData.append('authors', this.authors)
  formData.append('price', this.price)
  formData.append('categories', this.categories)

  // kirim data ke server
}
```

Objek formData inilah yang akan dikirimkan ke server atau diproses lebih lanjut.

Di samping cara di atas yaitu manual satu persatu dalam memasukkan data field, FormData juga menyediakan cara cepat untuk memasukkan semua data field yang dimiliki form ke dalam objek formData. Berikut yang kita dapat dari dokumentasi FormData.

```
let myForm = document.getElementById('myForm');
formData = new FormData(myForm);
```

Dengan sedikit modifikasi maka implementasi pada kasus kita menjadi sebagai berikut

```
// persiapan data
let formBook = this.$refs.formBook
formData = new FormData(formBook);
// kirim data ke server
```

Selesai.

Catatan: untuk bisa menggunakan cara singkat ini syaratnya satu yaitu field pada form harus ada atribut **name**-nya.

## Send Data To Server

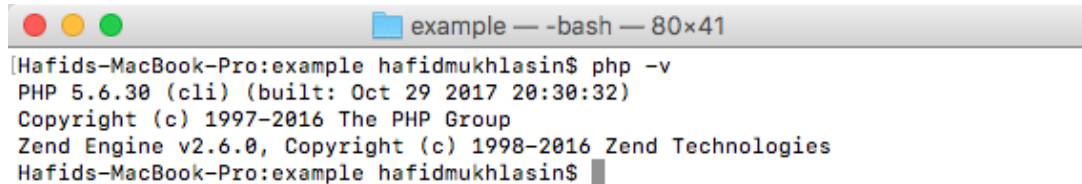
Setelah data di-*bundle* dalam satu objek formData, maka data siap untuk dikirim ke server. Pada bagian ini, kita akan mensimulasikan pengiriman data form ke server menggunakan PHP native. Oleh karenanya siapkan web server (nginx atau apache) serta PHP untuk dapat mencoba simulasi ini, atau bisa juga dengan menggunakan PHP built-in server.

Catatan: pada bab ini tidak akan dijelaskan tentang bagaimana instalasi PHP, dan Web Server. kamu bisa menggunakan panduan lain untuk menginstalasinya atau merujuk langsung ke website resmi PHP (<https://php.net>)

Pada tutorial ini, kita akan menggunakan built-in web server bawaan PHP. Oleh karena itu, pastikan PHP sudah terinstalasi dengan baik sehingga bisa diakses melalui terminal (command prompt atau powershell pada Windows).

Jalankan perintah

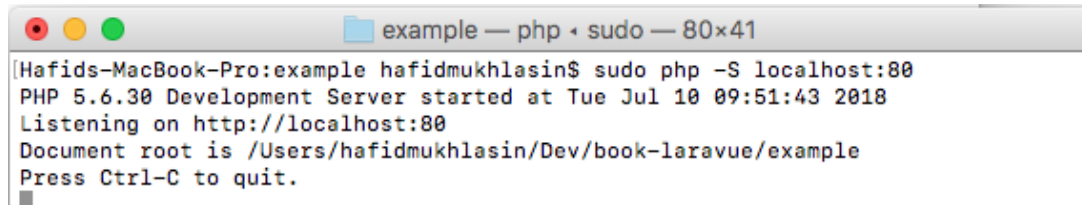
```
php -v
```



```
example — -bash — 80x41
[Hafids-MacBook-Pro:example hafidmukhlasi$ php -v
PHP 5.6.30 (cli) (built: Oct 29 2017 20:30:32)
Copyright (c) 1997-2016 The PHP Group
Zend Engine v2.6.0, Copyright (c) 1998-2016 Zend Technologies
Hafids-MacBook-Pro:example hafidmukhlasi$
```

Kemudian jalankan PHP built-in server dengan menggunakan format perintah berikut:

```
php -S localhost:80
```



```
example — php - sudo — 80x41
[Hafids-MacBook-Pro:example hafidmukhlasi$ sudo php -S localhost:80
PHP 5.6.30 Development Server started at Tue Jul 10 09:51:43 2018
Listening on http://localhost:80
Document root is /Users/hafidmukhlasi/Dev/book-laravue/example
Press Ctrl-C to quit.
```

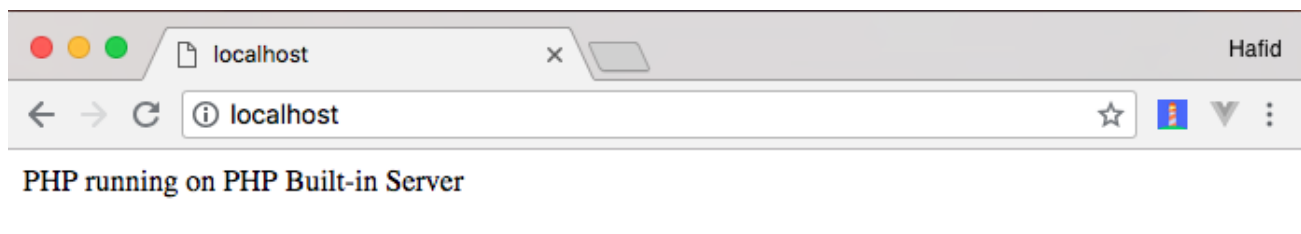
Catatan: 80 adalah nomer port dari webserver, kita bebas mengubahnya dengan nomer lain yang sedang tidak digunakan.

Perintah ini akan menjalankan web server dan menjadikan current directory sebagai web root.

Untuk menguji apakah kode PHP dapat berjalan dengan baik, maka buat file index.php pada **current directory** (pada contoh ini penulis meletakkan pada direktori yang sama dengan file form.html atau kode Vue). Isinya sebagai berikut.

```
<?php
echo "PHP running on PHP Built-in Server";
```

Jalankan pada browser alamat <http://localhost:80>, maka hasilnya.



```
localhost
PHP running on PHP Built-in Server
```

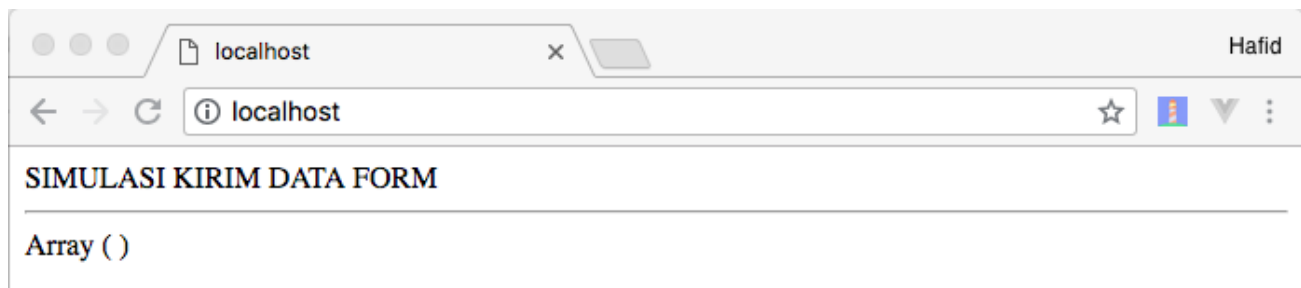
Jika berhasil, maka selanjutnya kita akan membuat kode PHP yang bertugas sebagai *endpoint* penerima data kiriman dari form dan mengembalikan data tersebut dalam bentuk array. Berikut ini kodenya.

```
<?php
// untuk mencegah error akibat CORS
header("Access-Control-Allow-Origin: *");
header('Access-Control-Allow-Credentials: true');
header("Access-Control-Allow-Methods: GET, POST, OPTIONS");

echo "SIMULASI KIRIM DATA FORM <hr>";

// menampilkan data yang dikirimkan dengan method post
print_r($_POST);
```

Kemudian coba akses melalui browser.



Catatan: fungsi CORS pada contoh ini digunakan agar Vue melalui pustaka HTTP client dapat mengakses file PHP yang diletakkan pada server yang berbeda dengan server dimana Vue di-hosting. Selengkapnya silakan baca di <https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS>

Untuk lebih mudahnya, pindahkan juga atau atur agar file-file Vue yang telah kita buat sebelumnya juga berada pada direktori yang sama dengan file index.php

Langkah selanjutnya adalah membuat kode Javascript/Vue untuk mengirimkan data ke server. Ada berbagai cara untuk melakukan request data ke server, diantaranya dengan engine XMLHttpRequest, fungsi fetch (fungsi native Javascript), dan pustaka axios.

Pada bagian ini, kita hanya akan membahas tentang cara request ke server dengan menggunakan XMLHttpRequest. Jadi XMLHttpRequest adalah engine yang digunakan untuk menangani permintaan data ke dan dari server. Engine ini cukup populer digunakan terutama untuk menjalankan AJAX.

```
if( this.errors.length === 0 ){
  //alert('Terima kasih telah mengisi data dengan benar!')

  // persiapkan data
  let formBook = this.$refs.formBook
  formData = new FormData(formBook);

  // kirim data ke server
  let xhttp = new XMLHttpRequest() // create objek XMLHttpRequest

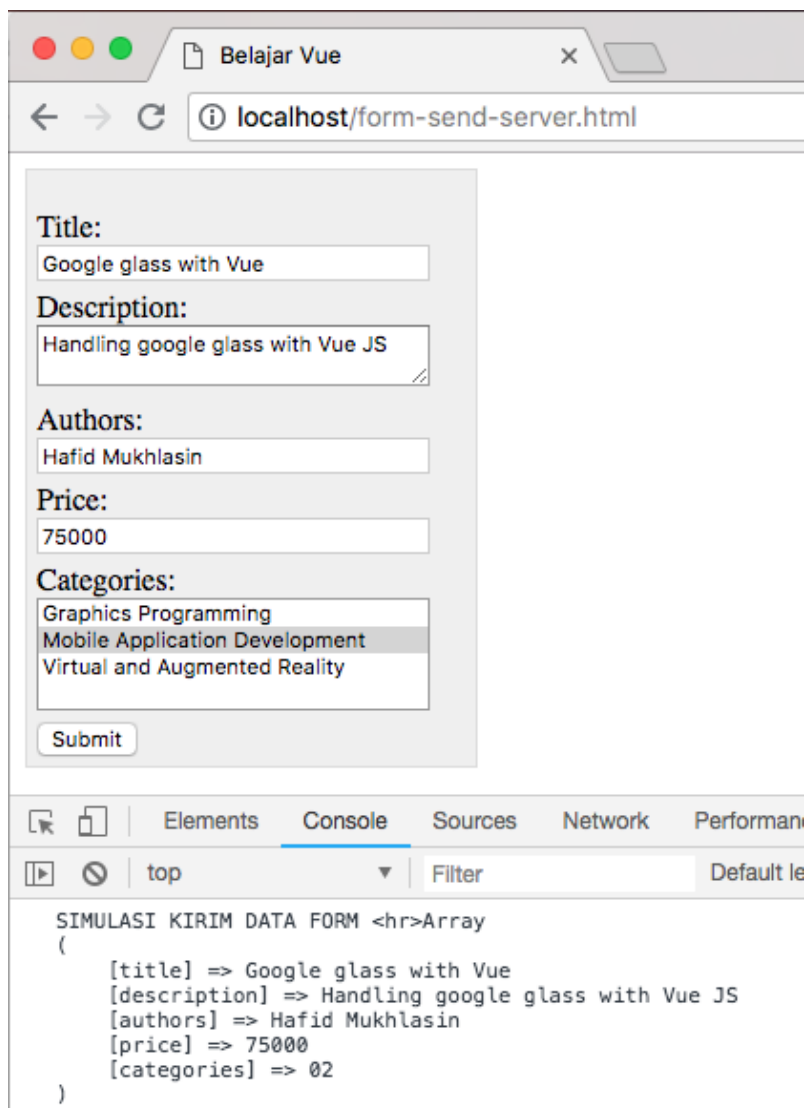
  // definisikan fungsi ketika terjadi perubahan state
```

```
xhttp.onreadystatechange = function() {
    // state ini menunjukkan data terkirim dan diterima server dengan
    baik
    if (this.readyState == 4 && this.status == 200) {
        // respon text dari server
        console.log(this.responseText)
    }
}
// sesuaikan dengan lokasi file index.php di lokasi komputer kamu
xhttp.open("POST", "http://localhost/index.php", true)

// bisa juga langsung nama filenya jika berada dalam satu folder yang
sama
// xhttp.open("POST", "index.php", true)

// kirim objek formData
xhttp.send(formData)
}
```

Mari kita ujicoba kode di atas pada browser.



## Handling File Upload

Pada sisi client, penanganan field bertipe file pada form pada dasarnya hampir sama saja dengan field bertipe lain.

Sebagai simulasi, kita gunakan kode sebelumnya. Pada template form tambahkan field input bertipe file.

```
<label>Cover:</label>
<input name="cover" ref="cover" type="file">
```

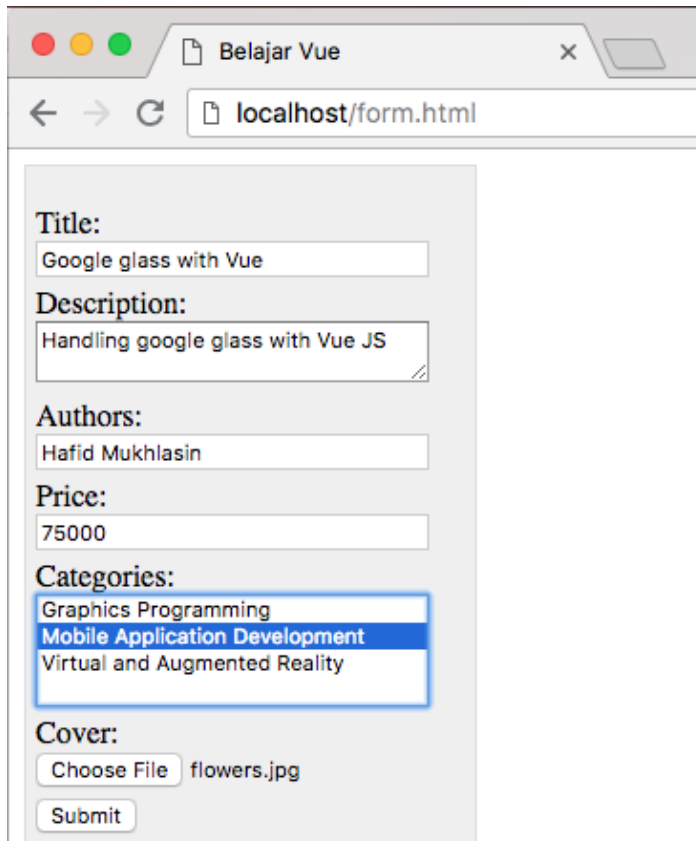
Kemudian pada kode Javascript-nya tambahkan kode berikut

```
// get file yang dibrowse user
let cover = this.$refs.cover.files[0]
// tambahkan ke object formData
formData.append("cover", cover);
```

Dengan cara ini maka file akan terkirim ke server. Untuk mensimulasikannya, edit file `index.php` dan tambahkan kode berikut.

```
// ...
print_r($_FILES['cover']);
```

Hasilnya.



Belajar Vue

localhost/form.html

**Title:**  
Google glass with Vue

**Description:**  
Handling google glass with Vue JS

**Authors:**  
Hafid Mukhlisin

**Price:**  
75000

**Categories:**  
Graphics Programming  
Mobile Application Development  
Virtual and Augmented Reality

**Cover:**  
Choose File flowers.jpg  
Submit

Ketika form disubmit maka pada console log akan terlihat sebagai berikut.

**Title:**  
Google glass with Vue

**Description:**  
Handling google glass with Vue JS

**Authors:**  
Hafid Mukhlisin

**Price:**  
75000

**Categories:**  
Graphics Programming  
Mobile Application Development  
Virtual and Augmented Reality

**Cover:**  
Choose File flowers.jpg  
Submit

```

SIMULASI KIRIM DATA FORM <hr>Array
(
  [title] => Google glass with Vue
  [description] => Handling google glass with Vue JS
  [authors] => Hafid Mukhlisin
  [price] => 75000
  [categories] => 02
)
Array
(
  [name] => flowers.jpg
  [type] => image/jpeg
  [tmp_name] => /private/var/tmp/phpRyZc
  [error] => 0
  [size] => 16119
)

```

## Kesimpulan

Form merupakan media yang digunakan user untuk berinteraksi dengan aplikasi atau web. User dapat menginput data dalam bentuk teks, array, atau file. Validasi merupakan hal yang harus kita lakukan sebab kita tidak tau apakah user benar-benar memasukkan data sesuai dengan harapan kita atau tidak, sekaligus dalam rangka keamanan data.

Pengiriman data ke server menggunakan Javascript membutuhkan tools http client, pada bab ini dibahas mengenai penggunaan tools XMLHttpRequest yang biasa digunakan untuk AJAX. Pilihan lainnya, kita bisa gunakan Fetch (native Javascript) atau yang direkomendasikan Vue yaitu pustaka Axios.

Pada bab berikutnya, kita akan belajar tentang komponen yang merupakan bagian dari Vue yang cukup penting untuk dipelajari guna memudahkan kita dalam mengembangkan aplikasi yang kompleks.

Ayoo lebih semangat lagi!