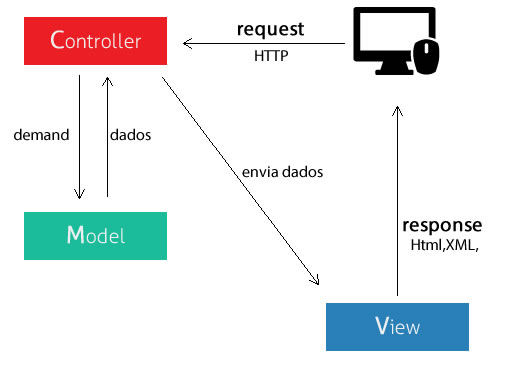
**MVC** é um padrão de arquitetura de software, separando sua aplicação em 3 camadas. A camada de interação do usuário(**view**) responsável pela exibição de dados por meio de html ou xml, a camada de manipulação dos dados(**model**) responsável pela leitura, escrita e validações de dados e a camada de controle(**controller**) responsável por receber todas as requisições do usuário, seus métodos chamados actions são responsáveis por uma página, controlando qual model usar e qual view será mostrado ao usuário.



**Vue.js** é um framework progressivo focado exclusivamente na camada visual, sendo fácil de integrar com outras bibliotecas ou projetos existentes, usado principalmente em SPA em conjuntos de “ferramentas externas” e “biblioteca de apoio”, exemplo: **Quasar.**

Os componentes **Vue** estendem elementos **HTML** básicos para encapsular código reutilizável. Os componentes são elementos personalizados aos quais o compilador do **Vue** associa determinados comportamentos. No **Vue**, um componente é essencialmente uma instância do **Vue** com opções predefinidas.

**MVC em Vue.js**

O **Vue** em si, é uma ótima ferramenta para estruturas com serviços **REST** e aplicativos web **SPA (single page apps),** porém muitos websites hoje, ainda são construído em arquitetura **MVC**, o que dificulta a integração do algumas estruturas **JavaScrip** como o **Vue.** Nós referenciamos **4 loaders** (vue, scss, css e js) para que possamos usar **.vue** (componentes de arquivo único), bem como importações de javascript e css.

O **Vue.js** se posiciona como **The progressive JavaScript Framework,**o que significa que pode começar a usar-lo como **jQuery,** além disso é sempre recomendado usar o **NPM** e alguns recursos do **ES6 (ECMAScript 6),**

como qualquer outra estrutura de front-end, o **Vue** sofre de um carregamento atrasado. Quando uma página é solicitada, o javascript precisa ser baixado e carregado. Em seguida, a estrutura em questão fará mais solicitações de servidor por dados. Quando isso termina, a ligação de dados é feita com a visualização. Para ajustar isso, precisamos recorrer ao carregamento de animações e outros recursos, mas como também estamos usando o **MVC,** podemos fazer melhor e acelerar esse processo de carregamento, aproveitando o poder do **razor (ASP.NET)** e fazendo o download dos dados para a visualização, juntamente com o restante da página.

Aplicativos mais complexos precisam de várias visualizações, pois há muita informação para uma página. É quando a combinação do **Vue com o mvc** realmente brilha, pois podemos carregar todas as visualizações relacionadas no **Vue** enquanto permanecemos na página **MVC**, evitando assim o recarregamento da página inteira.

<https://tableless.com.br/mvc-afinal-e-o-que/> **(MVC)**

<https://br.vuejs.org/v2/guide/index.html> **(Vue.js)**

<https://docs.google.com/presentation/d/1JYYBq63lB5r4fC4e8A3EVa5v2pZDcNMD5G4ieg6ltWo/edit#slide=id.p21> **(Slide Cazé)**

<https://github.com/lukehoban/es6features> **(ECMAScript 6)**

<http://www.lambdatwist.com/dot-net-vuejs/> **(Vue.js + .NET.MVC)**

<https://medium.com/corebuild-software/vue-js-and-net-mvc-b5cede228626> **(Vue.js + MVC)\***

<https://github.com/cristijora/aspnet-vue-mvc> **(Vue.js + MVC exemplo)**