方案一

前端使用ArgularJS编写SPA(Single Page Application)应用，后端服务器只提供RESTful接口且完全无状态化。

优点：

* 前后完全分离，前端开发者完全不需要关心服务端用了什么技术，只需要一份接口说明文档即可。
* 手机App与PC端网站可以共用同一套接口。

缺点：

* SEO困难。

方案二

使用NodeJS渲染Web页面，然后调用后端RESTful接口。

优点：

* 前后完全分离, 后端开发者可以专注于业务逻辑开发。

缺点：

* NodeJS代码如果异常处理不好容易直接挂掉进程。
* 增加了部署和维护成本
* 对前端开发者技术要求比较高
* 增加了一层NodeJS，提高了网络传输的开销

方案三

使用Velocity(或其它渲染引擎)，[前端开发](http://lib.csdn.net/base/javascript)者只写静态页，完成后交给后端进行“动态化”

优点：

* 前端完全不需要关注后端的任何东西。

缺点：

* 后端开发者必须熟悉HTML , CSS, JS，不能专注业务开发。
* 前后端沟通成本大
* 后端服务器如果需要为App提供接口，则需要写两套Controller

方案四

前端开发者学习使用Velocity语法，直接编写\*.vm文件

优点：

* 后端完全不需要关心页面问题

缺点：

* 前端人员学习成本高
* 需要在本地搭建、部署Java运行环境，对前端人员来说难度很大。
* 后端服务器如果需要为App提供接口，则需要写两套Controller

目前我们使用一、四结合的方案，对NodeJS保持谨慎态度。 前端开发者更倾向于SPA和NodeJS.