****

**本 科 毕 业 设 计**

|  |  |
| --- | --- |
| **题**  **目** | 基于Node.js的Web 框架生成工具 |

**作 者：** 韦 远

**专 业：** 计算机科学与技术

**指导教师：** 朱 晓 辉

**完成日期：** 2017年 5 月 31日

# 摘要

随着互联网技术的不断发展，网站已经成为人们不可或缺的一部分，但是随着Web技术的不断发展，各式各样的Web前端与后端的技术出现在了市场上面，让程序员不知道该如何选择。所以为了网站快速而高效的开发，本课题创建了一个以Angular为前端框架，Node.js为后端支撑，Mongodb为数据支持的网站模板，本系统功能架构大致分为五个部分：关键技术介绍模块，选择技术生成框架模块，Web框架生成模块，用户下载使用框架模块，系统扩展性设计模块，再配合Webpack强大的打包功能，以及前后端统一使用JavaScript语言开发，使用我们的这个模板来开发网站将体会到前所未有的快速轻量与高效。

关键字：Web框架，Node.js，Angular

# ABSTRACT

With the continuous development of Internet technology, the site has become an indispensable part of people, but with the continuous development of Web technology, a wide range of Web front-end and back-end technology in the market above, so that programmers do not know the how to choose. Therefore, for the rapid and efficient development of the website, this topic creates a website template with Angular as the front-end framework, Node.js as the back-end support, and Mongodb for the data support. The system's functional architecture is divided into four parts: Key Technology Module, select the technology to generate the framework module, the Web framework generation module, the user download the use of the framework module, system scalability design module, coupled with Webpack powerful packaging function, and the front and back unified use of JavaScript language development, the use of our template to develop the site will experience unprecedented fast Lightweight and efficient.

Keywords: Web framework, Node.js, Angular

# 目录

[摘要 2](#_Toc483868723)

[ABSTRACT 3](#_Toc483868724)

[目录 4](#_Toc483868725)

[引言 6](#_Toc483868726)

[第一章 系统概述 7](#_Toc483868727)

[1.1 研究背景 7](#_Toc483868728)

[1.2 研究意义 7](#_Toc483868729)

[1.3 开发环境和技术介绍 7](#_Toc483868730)

[第二章 系统需求分析 15](#_Toc483868731)

[2.1系统设计目标 15](#_Toc483868732)

[2.2 系统功能分析 15](#_Toc483868733)

[2.2.1 展现Web开发技术功能 15](#_Toc483868734)

[2.2.2 选择技术生成框架功能 16](#_Toc483868735)

[2.2.3 打包管理模块功能 16](#_Toc483868736)

[2.2.4 使用ES6/7新特性功能 17](#_Toc483868737)

[2.3 系统可行性分析 17](#_Toc483868738)

[2.4 研究方法与技术路线 18](#_Toc483868739)

[第三章 系统概要设计 19](#_Toc483868740)

[3.1 系统设计思路 19](#_Toc483868741)

[3.2 系统功能架构 19](#_Toc483868742)

[3.2.1 Web网站架构 19](#_Toc483868743)

[3.2.2 Web框架架构 21](#_Toc483868744)

[第四章 系统详细设计 25](#_Toc483868745)

[4.1 关键技术介绍模块 25](#_Toc483868746)

[4.2 选择技术生成框架模块 25](#_Toc483868747)

[4.3 Web框架生成模块 26](#_Toc483868748)

[4.4 用户下载使用框架模块 28](#_Toc483868749)

[4.5 系统扩展性设计模块 28](#_Toc483868750)

[4.6 框架设计模块 28](#_Toc483868751)

[4.6.1 框架前端设计 28](#_Toc483868752)

[4.6.2 框架后端设计 29](#_Toc483868753)

[第五章 系统实现 32](#_Toc483868754)

[5.1 系统展示 32](#_Toc483868755)

[5.2 系统编码 33](#_Toc483868756)

[第六章 结论 37](#_Toc483868757)

[全文总结 37](#_Toc483868758)

[展望 37](#_Toc483868759)

[参考文献 38](#_Toc483868760)

[致谢 39](#_Toc483868761)

# 引言

随着互联网技术的不断发展，网站已经成为人们不可或缺的一部分，人们无时不刻不在浏览各式各样的网站，而网站也经历了从初始的静态页面到现在的充满大量的用户交互的过程。程序员在这里就起到了很重要的作用，是他们将网站搭建出来，但是随着Web技术的不断发展，各式各样的Web前端与后端的技术出现在了市场上面，前端有Angular，Vue，React等，后端有Node.js，PHP，Java等，让程序员不知道该如何选择。所以为了网站快速而高效的开发，我们创建了一个以Angular为前端框架，Node.js为后端支撑，Mongodb为数据支持的网站模板，本系统功能架构大致分为五个部分：关键技术介绍模块，选择技术生成框架模块，Web框架生成模块，用户下载使用框架模块，系统扩展性设计模块，再配合Webpack强大的打包功能，以及前后端统一使用JavaScript语言开发，使用我们的这个模板来开发网站将体会到前所未有的快速轻量与高效。本文的第一部分主要讲述了课题研究的背景与意义，第二部分主要讲述了系统的需求分析，包括系统设计目标，功能和可行性分析以及研究方法技术路线，第三部分讲述了系统的设计思路以及功能架构，第四部分讲述了系统四个模块的详细设计，第五部分讲述了系统的展现于代码实现过程。

# 第一章 系统概述

## 1.1 研究背景

随着Web开发技术的不断更新增长，Web开发越来越多样化，复杂化，现如今，Web开发技术种类繁多，很多人在进行Web开发时还不知道如何选择开发技术以及不知道选择何种开发技术，有的甚至习惯于一层不变的使用一些已经落后的被淘汰的技术，从而使得Web开发变得低效而且复杂化。

## 1.2 研究意义

首先可以让用户了解到一些Web开发的常用的技术，然后根据项目的业务需求选择相应的技术生成Web模版，最后使用该模版来开发网站。从而实现Web开发的快速，高效，模块化等功能。

## 1.3 开发环境和技术介绍

本系统的网站开发采用AngularJs和Bootstrap作为前端框架，使用Webpack作为前端的打包管理工具，后端则采用Node.js作为支撑，数据库采用Nosql的Mongodb。

相关技术介绍：

1. Node.js

Node.js是一个用于执行JavaScript代码服务器端的开放源码的跨平台JavaScript运行时环境。 JavaScript历史上主要用于客户端脚本，其中使用JavaScript编写的脚本嵌入到网页的HTML中，由用户的Web浏览器中的JavaScript引擎在客户端运行。 Node.js使JavaScript可用于服务器端脚本，并在脚本发送到用户的Web浏览器之前运行脚本服务器端来生成动态网页内容。因此，Node.js已经成为“JavaScript无处不在”范例的基础元素之一，允许Web应用程序开发统一单一的编程语言，而不是依靠不同的语言编写服务器端脚本。

虽然.js是JavaScript代码的常规文件扩展名，但是“Node.js”的名称不是指在此上下文中的特定文件 - 它只是产品的名称。 Node.js具有能够进行异步I / O的事件驱动架构。这些设计选择旨在通过许多输入/输出操作以及实时Web应用程序（例如实时通信程序和浏览器游戏）来优化Web应用程序的吞吐量和可扩展性。

Node.js允许使用JavaScript和一系列处理各种核心功能的“模块”来创建Web服务器和网络工具。 为文件系统I / O，网络（DNS，HTTP，TCP，TLS / SSL或UDP），二进制数据（缓冲区），加密功能，数据流和其他核心功能提供模块 ] Node.js的模块使用一个旨在降低编写服务器应用程序的复杂性的API

Node.js应用程序可以在Linux，macOS，Microsoft Windows，NonStop，和Unix服务器上运行。他们可以替代地使用CoffeeScript （JavaScript替代），Dart或Microsoft TypeScript（强类型JavaScript）或可以编译为JavaScript的任何其他语言编写。

Node.js主要用于构建网络程序，如Web服务器。 Node.js和PHP之间最大的区别是，PHP中的大多数函数都将被阻塞直到完成（仅在之前的命令完成后才执行命令），而Node.js中的函数被设计为非阻塞（命令并行执行并使用回叫信号完成或失败）

Node.js为Web服务器提供事件驱动的编程，可以在JavaScript中开发快速的Web服务器。开发人员可以创建高度可扩展的服务器，而无需使用线程，通过使用事件驱动编程的简化模型，使用回调来表示完成任务。 Node.js的创建是因为许多服务器端编程语言中的并发性很难，而且往往导致性能下降。 Node.js将脚本语言（JavaScript）与Unix网络编程的强大功能相结合。

Node.js具有如下的技术特点：

1. Node.js是一个Javascript运行时环境，它处理一个循环中的传入请求，称为事件循环。
2. Node.js在单线程上运行，使用非阻塞I / O调用，允许它支持数万个并发连接，而​​不会导致线程上下文切换的成本。在使用观察者模式的所有请求之间共享单线程的设计旨在构建高度并发的应用程序，其中执行I / O的任何功能都必须使用回调。为了适应单线程事件循环，Node.js利用libuv库，该库依次使用固定大小的线程池，该线程池负责一些非阻塞异步I / O操作。这种单线程方法的缺点是，Node.js不允许通过增加其运行的机器的CPU内核数量而不使用附加模块（如集群） StrongLoop Process Manager 或pm2。但是，开发人员可以增加libuv线程池中的默认线程数;这些线程可能由服务器操作系统分布在多个核心上。在Node.js中执行并行任务由线程池处理。主线程调用函数将任务发布到线程池中的线程的共享任务队列中，并执行。固有的非阻塞系统功能（如网络）转换为内核侧非阻塞套接字，而固有地阻止系统功能（如文件I / O）在其自身线程上以阻塞方式运行。当线程池中的线程完成任务时，它会通知主线程，该线程又唤醒并执行已注册的回调。由于回调在主线程上串行处理，持久的计算和其他CPU绑定的任务将冻结整个事件循环直到完成。
3. V8是为Google Chrome构建的JavaScript执行引擎，由Google于2008年开放.V8编写C ++，将JavaScript源代码编译为本机代码，而不是实时解释。
4. Node.js使用libuv处理异步事件。 Libuv是基于Windows和POSIX的系统（如NonStop和Unix上的Linux，macOS，OSS）的网络和文件系统功能的抽象层。
5. Node.js的核心功能驻留在JavaScript库中。以C ++编写的Node.js绑定将这些技术相互连接并连接到操作系统。
6. npm是Node.js服务器平台的预装包管理器。它用于从npm注册表安装Node.js程序，组织第三方Node.js程序的安装和管理。不要将npm与CommonJS require（）语句混淆。它不用于加载代码;而是用于从命令行安装代码并管理代码依赖关系。在npm注册表中找到的软件包可以从简单的帮助程序库（如Underscore.js）到任务运行程序（如Grunt）。
7. Node.js可以与浏览器，支持JSON数据的数据库（如Postgres， MongoDB或CouchDB）和JSON结合使用，用于统一的JavaScript开发堆栈。通过适应本质上是MVC，MVP，MVVM等服务器端开发模式，Node.js允许在客户端和服务器端之间重用相同的模型和服务接口。
8. Node.js向操作系统注册，以便在连接时通知，并且操作系统将发出回调。在Node.js运行时，每个连接都是一个小的堆分配。传统上，相对重量级的操作系统进程或线程处理每个连接。 Node.js使用事件循环来实现可伸缩性，而不是进程或线程。与其他事件驱动的服务器相反，Node.js的事件循环不需要显式调用。而是定义了回调函数，服务器会自动在回调定义的末尾输入事件循环。当没有进一步的回调时，Node.js退出事件循环。
9. AngularJs

AngularJS是基于JavaScript的开源前端Web应用程序框架，主要由Google和个人和企业社区维护，以解决许多开发单页应用程序遇到的挑战。 JavaScript组件补充了Apache Cordova，用于开发跨平台移动应用程序的框架。它旨在通过为客户端模型视图 - 控制器（MVC）和模型视图模型（MVVM）架构以及通常在富Internet应用程序中使用的组件提供框架来简化开发和测试这些应用程序。 2014年，原来的AngularJS团队开始在Angular（应用平台）上工作。

AngularJS框架通过首先阅读HTML页面，其中嵌入了其他自定义标签属性。 Angular将这些属性解释为将页面的输入或输出部分绑定到由标准JavaScript变量表示的模型的指令。这些JavaScript变量的值可以在代码中手动设置，或从静态或动态JSON资源中检索。根据JavaScript分析服务Libscore，AngularJS被用于Wolfram Alpha，NBC，Walgreens，Intel，Sprint，ABC News以及2016年10月测试的100万个网站中的大约12,000个其他网站。 AngularJS是GitHub上所有时间最多的第六大项目。

AngularJS是MEAN堆栈的前端部分，由MongoDB数据库，Express.js Web应用程序服务器框架，Angular.js本身和Node.js服务器运行时环境组成。

AngularJS建立在这样一种观念的基础上，即声明性编程应用于创建用户界面和连接软件组件，而命令式编程更适合于定义应用程序的业务逻辑。框架适应和扩展传统的HTML，通过双向数据绑定来呈现动态内容，允许自动同步模型和视图。因此，AngularJS强调明确的DOM操作，目的是提高可测试性和性能。

AngularJS的设计目标包括：

将DOM操作与应用程序逻辑分离。这样做的难度受到代码结构化的影响。

将应用程序的客户端与服务器端分离。这允许开发工作并行进行，并允许双方重用。

为构建应用程序的旅程提供结构：从设计UI，通过编写业务逻辑到测试。

Angular实现了MVC模式来分离表示，数据和逻辑组件。使用依赖注入，Angular将传统的服务器端服务（如视图相关的控制器）带入客户端Web应用程序。因此，可以减少服务器上的大量负担。

AngularJS以类似于计算机科学基础的方式使用术语“范围”。

在计算机科学中的范围描述了程序中的特定绑定是否有效。 ECMA-262规范将范围定义为：在客户端Web脚本中执行Function对象的词汇环境; 类似于在lambda演算中定义范围。

作为“MVC”架构的一部分，范围形成“模型”，范围内定义的所有变量都可以通过“视图”以及“控制器”进行访问。 范围表现为胶水，并绑定“视图”和“控制器”。

在AngularJS中，“范围”是某种类型的对象，其本身可以在程序的任何给定部分中属于范围或超出范围，遵循与其他任何对象相似的JavaScript中可变范围的通常规则。 当下面使用术语“范围”时，它指的是角度范围对象，而不是名称绑定的范围。

AngularJS双向数据绑定是其最显着的功能，大大减轻了模板职责的服务器后端。 相反，根据模型中定义的范围中包含的数据，模板以纯HTML呈现。 角度的$ scope服务检测模型部分的更改，并通过控制器修改视图中的HTML表达式。 同样，对该视图的任何改变都反映在模型中。 这规避了主动操纵DOM并鼓励Web应用程序的引导和快速原型化的需要。 AngularJS通过将当前值与脏检查过程中较早存储的值进行比较来检测模型中的更改，与模型值更改时触发侦听器的Ember.js和Backbone.js不同。

1. Mongodb

MongoDB支持字段，范围查询，正则表达式搜索。查询可以返回特定字段的文档，还包括用户定义的JavaScript函数。查询也可以配置为返回给定大小的结果的随机抽样。

MongoDB文档中的字段可以使用主索引和辅助索引编制索引。

MongoDB提供了高可用性与副本集。副本集由两个或多个数据副本组成。每个副本集成员可以在任何时候起作用的主副本。默认情况下，所有写入和读取都在主副本上完成。辅助副本使用内置复制维护主数据的副本。当主副本失败时，副本集自动执行选举过程以确定哪个辅助应该成为主要副本。辅助程序可以可选地提供读取操作，但是该数据默认情况下才最终保持一致。

MongoDB使用分片水平缩放。用户选择一个分片键，确定集合中的数据将如何分发。数据被分割为范围（基于分片键）并分布在多个分片之间。 （碎片是一个或多个奴隶的主人）。或者，碎片密钥可以被散列以映射到分片 - 实现均匀的数据分发。

MongoDB可以在多台服务器上运行，平衡负载或复制数据，以防止硬件故障时系统启动并运行。

MongoDB可以用作具有负载平衡和多个机器上的数据复制功能的文件系统，用于存储文件。

MongoDB驱动程序包含了这个名为Grid File System 的功能。 MongoDB向开发人员公开了文件操作和内容的功能。 GridFS用于NGINX 和lighttpd的插件 GridFS将文件分成多个部分或块，并将这些块中的每一个作为单独的文档存储。

MapReduce可用于批量处理数据和聚合操作。

聚合框架使用户能够获得使用SQL GROUP BY子句的结果种类。聚合运算符可以一起组成一个管道 - 类似于Unix管道。聚合框架包括可以从多个文档中加入文档的$ lookup运算符，以及诸如标准偏差的统计运算符。

JavaScript可用于查询，聚合功能（如MapReduce），并直接发送到要执行的数据库。

MongoDB支持固定大小的集合，称为封顶集合。这种类型的集合维护插入顺序，并且一旦达到指定的大小，其行为就像一个循环队列。

1. Webpack

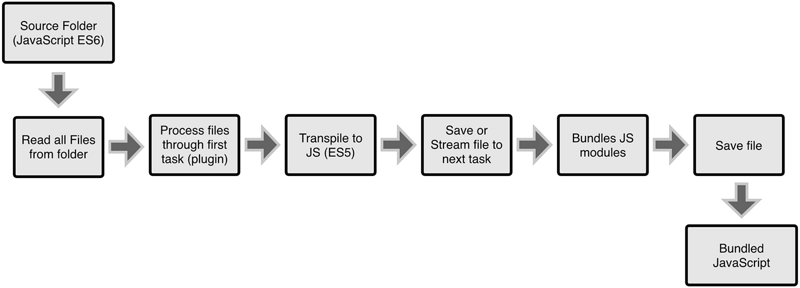
现今的很多网页其实可以看做是功能丰富的应用，它们拥有着复杂的JavaScript代码和一大堆依赖包。为了简化开发的复杂度，前端社区涌现出了很多好的实践方法

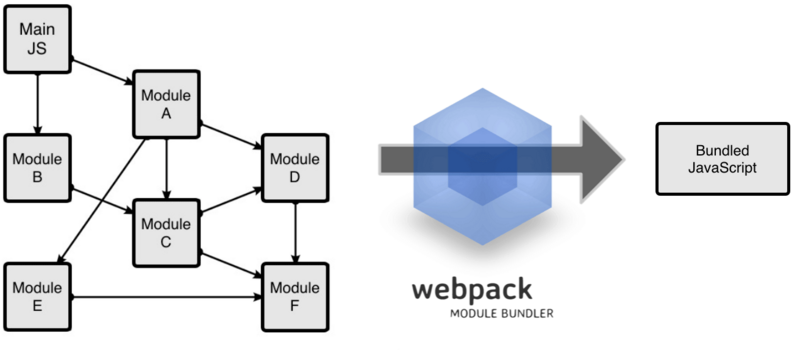
* 模块化，让我们可以把复杂的程序细化为小的文件;
* 类似于TypeScript这种在JavaScript基础上拓展的开发语言：使我们能够实现目前版本的JavaScript不能直接使用的特性，并且之后还能能装换为JavaScript文件使浏览器可以识别；
* Scss，less等CSS预处理器

这些改进确实大大的提高了我们的开发效率，但是利用它们开发的文件往往需要进行额外的处理才能让浏览器识别,而手动处理又是非常反锁的，这就为WebPack类的工具的出现提供了需求。

WebPack可以看做是模块打包机：它做的事情是，分析你的项目结构，找到JavaScript模块以及其它的一些浏览器不能直接运行的拓展语言（Scss，TypeScript等），并将其打包为合适的格式以供浏览器使用。

其实Webpack和另外两个并没有太多的可比性，Gulp/Grunt是一种能够优化前端的开发流程的工具，而WebPack是一种模块化的解决方案，不过Webpack的优点使得Webpack可以替代Gulp/Grunt类的工具。

Grunt和Gulp的工作方式是：在一个配置文件中，指明对某些文件进行类似编译，组合，压缩等任务的具体步骤，这个工具之后可以自动替你完成这些任务。  


Webpack的工作方式是：把你的项目当做一个整体，通过一个给定的主文件（如：index.js），Webpack将从这个文件开始找到你的项目的所有依赖文件，使用loaders处理它们，最后打包为一个浏览器可识别的JavaScript文件。  


如果实在要把二者进行比较，Webpack的处理速度更快更直接，能打包更多不同类型的文件。

# 第二章 系统需求分析

## 2.1系统设计目标

1. 设计一个可供用户了解Web开发相关技术，并能够选择相应的技术生成Web模版，下载这个Web模版。
2. 设计一套Web模版，其中包含用户用户所选的技术文件，以及模块打包管理工具，让用户能够使用这个模版开发Web站点。

## 系统功能分析

我们的Web网站系统为了让用户更好的了解当今Web开发的流行技术，并让用户能够任意组合技术生成Web框架，从而使用我们的框架实现更快速高效的网站开发。系统功能需求如图所示。

图2.2 系统功能分析

### 2.2.1 展现Web开发技术功能

用户在使用web框架建立网站的时候，首先要对web框架里面用到的技术有个深层次的了解，比如前端用到了什么技术，到底适不适合自己的项目需求，在选择web框架用到的技术时，要格外严谨，防止出现选择的技术不适合自己的项目需求的情况。

### 2.2.2 选择技术生成框架功能

用户需要能根据自己对UI框架，前端框架和后端框架的选择创建自己的web模板。

随着CSS3和HTML5的流行，我们的WEB页面不仅需要更人性化的设计理念，而且需要更酷的页面特效和用户体验。作为开发者，我们需要了解一些宝贵的CSS UI开源框架资源，它们可以帮助我们更快更好地实现一些现代化的界面，包括一些移动设备的网页界面风格设计。

随着互联网的不断成熟以及我们越来越多的用各种移动端的设备访问互联网，Web设计师和Web开发者的工作也变得越来越复杂，这个时候我们有必要选择一个好的前端框架与UI框架配合，达到减轻开发难度，提高开发效率，美观前端页面，更好管理前端模块的要求。

为了适应网站开发的快速高效的要求，我们后端采用了非阻塞式编程的Node.js作为支撑。

当用户将信息填写好，技术选择完毕后，我们后台就会对此进行一系列的处理。第一步：复制文件夹，第二步：根据基本信息和技术的选择改动配置文件，第三步：添加打包管理模块的相关功能，第四步：添加将ES6/ES7新语法转换为ES5的转换工具，第五步：压缩文件夹，提供用户下载。

### 2.2.3 打包管理模块功能

用户需要一个好的打包管理工具，对代码进行打包压缩混淆

随着当今互联网前端技术的疯狂增长，作为开发者使用的前端语言已经不只是JavaScript，CSS，html为了适应前端开发的快速高效性，JSX、TypeScript、coffee，less，sass，jade，dust等技术相应而生，但是我们最终要浏览器运行的还是JavaScript语言，这时，我们必须需要一款好的代码打包工具，将这些非JavaScript的语言能够打包成为浏览器可运行的JavaScript语言。

除此之外，客户在通过浏览器访问网站的时候，需要加载JavaScript代码，如果开发的JavaScript不进行压缩和混淆，会导致客户访问速度变慢以及安全性的问题，所以一款好的打包工具已经是现在web开发的必需品。

### 2.2.4 使用ES6/7新特性功能

用户需要能够使用ES6/ES7的新特性，需要能将ES6/ES7的语法转换成ES5的语法的工具

ES6/ES7其实是JavaScript的新标准，其中包含了大量的JavaScript的新特性，比如变量析构赋值，箭头函数，Promise等，这些新特性从语法层面上都将使Web开发变得更加简单，但是当今浏览器还不能完全兼容这些新的特性，为此我们需要一款能将ES6/ES7的语法转化成ES5的语法的工具。

## 2.3 系统可行性分析

可行性分析，是在该系统需求分析完成的基础上，估量系统的开发是否具有可行性，对该系统整个开发从技术、操作、经济的方面进行研究和分析，确认该系统是否可以开发成功以及实施后具备它该有的价值。该系统的可行性分析包括以下几个方面的内容：

1. 操作可行性：

本系统灵活方便，操作简单，界面友好，错误提示明确，便于使用人员快速熟悉和使用。

1. 技术可行性：

该系统对软、硬件系统无特殊要求，软件上，操作系统采用Windows系列操作系统，VScode作为前台开发工具以及Nodejs作为开发语言来完成的,后台使用Mongodb数据库，用基于express的kraken.js三大框架实现，这些都是现有的主流和成熟的开发工具，语言和框架，而且这些工具和语言都简单易学，开发人员可以较好掌握。同时，Node.js编程方便，界面清晰，技术比较成熟。因此在技术上是可行的。

## 2.4 研究方法与技术路线

研究方法：

首先调查现在互联网Web开发的常用技术，分析技术的特点以及学习难度，其次了解Web开发的模块打包管理工具，然后进行系统可行性分析与功能分析，然后进行详细设计，编写相关功能模块的源代码，随后进行细致修改，逐步完善系统功能；再进行系统测试，检查系统存在的错误与隐患，并且进行相应的修改，优化网站与Web模版，最终完成该系统。

技术路线：

（1）通过angularjs完成前端框架开发

（2）通过nodejs的express完成后端框架开发

（3）通过Webpack对各个模块进行管理

（4）通过CSS技术实现模板样式。

（5）通过nosql的mongodb对数据进行管理

# 第三章 系统概要设计

## 3.1 系统设计思路

现如今，web开发技术种类繁多，快速高效的开发乃是每个企业追求的目标，很多开发者不知道该如何选择技术，我从web开发效率以及开发成本的出发点上面考虑， web框架后端毅然决然的选择了异步IO的Node.js 。Node.js 的主要思路是：使用非阻塞的，事件驱动的 I/O 操作来保持在处理跨平台 (across distributed devices) 数据密集型实时应用时的轻巧高效。相比其他的后端语言Java，PHP等，Node.js显得更加轻量高效，同时学习成本也比较低。框架前端则是采用了现今比较流行的三个前端框架：Angular，Vue，React，用户可以根据自己的需要进行选择。UI框架这一块Bootstrap自然是必不可少的，我们还添加了一款基于Bootstrap的UI框架Flat UI，相比Bootstrap的CSS的样式，它的样式更具扁平化的特点。所以我集合多种web开发前沿的技术设计了这一套web开发的框架。

## 3.2 系统功能架构

### 3.2.1 Web网站架构

根据需求分析，本系统功能架构大致分为五个部分：关键技术介绍模块，选择技术生成框架模块，Web框架生成模块，用户下载使用框架模块，系统扩展模块。用户可以在我们的网站上面第一步了解相关技术，然后选择相应技术生成框架并下载，最后阅读框架的使用文档开始开发。如图所示：

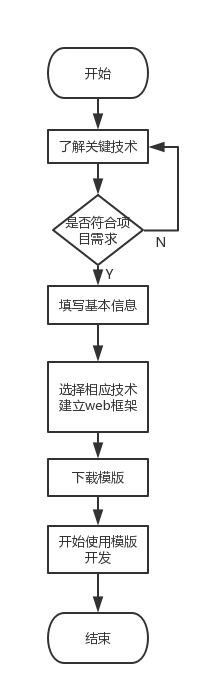


图3.2.1 Web网站功能流程图

### 3.2.2 Web框架架构

在设计框架时，主要遵从以下几个原则，如图所示。

图3.2.2.1 框架设计原则

* 易用性

在设计框架之初，我们就本着为软件开发企业设计一套使用简单方便的开发框架，通过Node.js的简单高效以及前后端统一使用JavaScript开发语言，极大提高了开发框架的易用性。

* 低成本

Web自动化开发框架极大的简化了开发流程，不仅提高软件的开发质量，更提高了软件的开发效率，降低了开发成本，这也是我们设计该框架根本的目的。

* 技术先进性

在框架设计中，我们采用了MVC 架构，MVC框架是用一种业务逻辑、数据、界面显示分离的软件架构，将业务逻辑聚集到一个部件里面，在改进和个化定制界面及用户交互的同时，不需要重新编写业务逻辑。其次，我们都采用了现在Web开发的主流技术AngularJs，Vue，React，Node.js，Bootstrap等开发了一套轻量级Web开发框架。应用该开发框架可以有效提升软件开发效率，降低开发成本。

* 可扩展性

利用我们的开发框架，开发人员可以根据自己的项目需求添加相应Web模块。

在框架的开发中，我们的后端主要围绕控制器、视图、模型进行组件的划分，前端主要是MVVM架构的体现，Web框架MVC架构如图所示

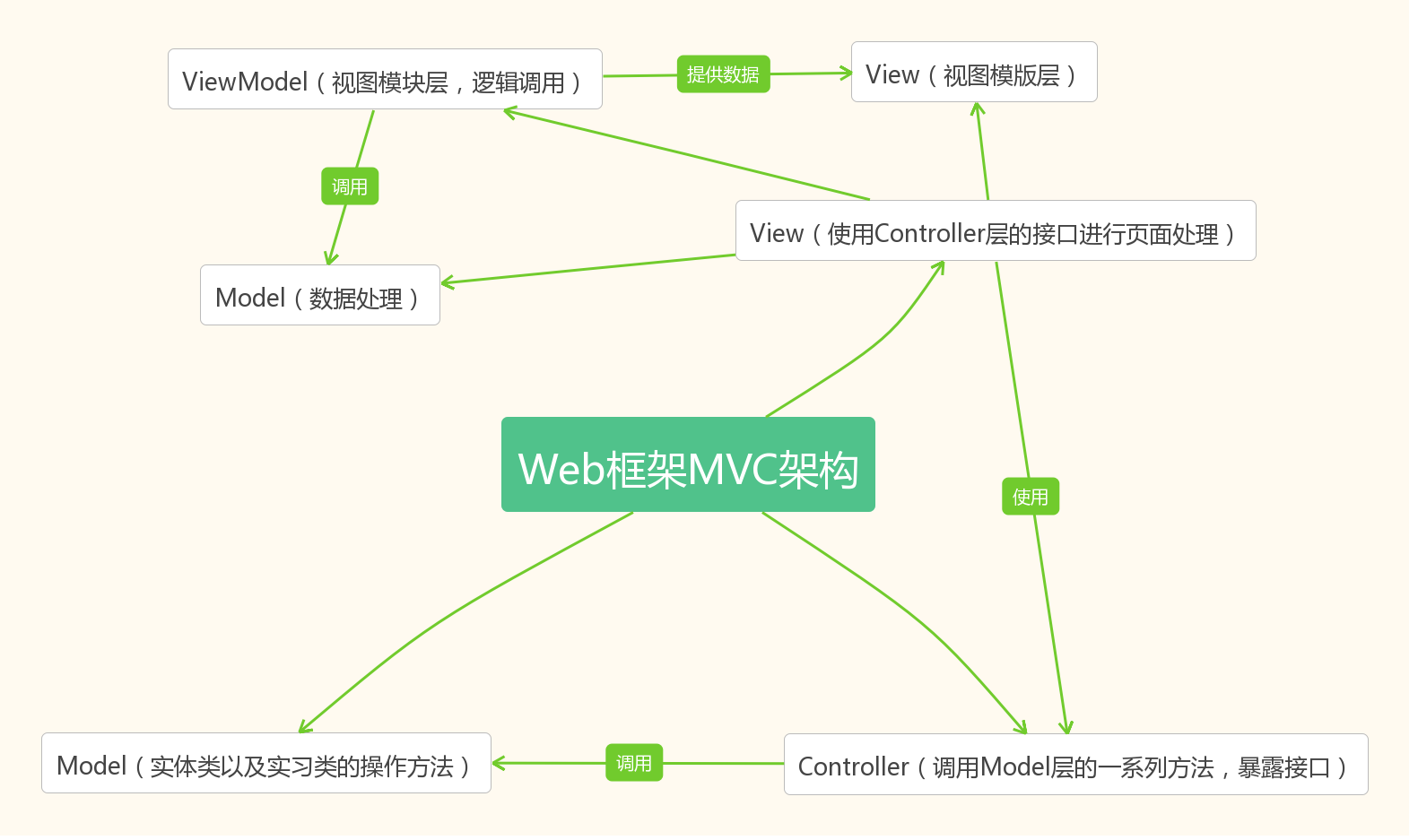


图3.2.2.2 Web框架MVC架构图

后端MVC全名是Model View Controller，是模型(model)－视图(view)－控制器(controller)的缩写，它是一种软件设计典范，用一种业务逻辑、数据、界面显示分离的方法组织代码，将业务逻辑聚集到一个部件里面，在改进和个性化定制界面及用户交互的同时，不需要重新编写业务逻辑。MVC用于映射传统的输入、处理和输出功能在一个逻辑的图形化用户界面的结构中，MVC关系如图所示。

图3.2.2.3 MVC关系图

* 视图

视图是用户看到并与之交互的界面。对传统的Web应用程序来说，视图就是由HTML元素组成的界面，在最新的Web应用程序中，HTML依旧在视图中扮演着重要的角色。MVC的优点是它能为应用程序处理很多不同的视图。视图并不真正处理数据，它只是作为一种输出数据并允许用户操作的方式。

* 模型

模型表示企业数据和业务规则。在MVC的三个部件中，模型拥有最多的处理任务。例如它可能用像EJBs和ColdFusion Components这样的构件对象来处理数据库，被模型返回的数据是中立的，就是说模型与数据格式无关，这样一个模型能为多个视图提供数据，由于应用于模型的代码只需写一次就可以被多个视图重用，所以减少了代码的重复性。

* 控制器

控制器接受用户的输入并调用模型和视图去完成用户的需求，所以当单击Web页面中的超链接和发送HTML表单时，控制器本身不输出任何东西。它只是接收请求并决定调用哪个模型构件去处理请求，然后再确定用哪个视图来显示返回的数据。

前端MVVM 的全名为Model-view-viewmodel，它是一种软件架构模式。MVVM有助于通过开发业务逻辑或后端逻辑（数据模型），分离图形用户界面的开发（无论是通过标记语言还是GUI代码）。 MVVM的视图模型是一个值转换器; 意味着视图模型负责从模型中显示（转换）数据对象，使得对象容易管理和呈现。在这方面，视图模型比视图更模型，并处理大部分（如果不是全部）视图的显示逻辑。视图模型可以实现调解器模式，组织对视图支持的一组用例的后端逻辑的访问。MVVM是Martin Fowler的演示模型设计模式的一个变体。 MVVM以同样的方式提取视图的状态和行为，但是Presentation Model会以不依赖于特定用户界面平台的方式抽象视图（创建视图模型）。MVVM和Presentation Model都来源于模型 - 视图 - 控制器模式（MVC）

* 模型

模型是指代表真实状态内容（面向对象的方法）的域模型，也可以是代表内容（以数据为中心的方法）的数据访问层。

* 视图

与MVC和MVP模式一样，视图是用户在屏幕上看到的结构，布局和外观。 （UI）

* 视图模型

视图模型是暴露公共属性和命令的视图的抽象。代替MVC模式的控制器或MVP模式的演示者，MVVM具有绑定器。在视图模型中，粘合剂介导视图和数据绑定器之间的通信视图模型已被描述为模型中数据的状态。

* 绑定器

声明式数据和命令绑定在MVVM模式中是隐含的。在Microsoft解决方案堆栈中，binder是一种名为XAML的标记语言。绑定器使开发人员不必编写锅炉板逻辑来同步视图模型和视图。当在Microsoft堆栈之外实现时，声明性数据绑定技术的存在是该模式的关键推动因素。

# 第四章 系统详细设计

详细设计阶段的根本目标是确定应该如何具体地实现所要求的系统，经过该阶段的设计工作，应该得出对目标系统的精确描述，详细设计的结果决定了这个项目最终的质量。下面根据概要设计的五大功能模块依次描述。

## 4.1 关键技术介绍模块

首先要对当今互联网上面的Web开发技术做一个采集，然后根据使用的流行程度，技术的特点以及学习的难度筛选出我们要使用的技术。再将这些技术的存放的我们的Mongodb数据库中，前端在通过AJAX从后端暴露的web开发技术的接口中拿到数据，前端在通过AngularJS和Bootstrap框架将这些技术展现出来，这样用户就可以根据自己的需要了解并学习这些技术，为以后的选择技术搭建框架做准备。

本地Mongodb数据库数据表如下：

Web开发技术表：

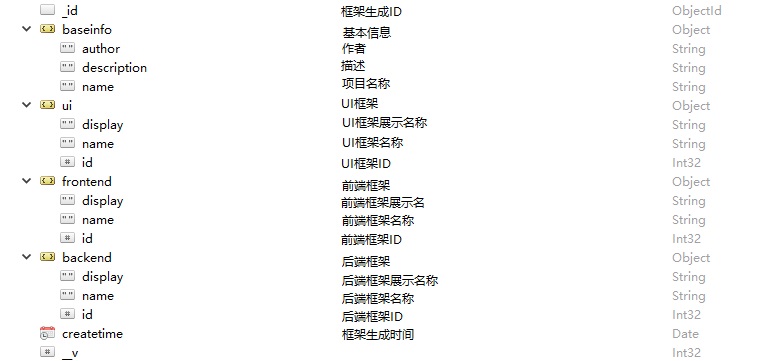
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | | 数据类型 | | 含义 |
| \_id | | ObjectId | | 技术ID |
| Name | | String | | 技术名称 |
| Detail | | String | | 技术详情 |
| Href | String | | 技术官网链接 | |
| Kind\_id | number | | 技术类别 | |

## 4.2 选择技术生成框架模块

在这个模块上面，用户需要根据自己的需求，填写相应的信息，并通过按钮点击选择相应的技术，然后通过AJAX的POST请求发送到数据库，这样我们的框架生成表就能够添加一条记录。

本地Mongodb数据库数据表如下：

框架生成表：

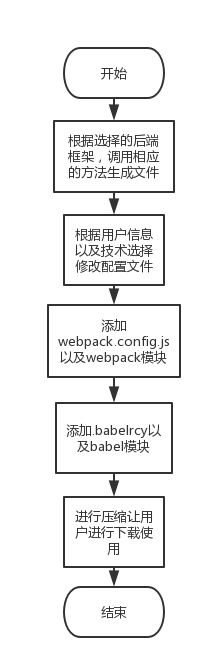


## 4.3 Web框架生成模块

这一个模块就是最重要的一部分，就是根据用户的信息以及选择的技术生成相应的Web框架。

1. 根据用户选择的后端框架，调用相应的后端的方法，然后生成相应的文件夹。
2. 根据用户填写的信息以及选择的技术，在生成出来的文件夹里面修改配置文件（package.json），添加用户选择的相应技术模块。
3. 为了让用户更好的对代码进行打包和管理，我们在配置文件（package.json）中添加了Webpack模块，以及在项目中添加Webpack的配置文件webpack.config.js，它不仅能对JavaScript代码进行压缩混淆，还能把类JavaScript语言代码转换成JavaScript代码，以及将Less/Sass转换成CSS文件。
4. 为了能让用户使用ES6/ES7的新特性，我们在配置文件（package.json）中添加了babel的相应模块，以及添加babel配置文件.babelrc，将ES6/ES7的代码转换成浏览器可以运行的ES5的代码。
5. 最后我们将生成的文件夹打成Zip压缩包，提供给用户进行下载使用。

流程图如下：



## 4.4 用户下载使用框架模块

用户通过我们的Web站点下载好创建的Web的框架之后，首先安装好Node.js环境，然后在通过NPM安装各个模块，然后仔细阅读其中的文档，了解各个目录的意义以及在项目中的作用，这样才能够更快速的使用这套Web框架开发站点。

## 4.5 系统扩展性设计模块

我们的这套Web框架不局限于网站上面陈列的技术，如果以后会有另外的技术添加进来，我们只需在数据库中添加这些技术，然后在resource文件夹中生成好目录结构，如果是后端框架，则会根据这个后端框架再生成一套相应的目录结构，如果是前端框架或者UI框架，只需要修改项目配置文件（package.json）。用户也可以自定义添加技术，只需根据上面的步骤进行操作。

## 框架设计模块

首先我们的Web框架完全是前后端分离的，易于前后端各个模块的分离，满足了软件开发中各个模块的高内聚低耦合的要求，这样各个模块才能易于管理与代码的重用，也能够更好的实现多人合作开发的需求，完全满足了当今Web网站开发的潮流与趋势。

### 4.6.1 框架前端设计

Web框架前端采用的MVVM架构，框架图如下所示：

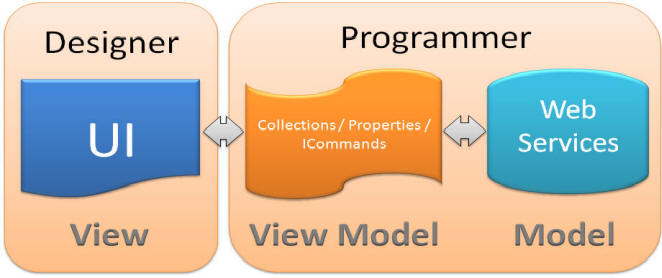


图4.6.1 MVVM框架

* Model层主要为应用程序提供数据，它能够通过AJAX向后端接口发出请求从而获取数据或者其他一系列操作
* View Model主要的就是负责联通View层和Model层，从Model层拿到的数据需要做一些业务逻辑的处理才能够显示到View层上面。
* View层主要是负责页面的排版和布局，将View Model层处理完的数据展现到前台页面上。

### 4.6.2 框架后端设计

Web框架后端采用MVC架构，具体目录结构如下图所示

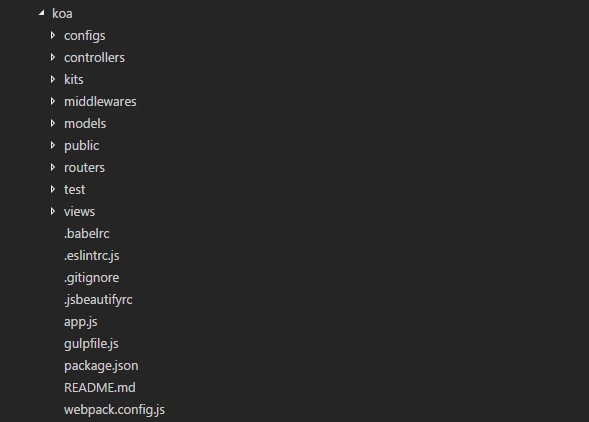


图4.6.2 框架后端目录图

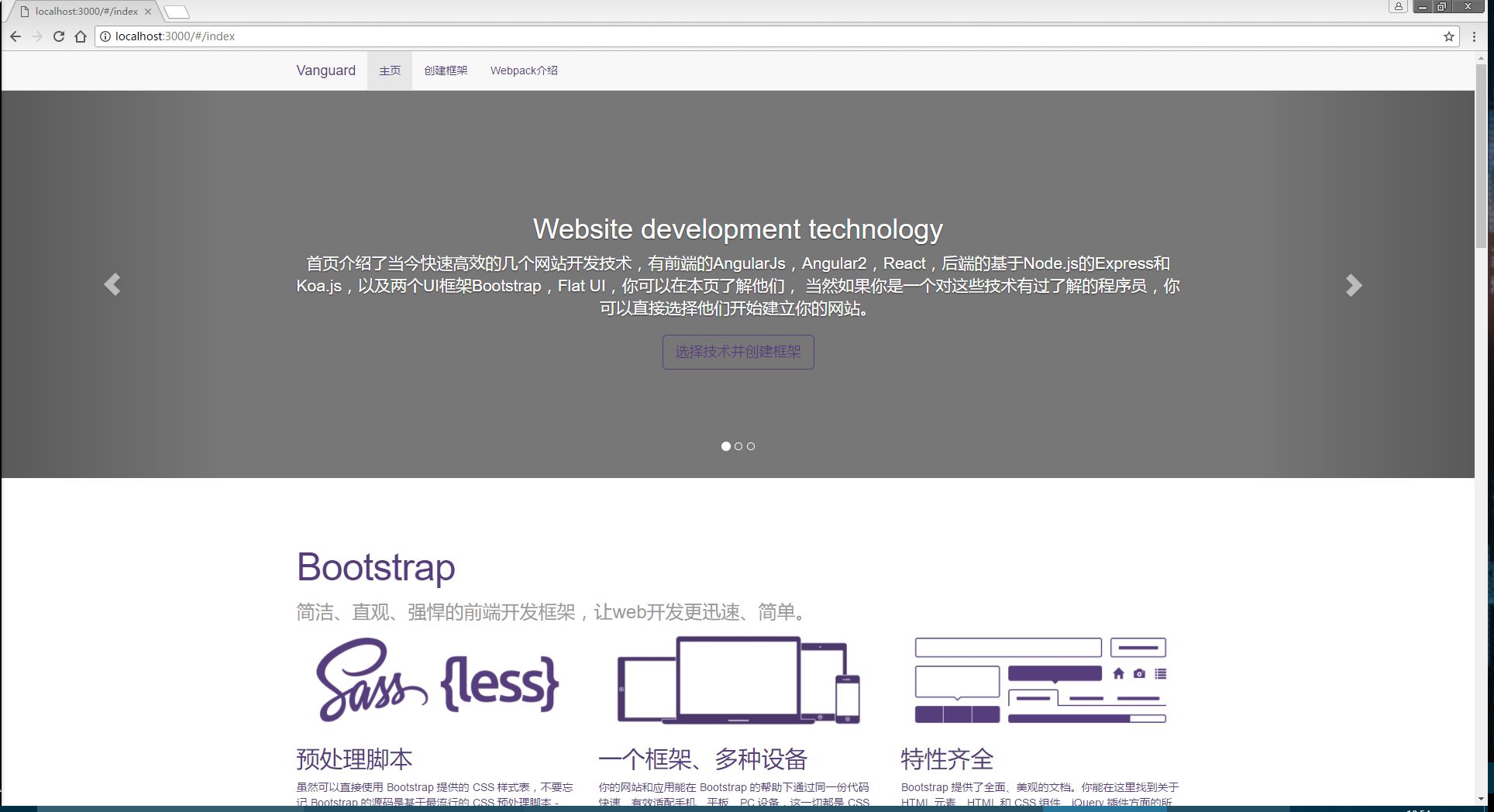
从上至下，介绍该目录结构：

1. Configs 这里主要是框架后端的一些配置，比如数据库的配置，中间件的配置，开发模式的配置等等。
2. Controllers 这里主要是暴露给前端的一些接口，可供前端通过http请求访问到数据。
3. Kits 这里是一些集成的软件包
4. Middlewares 这里是Node.js的特有的中间件配置，里面可以存放Node.js的一些中间件，比如像用户权限认证的一些中间件等。
5. Models 这里存放的一些数据模型和数据的一些操作方法。
6. Public 这里存放一些项目的静态文件，一些图片，CSS，JavaScript代码都可以放在这个目录下。
7. Routers 这里主要是暴露给前端的一些路由接口，可供前端通过http请求访问。
8. Test 集成测试和单元测试吗目录
9. Views 这里是一些前端的模版文件，比如dust，jade等
10. .babelr 这里就是babel的配置文件，里面可以将指定的ES6/7的代码转换成ES5的代码，这样浏览器可以兼容执行。
11. Webpack.config.js 这里是webpack的配置文件，可以将配置好的文件代码进行压缩混淆，也可以加载一些解析器，完成一些特定的功能，比如将less/sass转换成CSS。

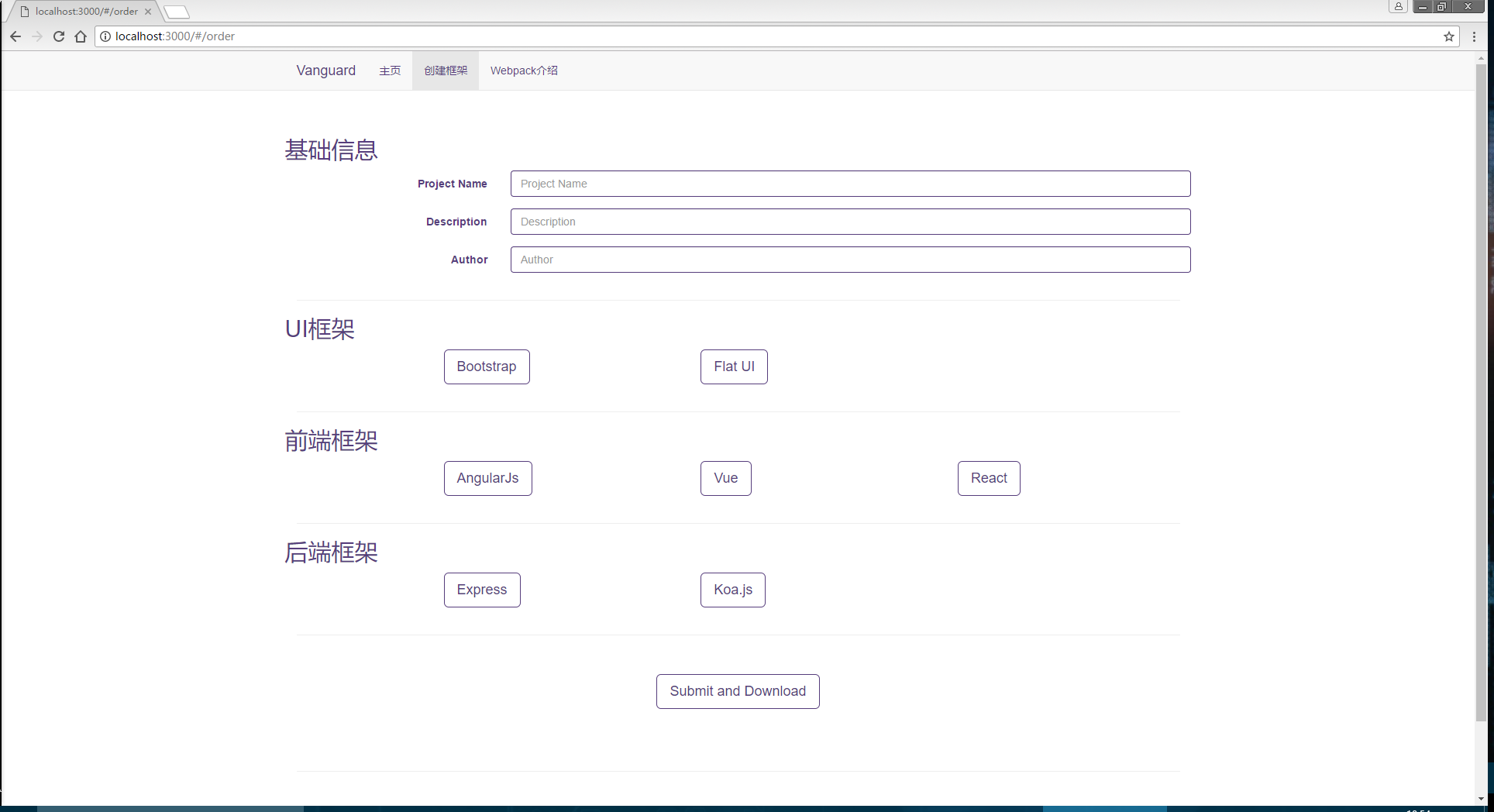
# 第五章 系统实现

## 5.1 系统展示

1. 技术介绍页面



1. 创建框架页面



1. Webpack介绍



## 5.2 系统编码

Web站点

1. 前端开发用AngularJs和Bootstrap结合，具体代码如下：
   1. 路由模块：

$stateProvider

.state('index', {

url: "/index",

views:{

'title':{

templateUrl: '/app/carousel.html',

},

'content':{

templateUrl: '/app/index.html',

}

}

})

.state('order', {

url: "/order",

views:{

'content':{

templateUrl: '/app/order.html',

}

}

})

* 1. 前端页面：

<h2 class="row">UI框架</h2>

<div class="row col-xs-offset-2">

<div class="form-group">

<div class="col-xs-4" ng-repeat="ui in uis">

<a class="btn btn-lg btn-outline" ng-class="{'selected': order.ui==ui}" ng-click="order.ui=ui">{{ui.display}}</a>

</div>

</div>

</div>

<hr/>

<h2 class="row">前端框架</h2>

<div class="row col-xs-offset-2">

<div class="form-group">

<div class="col-xs-4" ng-repeat="frontend in frontends">

<a class="btn btn-lg btn-outline" ng-class="{'selected': order.frontend==frontend}" ng-click="order.frontend=frontend">{{frontend.display}}</a>

</div>

</div>

</div>

<hr/>

<h2 class="row">后端框架</h2>

<div class="row col-xs-offset-2">

<div class="form-group">

<div class="col-xs-4" ng-repeat="backend in backends">

<a class="btn btn-lg btn-outline" ng-class="{'selected': order.backend==backend}" ng-click="order.backend=backend">{{backend.display}}</a>

</div>

</div>

</div>

* 1. 前端样式：

.btn-outline {

color: $color1;

background-color: transparent;

border-color: $color1;

}

.btn-outline:active, .btn-outline:focus, .btn-outline:hover {

color: #fff;

background-color: $color1;

border-color: $color1;

}

.selected{

color: #fff;

background-color: $color1;

border-color: $color1;

}

.form-control{

color: $color1 !important;

border: 1px solid $color1;

}

.form-control:focus{

border-color: $color2;

}

.carousel-caption{

padding-bottom: 100px;

}

1. 后端用Node.js开发，建立restful Api供前端ajax访问，数据库为mongodb：
   1. 接口路由：

router.get('/', function(req, res) {

book.find(function(err,doc){

res.json(doc);

})

});

router.post('/add', function(req, res) {

var newBook=new book(req.body.order);

newBook.save(function(err,doc){

if(err){

res.json(err);

}else{

res.json(newBook);

}

})

});

* 1. 底层数据库操作：

var mongoose = require('mongoose');

var book = require('../../models/module').order;

mongoose.connect('mongodb://localhost/hello');

var newBook=new book(req.body.order);

newBook.save(function(err,doc){

if(err){

res.json(err);

}else{

}

})

* 1. 文件操作：

express:function(order,callback){

var time=Date.parse(new Date());

flieService().copyfile(path+order.backend.name,path+order.backend.name+time,function(){

var packagejson=require(path+order.backend.name+time+"\\package.json");

console.log(packagejson.dependencies);

packagejson.name=order.baseinfo.name;

packagejson.description=order.baseinfo.description;

packagejson.author=order.baseinfo.author;

packagejson.dependencies="";

packagejson.dependencies="";

fs.writeFile(path+order.backend.name+time+"\\package.json", JSON.stringify(packagejson).replace(/,/g,",\n"), function(err) {

if (err) {

throw err;

}else{

flieService().zip(order.backend.name+time,path+order.backend.name+time,function(){

callback(order.backend.name+time);

})

}

});

})

},

# 第六章 结论

## 全文总结

通过三个多月的努力，我顺利完成了《基于Node.js的Web框架生成工具》毕业设计。在这次毕业论文设计过程中我通过上网查询，翻阅书籍，向老师和同学请教相关知识等方法解决了很多遇到的问题，学习到了很多知识，同时也认识到自己在技术方面有很多的不足，认识到理论结合实践的重要性，认识到在开发网站时应事先计划好开发的每一步骤。

本系统按照需求分析的功能基本实现，但是还有些功能需要优化，通过这段时间的毕业设计的实现及别与论文的撰写，我深知自己还有许多的方面需要学习和改正，在以后的学习中，会努力改正自己的不足之处。本系统还存在不足之处，恳请老师批评指正，我会继续完善。

## 展望

本课题在基于Node.js的Web框架生成工具做了大量的工作，但是由于个人的时间与精力，系统还有一些不足，现对未来可以完善系统的工作做出几点展望：

1. 本系统生成的框架还不够强大，还不能实现大多数项目需要实现的一些基本功能。
2. 本系统收集的暂时还是Web开发较为流行的一些技术，有些新技术还未收集齐全。

相信随着研究的深入，这些问题将会得到很好的解决，相信基于Node.js的Web框架生成工具这套系统会有很好的发展前景。

# 参考文献

* 1. 黄扬子 基于NodeJS平台搭建REST风格Web服务 无线互联科技 2015-08-25
  2. 郑杰 AngularJS前端框架与后端代码自动生成系统的设计与实现 西南交通大学 2017-05-10
  3. 冯晨超 基于AngularJS的物品和用户查询模块的设计与实现 南京大学 2015-05-01
  4. 张文盛; 郑汉华 基于MongoDB构建高性能网站技术研究 吉林师范大学学报(自然科学版) 2013-02-10
  5. 黄悦深 基于HTML5的移动Web App开发 图书馆杂志 2014-07-15
  6. 杨俊; 李艳梅 JavaScript面向对象编程探析 办公自动化 2010-04-15
  7. 张俊兰; 刘翼; 铁宏军 基于CSS技术的网页设计应用研究 延安大学学报(自然科学版) 2010-09-20
  8. 基于jQuery框架的页面前端特效的设计与实现 周玲余 计算机与现代化 2013-01-15
  9. Seonghoon Kang;Sukyoung Ryu ACM SIGPLAN Notices, 2012, Vol.47 (10), pp.621-638 Association for Computing Machinery Journal
  10. Lubbers P,Albers B,Salim F Pro HTML5 Programming:Pow-erful APIs for Richer Internet Application Development 2010

# 致谢

在这三个多月的时间内，我能成功的将此毕业设计做出，这要归功于朱晓辉老师严格的要求、合理的时间安排以及程序的技术支持。老师在这期间对我的帮助和指导，更令我受益匪浅，在此表示深深地感谢。本文是在朱晓辉老师的关心和指导下完成的，在论文完成过程中，本人还得到了其他同学与朋友的热心帮助，本人向他们表示深深的谢意！