四川农业大学

本科毕业论文（设计）

（2018 届）

|  |  |
| --- | --- |
| 题 目： | 基于node.js的文档框架的研究与实现 |
| 学 院： | 信息工程学院 |
| 专 业： | 计算机科学与技术 |
| 学生姓名： | 许娜 学号： 20158867 |
| 导师： | 潘勇浩 职称： 讲师 |
| 完成日期： | 2019 年 3月 10 日 |

目 录

1 绪论 2

1.1 选题背景 2

1.2 研究现状 2

2 需求分析 2

2.1 项目需求 2

2.2 需求的提出 3

3 项目可行性分析 4

3.1 项目可行性 4

3.2 技术可行性 4

4 技术实现路线 4

4.1 相关理论知识 4

4.2 相关技术知识 5

4.3 开发环境 5

4.4 Express的MVC模式 5

4.5 微信小程序的MVVM模式 6

4.6 最终开发模式的确立 6

5 总体设计 7

5.1 框架分层 7

5.2 框架流程控制 7

6 详细设计和实现 8

6.1 开发环境及工具详情 8

6.2 微信小程序端的实现 8

6.2.1 小程序框架概述 8

6.2.2 具体实现 9

6.3 服务器端的实现 11

6.3.1 Express框架概述 11

6.3.2 具体实现 11

7 程序测试 17

7.1 界面测试 17

7.1.1 微信小程序界面兼容测试 17

7.1.2 管理员Web界面兼容性测试 17

7.2 功能模块测试（以用户注册为例） 18

7.2.1 程序核心部分代码 18

7.2.2 邮件发送成功界面 21

7.2.3 验证码接收界面 21

7.2.4 注册成功 22

8 系统存在问题及其原因以及解决办法 23

8.1 小程序网络请求API次数限制 23

8.2 网络请求函数过于集中 23

8.3 NodeJS多重回调嵌套问题 23

9 结论 23

参考文献 24

致谢 25

基于node.js的文档框架的研究与实现

计算机科学与技术专业 许娜

导师 潘勇浩

**摘要**：随着微信（WeChat）应用的不断发展，它作为一个为智能终端提供及时通讯服务的免费用程序。到目前为止已经覆盖中国94%以上的智能手机，据统计，2016年一季度，微信月活跃数达8.06亿[1]。在一个大用户基数的平台下来实现能够提升人们的工作办事效率的应用程序，是一个完美的解决实际问题的可行的解决方案。本文将从应用前台技术（微信小程序）、应用服务后台技术（NodeJS）、数据持久化技术（MySQL）等方面来论述，讲解通过提供的技术来实现高效的有意义的数据的增、删、查、改。来实现有实际应用意义的应用程序，提升人们的工作效率。

**关键词**：微信小程序；NodeJS；Mysql；志愿服务

**Design and Implementation of Volunteer Service Platform**

**on Wechat Applet**

Computer Science and Technology Bowen Liu

Directed by HUANG Qiang

**Abstract**: With the continuous development of the WeChat application, it serves as a free program for providing timely communication services to intelligent terminals. So far it has covered more than 94% of smart phones in China, and its monthly active number has reached 806 million. It is a feasible solution to solve practical problems by implementing an application that can improve the work efficiency of people on a large user base platform. This article discuss the application foreground technology (WeChat Applet), application service backend technology (NodeJS), data persistence technology (MySQL) and other aspects, and explain how to achieve effective and meaningful data addition, deletion, Check, Change through the provided technologies. To achieve practical application applications, to improve people's work efficiency.

**Key words**: WeChat applet; NodeJS; Mysql; volunteer service

1 绪论

1.1 选题背景

在如今信息暴涨的互联网时代，网络成为了时代的代名词，为了缩短人为的工作时间，提升我们的工作效率，不断的有各种替代线下平台的线上服务平台，种类繁多，层出不穷。同样，对于一个地区的志愿者组织也是一样，通过调研显示，很多地区志愿者服务存在志愿者和志愿服务任务不能及时对接、流动志愿者的集中管理等问题。为了克服以上问题，各个地区的志愿者组织也相应的建立起自己的志愿服务平台（Volunteer Service Platform）。但其中以门户展示为主的网站平台居多，少有能够将志愿服务的日常工作线上化。且，缺少移动端的支持，对于移动设备（mobile phone）盛行的今天，这是急需 改善的一个现状。

1.2 研究现状

志愿服务线上化是一个趋势，目前真正将志愿服务线上化的组织很少，能够将其移动化的组织更少。志愿服务依靠固定的志愿服务组织的力量是远远不够的，面对大量流动志愿者，缺少的是利用互联网平台将他们联系集合起来。单一网站平台远远不足，但面对移动手机app（application）的开发，存在着开发成本高，开发难度大，版本更新过程复杂等问题。在UE（用户体验）方面，现在的人们更加容易接受的方式是扫描一个二维码或者打开一个链接就能够使用到手机App的体验。手机应用（Application）现有的开发模式分为三种。主要实现方式有本地应用（Native App) 、Web 应用（Web App) 和混合应用（Hybrid App) 三种模式[2]。 Native App即原生应用。而在微信生态下的小程序，触手可及、用完即走的特点，加上其与手机硬件的多样交互接口（JSSDK）。使得其能够同时满足多方面的需求，不失为一种好的移动应用开发的解决方案。

2 需求分析

2.1 项目需求

从项目实际应用的角度出发，我们需要是实现的大体目标是，以雅安地区为项目试验地区，通过本项目能够有效的联系志愿组织和志愿者，能够及时的发布和完成雅安地区的志愿者任务。为了鼓励更多其他职业的人加入到我们的志愿服务中来，例如：学生等。本应用采取了鼓励措施，以完成志愿服务换取虚拟金币兑换上传到平台上的可兑换物品，如：签字笔，牛奶等。这样可大大提升志愿者和非志愿者的服务热情，在提升工作效率的同时，也能够提升志愿服务工作质量。

2.2 需求的提出

功能需求：

* 用户登录注册、密码修改、密码找回
* 用户接受任务、上传任务图证
* 用户兑换实物商品
* 志愿组织发布志愿任务
* 管理员审核当前上传待发布任务（由志愿组织上传的任务）
* 管理员发布志愿任务（管理员直接发布任务）
* 管理员审核图证
* 管理员发布实物退换平台物品
* 管理员管理平台用户

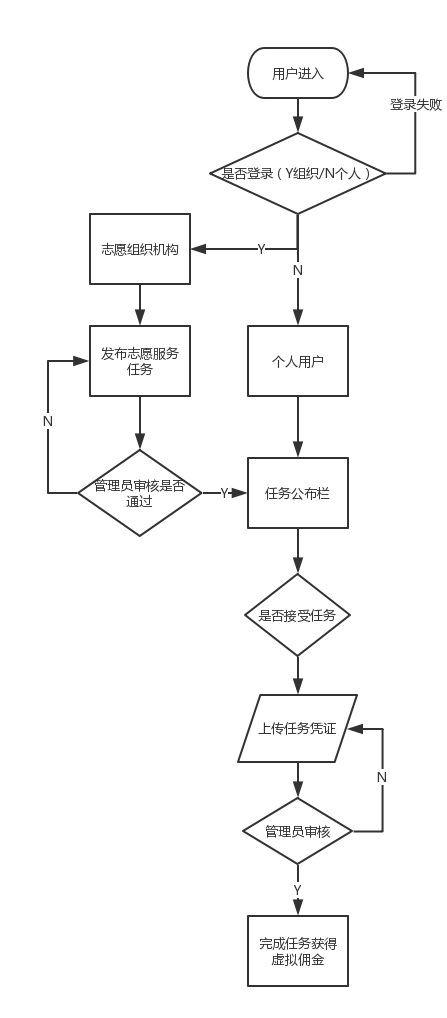


图 1 功能流程图

开发需求：

* 前台应具有良好的用户界面和用户体验设计
* 后台开发应保持低耦合，便于后期的业务扩展
* 前后台约定数据传输类型，便于数据通信

3 项目可行性分析

3.1 项目可行性

志愿服务是几乎是每个文明社会不可缺少的一部分，各个国家，各地政府都不断的重视其当地的志愿工作，所以，一个移动化的志愿服务平台在当今社会是可行的、可适用的、可被人们接受的、可推广的。

3.2 技术可行性

本志愿服务平台应用采用微信小程序，web两个客户端进行平台的业务逻辑以及展示功能的体现。以基于Node.js（Node.js 是一个基于 Chrome V8 引擎的 JavaScript 运行环境，使用了一个事件驱动、非阻塞式 I/O 的模型，使其既轻量级又有效[3]。其中的npm包管理平台更是为nodeJS的操作带来了巨大的方便性）的MVC框架Express作为服务器的开发。并且以MongoDB和MySQL作为数据库的持久化保存也是可行的。 NodeJS 事实上既是一个运行时环境，同时又是一个库[4]。其适合的适用场景为应用提供高可用的场景，有研究表明运行NodeJS的服务器能支持数万个并发连接，这 差不多是PHP语言编写的服务器程序处理能力的两倍[5]。故，在技术层面来说是可行的。

4 技术实现路线

项目所用技术包括：HTML、CSS、JavaScript、微信小程序MINA框架、NodeJS、 MySQL 、Express（基于NodeJS的MVC框架）

4.1 相关理论知识

（1）数据库原理与应用

（2）数据结构

（3）软件工程

（4）超文本传输协议（HTTP，HyperText Transfer Protocol）

（5）异步编程理论基础

4.2 相关技术知识

（1）MINA 微信小程序框架开发技术

（2）ECMAScript6.0 相关编程知识和技术

（3）Express（基于nodeJS的MVC框架）开发技术

（4）WebSocket 实时通信技术

（5）Ajax异步请求技术

（6）异步编程技术

4.3 开发环境

（1）操作系统：MacOS Sierra 64-bit

（2）开发工具：WebStorm、Atom、WorkBench、微信web开发工具

（3）浏览器：Google浏览器、Safari浏览器、Edge浏览器

（4）测试工具：微信客户端、IETest

4.4 Express的MVC模式

MVC是一种使用MVC(ModelViewController模型—视图—控制器)设计创建Web应用程序的模式：即Model-View-Controller[6]。Model是描述业务逻辑，业务模型、数据操作、数据模型等一系列类的集合，定义了数据修改和操作的业务规则。View表示UI组件[7]。由Express生成项目时产生的views负责，本项目用使用的是EJS模板引擎，主要负责接收用户的输入和数据如何显示。 Controller负责处理用户交互和数据处理。即用户通过视图向控制器发出请求，由控制器选择相应的模型来处理，最后把结果返回给View。其优点主要是：耦合性低，扩展性高，重用率高，可维护性高。其缺点主要是：增加了系统结构和实现的复杂性，Controller比较臃肿，模型的效率低。

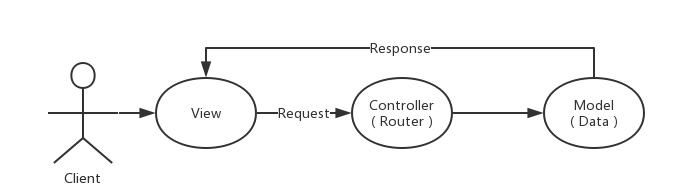


图 2 MVC模式流程图

4.5 微信小程序的MVVM模式

MVVM（Model-View-ViewModel）开发框架是通过 MVP（Model-View-Presenter） 模式与 WPF 结合而产生的一种新型的架构框架[8]。小程序的开发框架（MINA）的目标是通过尽可能简单、高效的方式让开发者可以在微信中开发具有原生 APP 体验的服务。框架提供了自己的视图层描述语言 WXML 和 WXSS，以及基于 JavaScript 的逻辑层框架，并在视图层与逻辑层间提供了数据传输和事件系统。其核心是一个响应的数据绑定系统，与我们的传统网页的开发模式不同，传统网页在客户端与服务器取得数据通信后，需要开发者进行DOM操作将新的数据更新到网页上。微信小程序的框架类似于现在的MVVM框架VUE.js。只要客户端与服务器通信获取到新的数据时，客户端的UI会自动进行一个更新，这就是我们所说的数据驱动页面更新，即Model-View-View Model。这个模式提供了对View和View Model的双向数据绑定。该机制使得View Model的状态改变可以自动传递给View。MVVM的优点主要是：能够实现业务逻辑 ，显示逻辑和用 户界面的分离。带来的好处是使得我们的程序代码 结构清晰 ，易于阅读 、测试 、维护 、替换 、扩展和改进[9]。MVVM的缺点主要是：双向绑定使得异常情况很难被调试，双向绑定附加的事件监听机制带来额外的内存消耗，对于初次开发者来说其开发逻辑也是不友好的。

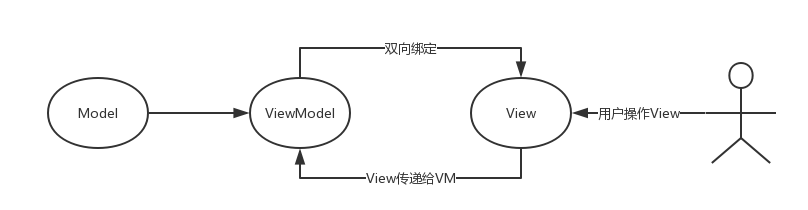


图 3 MVVM模式流程图

4.6 最终开发模式的确立

由于微信小程序的实现同样需要后台语言的支持，为我们所熟知的后台开发语言有Java、ASP.NET、Python、PHP等。每一种语言都有其所对应的MVC模式，例如Java的Spring MVC、Python的Django MVC、THINKPHP等MVC框架。但为什么我们需要选着以NodeJS语言为基础的Express MVC框架为该项目的后台开发平台尼。因为微信小程序是基于JavaScript语言来开发的，而我们的NodeJS也是基于JavaScript这门语言来开发的。我们都知道在前后台进行数据通信的时候，往往需要我们来规定前后台的数据通信的格式。然后，在以往的开发经验中发现，不同系的语言之间需要将数据格式作多重转换，比较费时费力。故该项目采取的是都以JavaScript这门语言为编程语言的前后台平台（前端：微信小程序MINA框架、后台：Express MVC框架）来实现整个项目。

5 总体设计

5.1 框架分层

本项目的前台和后台分别使用的是微信小程序的MINA（MVVM模式）开发框架和基于NodeJS的后台开发框架Express（MVC模式）。MVVM模式分为三层，分别为Model层：数据计算层、View层：视图显示层、View-Model：数据双向绑定层。MVC模式分为三层，分别为Model层：数据计算层、View层：视图显示层、Controll层：控制层（路由层）。

5.2 框架流程控制

流程控制分为两部分，前台开发的微信小程序MVVM的流程控制，如图4。后台开发MVC的流程，如图5。

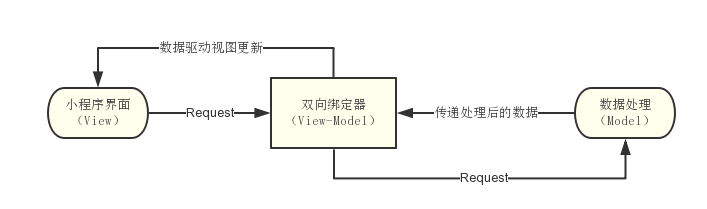


图 4 小程序流程控制图

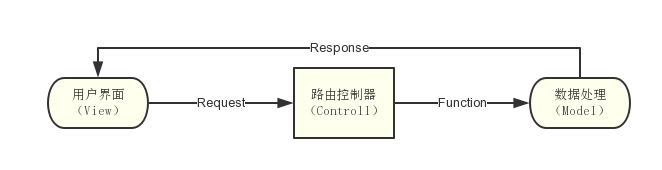


图 5 后台开发流程控制图

6 详细设计和实现

6.1 开发环境及工具详情

本设计在MacOS Sierra操作系统上开发完成，使用WebStorm作为建立软件后台开发工程、编写程序代码的主要开发工具（WebStorm 是jetbrains公司旗下一款JavaScript 开发工具。被广大中国JS开发者誉为“Web前端开发神器”、“最强大的HTML5编辑器”、“最智能的JavaScript IDE”等[10]。与IntelliJ IDEA同源，继承了IntelliJ IDEA强大的JS部分的功能。其提供的拼写检查和代码分析，极大的解决JavaScript开发不报错、调试难的痛点[11]）。且，WebStorm能够扩展各种框架，都能够做到代码高亮，错误提示等，这是作为JavaScript系开发的一个必备的开发工具。微信小程序的开发则直接使用微信官方提供得开发工具-微信web开发者工具进行开发。

由于考虑到项目中Web部分需要运行在不同的浏览器当中，因此选用比较有代表性的4款浏览器作为了运行环境。浏览器的内核主要有四大内核，分别是Tirdent、Geckos、Presto、Webkit、 KHTML[12]。Chrome浏览器（浏览器版本48.0.2564.109，内核版本WebKit 537.36）、Edge浏览器（浏览器版本 25.10586.0.0，内核版本Edge 13.10586）、IE浏览器（浏览器版本 11.545.10586.0，内核版本Trident/7.0）

6.2 微信小程序端的实现

6.2.1 小程序框架概述

小程序开发框架的目标是通过尽可能简单、高效的方式让开发者可以在微信中开发具有原生 APP 体验的服务。框架提供了自己的视图层描述语言 WXML 和 WXSS，以及基于 JavaScript 的逻辑层框架，并在视图层与逻辑层间提供了数据传输和事件系统，可以让开发者可以方便的聚焦于数据与逻辑上。框架的核心是一个响应的数据绑定系统。整个系统分为两块视图层（View）和逻辑层（App Service）框架可以让数据与视图非常简单地保持同步。当做数据修改的时候，只需要在逻辑层修改数据，视图层就会做相应的更新。这就是现在前端开发的主流开发模式MVVM的实现，即数据驱动视图（UI）更新。这样做更加方便了开发人员，开发人员不用再去关注数据更改后视图的变化，只用关注于数据逻辑。

6.2.2 具体实现

小程序包含一个描述整体程序的app和多个描述各自页面的page，一个小程序的主体包括三个部分（其都必须放在根目录下）：app.js、app.json、app.wxss。一个小程序的页面主要由四个文件组成：page.js、page.wxml、page.wxss、page.json。在每个页面中.wxml文件利用框架中的组件负责实现UI部分，.wxss文件负责描述WXML的组件样式，而.js文件则负责最重要的部分，通过页面的不同的生命周期函数以及框架提供的API与服务器通信，进行数据的增加，删除，查询，修改并通过数据双向绑定机制驱动页面组件进行UI更新。而.json文件则是相关页面的配置文件。

部分示例代码：

.WXML示例代码：

<scroll-view scroll-y="true" class="scrollView">

<view class="taskbox" wx:for="{{info}}" wx:key="\*this">

<view id="{{item.id}}" bindtap="taskOk">

<view class="box name" id="{{item.id}}" >任务：{{item.name}}</view>

<view class="box desc">简述：{{item.desc}} ...</view>

<view class="box coin">

<view class="coinleft">金币：{{item.coin}}</view><image src="/images/coin.png" class="pic"></image>

</view>

<view class="box wokers">所需人数：{{item.worker}} 人</view>

</view>

<view class="box address" id="{{item.id}}" bindtap = "viewonmap">

<view class="addressleft">地点：{{item.address}}...</view><image src="/images/location.png" class="locationpic"></image>

</view>

<view class="box tel" bindtap="telphone" id="{{item.id}}">电话咨询</view>

</view>

</scroll-view>

.WXSS示例代码：

.middle{

width:100%;

height:62%;

}

.footer{

width:100%;

height:12%;

background: rgb(255,255,255);

border-top:1px solid rgb(235,235,235);

}

.button{

width:21%;

height:100%;

text-align:center

}

JAVASCRIPT示例代码：

//从新加载任务页面

reload: function () {

// wx.redirectTo({

// url: '../index/index'

// })

},

toThing:function(){

wx.navigateTo({

url: '../things/things'

})

},

toDesc:function(){

wx.navigateTo({

url: '../about/about'

})

}

6.3 服务器端的实现

6.3.1 Express框架概述

Express是一个高效而强大的基于Node.js平台[13]，它提供一系列强大的特性，能够帮助开发者创建各种 Web 和移动设备应用，这也是我选着Express作为该项目微信小程序后台开发的原因之一。其次，Express是Node.js的 一个简洁、灵活的MVC幵发框架，它提供一系列强大的特性，具有丰富的HTTP 快捷方法和任意排列组合的Connect中间件[14]。让开发者创建健壮、友好的 API 变得既快速又简单[15]。我们知道NodeJS是一个基于Chrome V8引擎的JavaScript运行环境，使用事件驱动，非阻塞式I/O模型，使得其轻量又高效，Express不对Nodejs已有的特性做二次抽象，只是在NodeJS的基础上扩展了Web应用所需的基础功能，所以，Express框架在其性能上是完全继承NodeJS的，其对于高并发的处理也是很优秀的。

6.3.2 具体实现

在自己的开发电脑上安装Express的开发环境后，能够通过终端利用命令直接生成一个Express的MVC项目，生成项目的命令为（cd 进入项目存放目录后）：express –e projectName 即可自动生成一个Express项目，项目文件结构如下图：

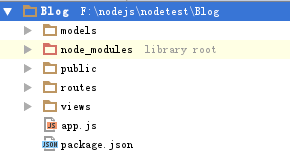


图 6 Express项目文件结构图

由上图可见Views、Routes、Models三个文件夹就是经典的MVC项目配置，整个项目的入口为app.js文件，由客户端发起的请求通过app.js进入Routes文件夹中的Controll文件，Controll文件根据不同的请求路径及不同的请求参数，寻找到不同的Model处理函数，处理函数做出处理后返回到Views文件夹中，重新渲染页面响应给客户端。

部分示例代码：

App.js代码：

var express = require('express');

var path = require('path');

var favicon = require('serve-favicon');

var logger = require('morgan');

var cookieParser = require('cookie-parser');

var bodyParser = require('body-parser');

var session = require('express-session');

var MongoStore = require('connect-mongo')(session);

var flash = require('connect-flash');

var routes = require('./routes/index');

var settings = require('./settings');

var users = require('./routes/users');

var app = express();

// view engine setup

app.set('views', path.join(\_\_dirname, 'views'));

app.set('view engine', 'ejs');

app.use(flash());//这时就可以使用flash的功能了

app.use(session({

secret:settings.cookieSecret,

key:settings.db,

cookie:{maxAge:1000\*60\*60\*24\*30},

store:new MongoStore({

db:settings.db,

host:settings.host,

port:settings.port

})

}));

// uncomment after placing your favicon in /public

//app.use(favicon(path.join(\_\_dirname, 'public', 'favicon.ico')));

……

module.exports = app;

Routes（路由控制）代码：由视图层（View）接受用户的数据请求操作后首先到达路由层（Routes）进行处理

//图片管理:上传图片

app.post('/addPictures', upload.array("file"), function(req, res, next){

var picTitle;

var picDesc;

var picAuthor;

var time;

var imageIndex;

var isdelete = "no";

if(req.body != undefined){

picAuthor = req.body.author;

picDesc = req.body.desc;

picTitle = req.body.title;

time = req.body.time;

}else{

res.send("上传参数错误,请重新上传");

return;

}

if(req.files==undefined || req.files.length == 0){

res.send("请选择要上传的图片...");

return;

}else{

var str="文件上传成功...";

for(var i=0;i<req.files.length;i++){

var filepath = \_\_dirname + "/../public/images/photos/" + req.files[i].originalname;

imageIndex = "/images/photos/" + req.files[i].originalname;

fs.renameSync(req.files[i].path,filepath);

}

//将数据存入数据库

var sql = "insert into pictures (title,imgdesc,imgurl,author,time,isdelete) values('"+picTitle+"','"+picDesc+"','"+imageIndex+"','"+picAuthor+"','"+time+"','"+isdelete+"');";

connection.query(sql,function(err,resInsert){

if(err){

console.log(err);

res.send('图片添加失败,请重试');

return;

}else{

res.send('图片添加成功');

return;

}

});

}

});

Views示例代码：视图层（View）接受用户的相关操作并更新视图

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8" />

<title>Blog</title>

<link rel="stylesheet" href="/stylesheets/styles.css">

</head>

<body>

<header>

<h1 class="font"><%= title %></h1>

</header>

<article>

<% if (success) { %>

<div><%= success %></div>

<% } %>

<% if (error) { %>

<div><%= error %> </div>

<% } %>

</article>

</body>

</html>

Model示例代码：由路由层（Route）接受视图层（View）传输的请求后，进行路由定向，到相应的数据处理层（Model）进行数据处理并返回

/用户注册(面向对象的方式) 文档,集合

var mongodb = require('./db');//导入已建立好的mongodb的连接

function User(user) {

this.name = user.name;

this.password = user.password;

this.email = user.email;

};

module.exports = User;

//存储用户信息

User.prototype.save = function(callback) {

//要存入数据库的用户文档

var user = {

name: this.name,

password: this.password,

email: this.email

};

//打开数据库

mongodb.open(function (err, db) {

if (err) {

return callback(err);//错误，返回 err 信息

}

//读取 users 集合

db.collection('users', function (err, collection) {

if (err) {

mongodb.close();

return callback(err);//错误，返回 err 信息

}

//将用户数据插入 users 集合

collection.insert(user, {

safe: true

}, function (err, user) {

mongodb.close();

if (err) {

return callback(err);//错误，返回 err 信息

}

callback(null, user[0]);//成功！err 为 null，并返回存储后的用户文档

});

});

});

};

7 程序测试

7.1 界面测试

7.1.1 微信小程序界面兼容测试

以apple手机iPhone5s（4 in）、iPhone 8 Plus（5.5 in）和Android手机华为Mate8（6 in）三款手机为测试机，分别运行开发完成的小程序应用，结果良好，兼容性良好，测试截图如下：

图 6 华为MATE8测试图 图 7 iPhone8 plus测试图

7.1.2 管理员Web界面兼容性测试

测试环境为以Webkit为内核原型的Chrome浏览器，Safari浏览器，和以Trident为内核的IE浏览器（向下兼容到IE9）。测试结果良好，兼容性良好。测试截图如下：

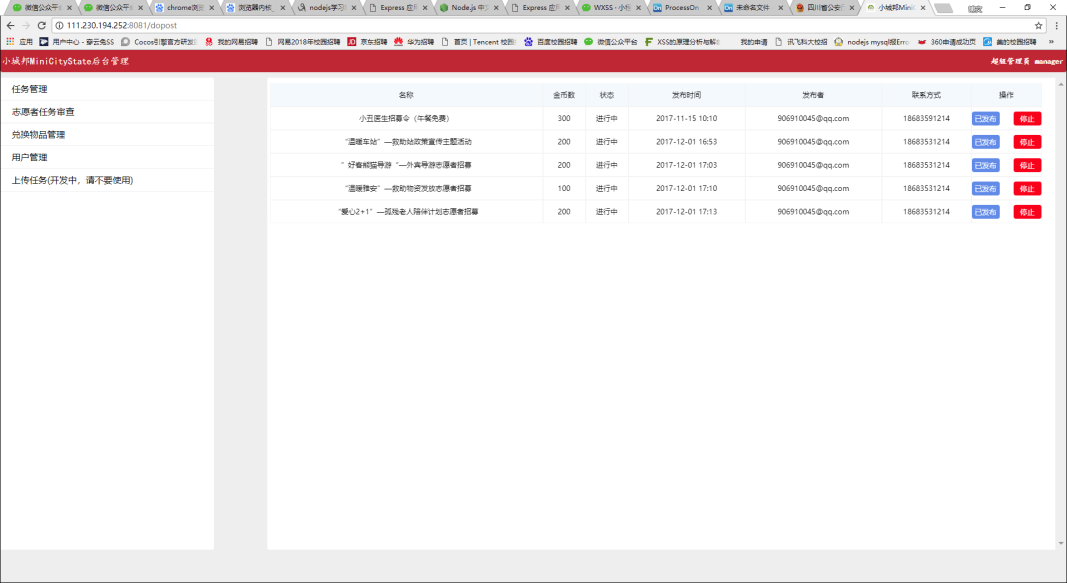


图 8 Chrome浏览器测试截图

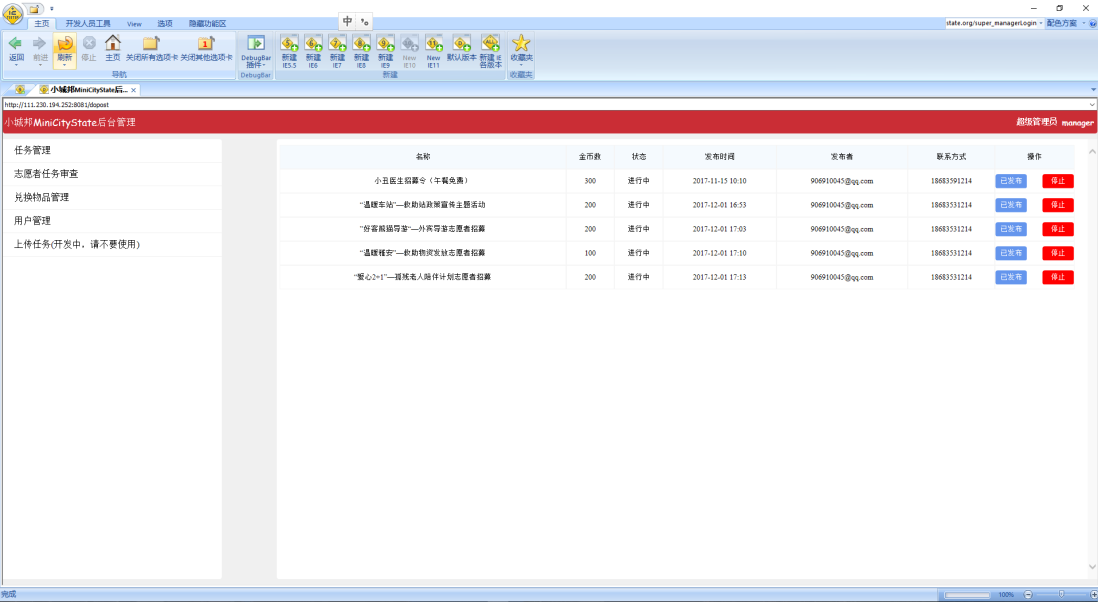


图9 IE9浏览器测试截图

7.2 功能模块测试（以用户注册为例）

7.2.1 程序核心部分代码

小程序注册页面代码：

<view class="container">

<image src="/images/logo.jpg" class="pic"></image>

<view class="welcome">欢迎注册</view>

<view class="scrollView" style="height:{{scrollViewHeight}}">

<scroll-view scroll-y="true" class="scrollView" style="height:{{scrollViewHeight}}">

<input placeholder="{{placeName}}" class="input name" value="{{name}}" bindblur="bindName"/>

<input placeholder="{{placeEmail}}" class="input tel" value="{{email}}" bindblur="bindEmail"/>

<view class="getCodeCon"><view class="getCode" bindtap="getCode">{{getCodeText}}</view></view>

<input placeholder="请输入验证码" class="input checkCode" bindblur="bindCode"/>

<input placeholder="请输入密码" password="true" class="input pwd" bindblur="bindPwd"/>

<view class="isUnit">

<view style="font-size:13px;margin-bottom:5px">是否是集体用户?</view>

<radio-group class="radio-group" bindchange="radioChange" style="color:red">

<radio value="yes" checked=""/>Y

<radio value="no" checked=""/>N

</radio-group>

</view>

<view class="reg" hover-class="regHover" bindtap="register">注册</view>

</scroll-view>

</view>

</view>

服务器接受注册请求代码：

if(req.url == "/register"){

req.on("data",function(data){

console.log(JSON.parse(data.toString()));

var result = JSON.parse(data.toString());

insertUserInfo(result.name,result.email,result.pwd,result.isUnit,res);

});

}

//TODO:发送邮箱验证码

var transporter = nodeMailer.createTransport('SMTP',{

host: "smtp.163.com",

secureConnection: true,

port:465,

auth: {

//你自己的可用的邮箱账号和密码

user: '18089598383@163.com',

pass: '……',

}

});

//TODO:发送邮件,@内容,@目标邮箱

var sendmail = function(html,toEmail,res,randomNum,object){

var option = {

from:"18089598383@163.com",

to:toEmail

}

option.subject = object;

option.html= html;

transporter.sendMail(option, function(error, response){

if(error){

console.log("fail: " + error);

}else{

console.log("success: " + response.message);

res.end("send success&"+randomNum);

}

});

}

//TODO:生成随机码

var createRandomNum = function(){

return Math.floor(Math.random()\*10);

}

7.2.2 邮件发送成功界面



图10 用户注册邮件发送截图

7.2.3 验证码接收界面



图 11 验证码接收成功截图

7.2.4 注册成功



图 12 用户注册成功截图

8 系统存在问题及其原因以及解决办法

8.1 小程序网络请求API次数限制

微信小程序中支持数据请求（wx.request）、文件上传下载（wx.uploadFile wx.downloadFile）等数据请求API，但微信小程序的开发框架中限制了网络请求的最大并发数为10个，所以，在开发过程中，存在着部分网络请求报错不能执行的问题，解决方案是将不同的网络请求API放置于小程序的不同生命周期函数中即可解决。

8.2 网络请求函数过于集中

小程序的每一个页面都具有自己的生命周期函数，这也为开发者带了很多可操作的方式。最常用的就是Page的onLoad函数，其的定义为“监听页面加载的函数”，故其会在页面加载时候执行。在一个页面加载的时候最好的工作就是进行网络请求，当页面加载完成的时候即可显示。但由于页面中过多网络请求集中于onLoad函数中，显得其十分臃肿，不易维护和阅读。解决方案是将不必要在页面加载时就需要进行的网络请求放置在onShow、onReady等生命周期函数中。

8.3 NodeJS多重回调嵌套问题

众所周知，NodeJS是基于天生异步的JavaScript语言的，故NodeJS也是极大继承其异步回调的属性，在开发过程中，当项目逐渐完善时候，容易出现多重回调函数嵌套的问题，导致阅读复杂，逻辑不清晰。NodeJS提供了async模块来解决这样的问题，将于项目重构时加入async模块，解决多重回调嵌套问题。

9 结论

本项目经过两个月的间断开发，成功利用微信小程序，Express框架实现了需求分析中所列功能，并成功上线使用。在开发中存在许多问题，但也逐一解决了。经过部分使用用户的反馈，项目中微信小程序端用户界面以及用户体验有待进一步优化和提升。后台开发在代码逻辑上也有待进一步的修改和完善，开发者也在积极筹备重构中。

参考文献

[1] 刘红卫. 微信小程序应用探析[J]. 无线互联科技, 2016(23):11-12.

[2] 龙军, 杨鹏. 基于Android的网络资源应用探析[J]. 电脑知识与技术, 2016, 12(5):191-193.

[3] 李凯. 基于Cordova的电影观赏系统的设计与实现[D]. 华中科技大学, 2016.

[4] Erasmus M. CoffeeScript Programming with JQuery, Rails, and Node. js[M]. Packt Publishing Ltd, 2012:111-123.

[5] 朴灵．深入浅出Node．JS[M]．北京：人民邮电出版社，2013：5—89．

[6] 刘杰夫. 基于MVC框架下的农业信息网络的开发[D]. 广西大学, 2017.

[7] 阿里云． MVC MVVM和MVP三种设计模式之个人理解[EB／OL]． https://www.aliyun.com/jiaocheng/374212.html, 2018-02-08.

[8] 闫林. 基于MVVM模式企业办公自动化管理系统研究与实现[D]. 河北工业大学, 2015.

[9] 陈涛. MVVM设计模式及其应用研究[J]. 计算机与数字工程,2014,10:1982-1985.

[10] 朱军. 抗阻组合训练方法手段多媒体网络课件的设计与制作[D]. 首都体育学院, 2015.

[11] JetBrains.WebStorm home page.[EB/OL]. http://www.jetbrains.com/webstorm/ ,2018-02-08

[12] 王静. Web技术在电子文档安全监管系统中的研究与应用[D]. 北京邮电大学, 2015.

[13] 刘超凡. 面向SaaS平台的快速部署系统的设计与实现[D]. 北京邮电大学, 2015.

[14] 邓茜. 移动智能终端网络化应用承载技术的研究与设计[D]. 北京邮电大学, 2014.

[15] 邱伟伟. 基于Node.js的万向节制造物联网系统[D]. 浙江工业大学, 2016.

致谢

在本篇论文完成的同时，我首先要感谢我的导师黄强老师。在黄强老师的悉心指导下，我的论文才能顺利的完成。在本篇论文的完成过程中，黄强老师对我提出了很多建设性的指导意见，细心地和我讲解论文中出现的问题，给予专业上的指导，信心上的鼓励。我能顺利完成本篇论文和老师的认真指导和辛苦的批改是分不开的，在此我由衷的向黄强老师表示感谢。

我也要感谢我的室友，他们在我写作的这段时间提供了良好的写作环境和各种有利条件，不断鼓励和帮助我，让我专注于写作一事而不用关注琐事。

最后，还要感谢参加评阅论文和答辩评审的各位专家老师，感谢你们在忙碌的日程中抽空参与我们的毕业事宜。