**­­­湖 南 科 技 大 学**

**毕 业 设 计（ 论 文 ）**

|  |  |
| --- | --- |
| **题目** | **基于WEB平台的阅读APP设计与实现** |
| **作者** | **曹慧琳** |
| **学院** | **计算机科学与工程** |
| **专业** | **物联网工程** |
| **学号** | **1305040226** |
| **指导教师** | **符琦** |

二〇一七年 四月 二十六日

**­**

**­湖 南 科 技 大 学**

**毕业设计（论文）任务书**

计算机科学与工程 院 物联网工程 系（教研室）

系（教研室）主任:（签名） 年 月 日

**学生姓名: 曹慧琳 学号: 1305040226 专业: 物联网工程**

1 设计（论文）题目及专题： 基于WEB平台的阅读APP的设计与实现

2 学生设计（论文）时间：自 2016 年 12 月 30 日开始至 2017 年 5 月 20 日止

3 设计（论文）所用资源和参考资料：

《Learning TypeScript》

《ES6标准入门(第2版)》

《React全栈：Redux+Flux+webpack+Babel整合开发》

《Go语言编程(图灵原创 6)》

4 设计（论文）应完成的主要内容：

1. 学习基于React+Redux的单页应用的搭建
2. 学习基于Go语言的RESTful API的实现原理

3) 设计并完成阅读App应用

4) 完成App的部署和测试

5 提交设计（论文）形式（设计说明与图纸或论文等）及要求：

1. 论文思路清晰，条理清楚，目的明确，内容充实，格式规范，层次分明，图表清晰，结论正确；按《湖南科技大学本科生毕业设计（论文）要求与规范》编排、打印；论文纸质文稿。

2. 上交论文（纸质）打印文稿和系统源码（电子版）。

6 发题时间： 2016 年 12 月 30 日

指导教师： （签名）

学 生： （签名）

**湖 南 科 技 大 学**

**毕业设计（论文）指导人评语**

**指导人：** （签名）

年 月 日

**指导人评定成绩：**

**湖 南 科 技 大 学**

**毕业设计（论文）评阅人评语**

**评阅人： （签名）**

**年 月 日**

**评阅人评定成绩：**

**湖 南 科 技 大 学**

**毕业设计（论文）答辩记录**

**日期：**

**学生： 学号： 班级：**

**题目：**

**提交毕业设计（论文）答辩委员会下列材料：**

**1 设计（论文）说明书 共 页**

**2 设计（论文）图 纸 共 页**

**3 指导人、评阅人评语 共 页**

**毕业设计（论文）答辩委员会评语：**

­

**答辩委员会主任：** （签名）

**委员：** （签名）

（签名）

（签名）

（签名）

**答辩成绩：**

**总评成绩：**

**摘 要**

在经济高速发展的今天，英语已经成为了我们了解全球文化，获取多方面知识以及和国外进行交流的必不可少的工具。英语的学习也成为了衡量一个学生获取信息的标准之一，国家对于英语教育的重视也越来越强。而对于英语的学习，单词的记忆毋庸置疑是基础。对于英语单词的记忆方式也是在不断的更新进步。随着移动互联网时代的到来，特别是智能手机的广泛应用，英语单词记忆的平台逐渐由单词本、计算机过渡到了手机、平板等移动设备上来。背单词的软件也随之成为了当下比较流行的、方便的、有效的辅助记忆的工具。本文介绍了一款基于WEB平台的背单词应用，它具备基本的单词背诵，真人发音等基本功能，实现了用户个性化选择所要背诵的单词，个性化的分组和选择背诵方式。这个应用将成为一种帮助英语作为外语的大学生的学习工具。

关键词：图书阅读；兴趣推荐；WEB；移动应用

**ABSTRACT**

In today's rapid economic development, English has become an indispensable tool for us to understand the global culture, to acquire various knowledge and to communicate with foreign countries. English learning has become one of the standards to measure a student's access to information, and national emphasis on English education has become more and more stronger. As for learning English, it is on doubt that the memory of words is the foundation.

The way of memorizing English words is also constantly updated progress. With the arrival of the era of mobile Internet, especially the wide application of smart phones, the English word memory platform gradual from the word, the computer transition to mobile phones, tablet and other mobile devices. Back word software has become the current popular, convenient and effective auxiliary memory tools. This article describes a Web-based word back platform application, it has the basic word recitation, live sound and other basic functions, the realization of the user personalization options to recite the words, personalized grouping and selection recitation. This application will become a learning tools to help college students in English as a foreign language learning tool.

**Keywords:** English learning；words back；Web Mobile Application

**目 录**

[第一章 前言 1](#_Toc452128801)

[1.1 英语词汇的学习 1](#_Toc452128802)

[1.2 大学的英语教育现状 1](#_Toc452128803)

[1.3 移动设备学习：理论与实践 1](#_Toc452128804)

[1.4 语言学习与智能手机 2](#_Toc452128805)

[1.5 当前单词记忆软件的发展现状 2](#_Toc452128806)

[1.6 本项目的特点 2](#_Toc452128807)

[第二章 开发工具及技术简介 3](#_Toc452128808)

[2.1 WebStorm 3](#_Toc452128809)

[2.2 ECMAScript 6 3](#_Toc452128810)

[2.3 React 3](#_Toc452128811)

[2.4 WebPack 3](#_Toc452128812)

[2.5 Node.js 3](#_Toc452128813)

[2.6 Koa 4](#_Toc452128814)

[2.7 HTML5 4](#_Toc452128815)

[2.8 Redux 4](#_Toc452128816)

[2.9 MongoDB 4](#_Toc452128817)

[2.10 Babel 5](#_Toc452128818)

[第三章 需求分析 6](#_Toc452128819)

[3.1 应用程序功能需求 6](#_Toc452128820)

[3.1.1 登录功能 6](#_Toc452128821)

[3.1.2 配置功能 6](#_Toc452128822)

[3.1.3 背诵列表功能 6](#_Toc452128823)

[3.1.4 背诵功能 7](#_Toc452128824)

[3.1.5 主界面功能 7](#_Toc452128825)

[3.2 服务端功能需求 7](#_Toc452128826)

[3.2.1 用户登录服务 7](#_Toc452128827)

[3.2.2 单词词库查询服务 8](#_Toc452128828)

[3.2.3 背诵列表查询服务 8](#_Toc452128829)

[3.2.4 背诵进度上传服务 8](#_Toc452128830)

[3.2.5 真人发音服务 8](#_Toc452128831)

[3.3 性能需求 8](#_Toc452128832)

[3.4 运行需求 8](#_Toc452128833)

[第四章 数据库设计 9](#_Toc452128834)

[4.1数据库的需求分析 9](#_Toc452128835)

[4.2 数据库的概念设计 9](#_Toc452128836)

[图4.1数据库实体关系图 10](#_Toc452128837)

[4.3 数据库的逻辑设计 10](#_Toc452128838)

[表4.2 User的逻辑设计表 10](#_Toc452128839)

[表4.3 WordBook的逻辑设计表 10](#_Toc452128840)

[表4.4 Word的逻辑设计表 11](#_Toc452128841)

[表4.5 WordGroup的逻辑设计表 11](#_Toc452128842)

[表4.6 WordGroupItem的逻辑设计表 12](#_Toc452128843)

[表4.7 History的逻辑设计表 12](#_Toc452128844)

[第五章 详细设计 13](#_Toc452128845)

[5.1 爬虫模块设计 13](#_Toc452128846)

[图5.1 爬虫模块数据处理流程图 13](#_Toc452128847)

[5.1.1词库数据采集模块 13](#_Toc452128848)

[5.1.2 单词信息采集模块 14](#_Toc452128849)

[5.1.3 数据清洗模块 14](#_Toc452128850)

[5.1.4 数据验证模块 14](#_Toc452128851)

[5.2 后端API接口设计 14](#_Toc452128852)

[5.2.1 获取用户信息的API 14](#_Toc452128853)

[5.2.2 修改用户信息的API 15](#_Toc452128854)

[5.2.3 获取单词书列表API 15](#_Toc452128855)

[5.2.4 获取某本单词书单词列表的API 16](#_Toc452128856)

[5.2.5 获取用户单词分组列表的API 16](#_Toc452128857)

[5.2.6 获取某个单词分组的API 17](#_Toc452128858)

[5.2.7 上传背诵记录的API 17](#_Toc452128859)

[5.2.8 生成分组信息的API 18](#_Toc452128860)

[5.2.9 删除分组信息的API 18](#_Toc452128861)

[5.3 前端交互设计 19](#_Toc452128862)

[5.3.1 路由设计 19](#_Toc452128863)

[5.3.2主界面布局设计 19](#_Toc452128864)

[5.3.3 主界面交互设计 20](#_Toc452128865)

[5.3.4 单词分组列表界面布局设计 20](#_Toc452128866)

[5.3.5 单词分组列表界面的交互设计 21](#_Toc452128867)

[5.3.6卡片背单词界面的布局设计 21](#_Toc452128868)

[5.3.7 卡片背单词界面的交互设计 22](#_Toc452128869)

[5.3.8 列表背单词界面布局设计 22](#_Toc452128870)

[5.3.9列表背单词页面交互设计 23](#_Toc452128871)

[第六章 系统实现与测试 24](#_Toc452128872)

[6.1爬虫模块实现 24](#_Toc452128873)

[6.1.1词库数据采集模块实现 24](#_Toc452128874)

[6.1.2 单词信息采集模块实现 24](#_Toc452128875)

[6.1.3 数据清洗模块实现 25](#_Toc452128876)

[6.1.4 数据验证模块实现 26](#_Toc452128877)

[6.2 后端API接口实现 27](#_Toc452128878)

[6.2.1 获取用户信息的API实现 27](#_Toc452128879)

[6.2.2 修改用户信息的API实现 28](#_Toc452128880)

[6.2.3 获取单词书列表API实现 29](#_Toc452128881)

[6.2.4 获取某本单词书单词列表的API实现 29](#_Toc452128882)

[6.2.5 获取用户单词分组列表的API实现 30](#_Toc452128883)

[6.2.6 获取某个单词分组的API实现 31](#_Toc452128884)

[6.2.7 上传背诵记录的API实现 32](#_Toc452128885)

[6.2.8 生成分组信息的API实现 33](#_Toc452128886)

[6.2.9 删除分组信息的API实现 34](#_Toc452128887)

[6.3 前端逻辑实现 35](#_Toc452128888)

[6.3.1前端ACTION定义 35](#_Toc452128889)

[6.3.2 前端ACTION提供的方法 36](#_Toc452128890)

[6.3.3 前端Middleware实现 37](#_Toc452128891)

[6.3.4 前端Reducer和Store实现 37](#_Toc452128892)

[6.4 前端组件实现 38](#_Toc452128893)

[6.4.1 Root组件实现 38](#_Toc452128894)

[6.4.2 Router组件实现 38](#_Toc452128895)

[6.4.3 GlobalToast组件实现 39](#_Toc452128896)

[6.4.4 IndexPage组件实现 40](#_Toc452128897)

[6.4.5 WordGroup组件实现 42](#_Toc452128898)

[6.4.6 WordGroupLearn组件实现 43](#_Toc452128899)

[6.4.7 WordGroupLearnCard组件实现 43](#_Toc452128900)

[6.4.8 WordGroupLearnList组件实现 45](#_Toc452128901)

[6.5 系统测试 47](#_Toc452128902)

[6.5.1 后端API功能测试 47](#_Toc452128903)

[6.5.2 前端功能测试 50](#_Toc452128904)

[第七章 总结 54](#_Toc452128905)

[参考文献 57](#_Toc452128906)

[致谢 58](#_Toc452128907)

1. 前言

1.1 阅读——中华名族的优良传统

中华名族是个有着优良的读书传统和悠久的读书历史的民族。上下五千年的历史长河中，文化典籍可以说是浩如烟海。古有悬梁刺股，凿壁借光，囊萤映雪之典故，近代有周总理“为中华之崛起而读书”的豪情，如今更有许许多多在读书人用成功诠释着读书的魅力。书籍是中华传统文化的载体，承载着中国人自古以来的精神追求。

## 1.2 阅读的重要性

读万卷书，行万里路’，读书不仅是一个人修养的一种标志，也是自我完善，自我提升，凝聚智慧的重要途径。勤于读书，善于读书，不仅能够增加你的知识储备，职业技能，还能提高你的精神气质，言谈举止之间透露出那种文雅脱俗，不卑不亢的境界。

## 1.3 当前阅读现状

目前，中国的经济实力已跃居世界前列，国家硬实力不断加强。但与之鲜明对比的是，中华文化的软实力确明显滞后。调查显示，我们国民阅读的整体已处于偏低水平，更多的出现功利读书的现象。读书，成了杀出重围获取高等学府的“入场券”，成了养家糊口“稻粮谋”的证书职称。现代人背负着生活重压，忙于考试，忙于工作，忙于挣钱，却忘了读书真正的意义。在全面实现小康社会的新时代，继承传统的读书文化，发扬中华名族的名族精神将会是我们面临的新的挑战。

## 1.4 阅读与移动设备

在发扬中国传统的读书文化，寻找新的读书方式的过程中，很多人把目光投向了智能手机。快节奏生活、碎片化阅读以及智能手机的普及，使用手机阅读渐渐成为主流，手机报、手机杂志、手机电子书等移动阅读方式也在慢慢融入人们的生活。这种方便，快捷，海量的读书方式也逐渐得到大家的认可。当然如今的手机阅读还存在着大量的问题，阅读的体验，长时间对身体的消耗，经典阅读微乎其微。数字化阅读方式使得泛读化、快餐化的浅阅读倾向呈现出“短、平、快”的阅读趋势，以丰富情趣、提高修养为目的的素质性阅读逐渐弱化，导致人们的阅读动机以休闲娱乐为主导，而文化传承、思想积淀呼唤的精读和深度阅读被大多数人摒弃。

## 1.5 本项目的特点

在阅读数据采集上，该项目以传统文化为核心，更多的去承载经典的文学作品；在设计上，采用简洁的设计风格，给读者一种清新的阅读心境；在阅读方式上，项目给予用户更真实的读书体验，智能适应用户的读书习惯；在技术上，采用全新的Web技术，实现了多平台同步使用，同步更新；在资源利用上，实现了按需加载，更多的从用户的角度考虑，节省内存，节省流量。这种全新的阅读体验将会为全民阅读的新时代贡献出不可或缺的力量。

1. 开发工具及技术简介

## 2.1 Visual Studio Code

Visual Studio Code是微软在Build 2015大会上发布的一款免费的现代化轻量级代码编辑器，支持了几乎所有主流开发语言的语法高亮、智能代码补全、自定义热键、括号匹配、代码片段、代码对比 Diff、GIT 等特性并支持插件扩展，支持Linux, OS X, and Windows等多个平台，可以说是程序员的开发利器。

## 2.2 TypeScript

TypeScript是由微软开发的开源的编程语言，它是JavaScript的超集， 并添加了可选的静态类型和基于类的面向对象编程。TypeScript完美的兼容了ES6，ES7的新特性，可编译成可读的，标准的JavaScript。TypeScript支持任意浏览器，任意环境，任意系统并且是开源的。

本项目前端逻辑全部由TypeScript编写而成。

## 2.3 ECMAScript 6

ECMAScript 6 是JavaScript这门语言的下一代标准，于2015年6月发布。

ES6在JavaScript标准的基础上添加了许多新的语言特性，是的JavaScript变得更加强大，更富表现力，也解决了JavaScript的很多局限性，使JavaScript走上了企业级的开发道路。

本项目前端逻辑TypeScript采用ECMAScript 6的语法实现。

## 2.4 React

React，颠覆前端UI开发的框架。React是由Facebook开发的用于开发数据不断变化的大型应用程序的UI框架。React采用了Virtual DOM的新思想，从根本上解决了频繁DOM操作的性能瓶颈问题。React采用组件化的构建思想，组件生命周期管理的方式，将每个功能相对独立的模块封装为组件，项目内部组件可重复使用，提高了代码重用率和开发效率。

本项目前端框架基于React框架构建。

## 2.5 Redux

Redux是FacebookFlux架构的一种简化实现，它是前端数据的一个状态管理器。Redux采用了reduce和纯函数的概念，每个新的state都是由旧的state在action的作用下生成的。state集中在单一树状对象上，即store，是整个项目的数据层。Redux通过这种数据管理方式解决了React数据处理比较混乱的问题。

本项目采用了Redux-Immutable的形式实现前端的数据管理，Immutable 内部使用了 Trie 数据结构来存储， 两个对象的区别取决去hashCode的值，这给reactxuniDOM是否刷新的判断带来了质的飞跃。

## 2.6 webpack+gulp

Webpack 是一个模块打包器。它将根据模块的依赖关系进行静态分析，然后将这些模块按照指定的规则生成对应的静态资源。 gulp是前端开发过程中对代码进行构建的工具，是自动化项目的构建利器；它不仅能对网站资源进行优化，而且在开发过程中很多重复的任务，比如文件的的测试、检查、合并、压缩、格式化、浏览器自动刷新、部署文件生成，并监听文件在改动后重复指定的这些步骤能够使用正确的工具自动完成。webpack和gulp的结合，完美的发挥了各自的优点，将前端项目的打包，开发体验提升到最好。

## 2.7 Go

Go语言是由Google开发的一个开源项目，目的之一为了提高开发人员的编程效率。 Go语言语法灵活、简洁、清晰、高效。它对的并发特性可以方便地用于多核处理器 和网络开发，同时灵活新颖的类型系统可以方便地编写模块化的系统。go可以快速编译， 同时具有垃圾内存自动回收功能，并且还支持运行时反射。Go是一个高效、静态类型， 但是又具有解释语言的动态类型特征的系统级语法。

本项目的后端服务搭建和API采用Go语言实现。

## 2.8数据库 MongoDB

MongoDB 是由C++语言编写的，是一个基于分布式文件存储的开源数据库系统。

MongoDB 旨在为WEB应用提供可扩展的高性能数据存储解决方案。

MongoDB 将数据存储为一个文档，数据结构由键值对组成。MongoDB 文档类似于 JSON 对象。字段值可以包含其他文档，数组及文档数组。

本项目因为源数据都采用JSON格式，采用MongoDB可以不需要经过多余的转换数据步骤就可以使用数据。

## 2.9 REST API

REST，REpresentational State Transfer，表示表述性无状态传输，无需session。REST是基于HTTP协议的，其核心的概念就是资源，API的实现是对资源的增删改查。REST 通常使用 JSON 数据格式。

## 本项目API请求全部采用REST API规范。

1. 需求分析

本章详细描述了本项目的详细需求，为整个项目开发和技术实现提供指南，同时确定项目的内容和范围，为评价和测试该项目提供依据。

该项目客户端功能模块大致分为：登录注册模块、图书搜索模块、图书分类模块、图书阅读模块、图书收藏模块、图书分享模块、图书推荐模块、阅读历史查看模块。

下面将分别对于这些功能模块进行需求分析的详细说明。

3.1.1 登录注册功能

为了方便用户的使用，数据同步等，软件需要登录注册，该模块设计的主要需求为：

（1）用户可以采用游客的方式试用软件，在没有登录时可以体验应用的部分功能。

（2）用户点击登录按钮，进入登录页面，用户可试用已经注册的账号密码登录，也可使用微信，微博，QQ等第三方应用账号登录。

（3）还没有账户的用户可点击注册按钮，进入注册页面进行注册，注册成功后，将自动登录。

（4）用户首次注册成功后，将进行阅读兴趣的选择，阅读方式的配置等，将用户的配置信息存储到服务器进行同步。

（5）用户可在user页面退出登录

3.1.2图书搜索功能

用户可输入关键词进行图书搜索，该模块设计的主要需求为：

（1）用户进入主页可在页面中看到搜索栏。

（2）在搜索栏中输入关键词即可搜索想看的图书。

（3）根据用户的输入实时显示搜索结果。

3.1.3图书分类功能

用户可筛选特定类型的书籍，该模块设计的主要需求为：

（1）用户进入主页可看到图书的所有分类。

（2）点击类型，显示对应类型的图书列表，列表中包含图书名称及介绍信息。

（3）点击列表中图书，进入图书的详细介绍页。

3.1.4图书阅读功能

用户可阅读选择的图书，该模块设计的主要需求为：

（1）用户点击图书详细介绍页的去读或者目录栏的特定章节，即可进入阅读模式。

（2）阅读模式主页显示对应章节的书的内容，并实现逼着的翻页动画。

（3）页面左下角显示目前的读书进度。

（4）页面的右下角显示当前时间和当前设备的电量，以及是否处于充电模式。

（5）单击页面中的任何位置，弹出设置页面。

（6）设置页面可设置屏幕亮度，字体大小，白天夜间模式等。

3.1.5图书收藏功能

用户可对自己喜欢的书籍进行收藏，该模块设计的主要需求为：

（1）单击阅读页面,弹出收藏按钮。

（2）如果用户喜欢该书籍，可点击收藏按钮进行收藏，收藏按钮变为已收藏按钮。

（3）如果用户已经收藏可点击已收藏按钮，取消收藏。

（4）用户可进入收藏页面，查看自己已经收藏图书的图书列表。

3.1.6图书分享功能

用户可对自己喜欢的书籍进行分享，该模块设计的主要需求为：

（1）单击阅读页面,弹出分享按钮。

（2）如果用户喜欢该书籍，可点击分享按钮，弹出选择分享方式的页面。

（3）分享方式有朋友圈、qq空间、新浪微博、微信好友、qq好友。

（4）单击分享方式即可分享对应书籍。

3.1.7图书推荐功能

服务器根据用户的兴趣和读书种类，给用户推荐其可能喜爱的图书，该模块设计的主要需求为：

（1）用户在进入收藏页面时，可在页面尾部看到系统推荐图书的图书列表。

（2）点击感兴趣的图书即可查看图书详细信息和阅读图书。

3.1.8阅读历史查看功能

用户可查看自己的阅读历史，该模块设计的主要需求为：

（1）用户再次进入应用，应用会提醒用户是否回到上次阅读，用户点击是可直接进入上次阅读的页面。

（2）用户查看某本已经看过的书籍时，应用会提醒用户是否回到该书的上次阅读，用户点击是可直接进入该书的上次阅读的页面。

（3）用户可在User页面查看读过书的数量及近期读书的时间。

3.2 服务端功能需求

服务器端主要给客户端提供RESTful API支持。下面针对具体服务进行需求上的详细说明。

3.2.1 用户登录注册服务

用户登录注册服务主要响应客户端传送过来的登录注册请求，将用户信息保存到数据库并返回成功登陆注册的提示信息，如果登录失败，则返回友好的错误信息。

3.2.2图书关键词查询服务

图书关键词查询服务主要响应客户端的查询请求，通过参数的关键字词信息，在数据库中查找相关信息，并将信息返回给客户端。

3.2.3获取所有分类服务

获取所有分类服务主要响应客户端获取所有分类的请求，将数据库中数据所有类型返回给客户端。

3.2.4获取当前分类图书列表服务

获取当前分类图书列表服务主要响应客户端特定类型图书列表的请求，将数据库中属于该类型的图书列表返回给客户端。

3.2.5图书详细信息查询服务

图书详细信息查询服务主要响应客户端获取特定图书详细信息的请求，将数据库中该书的介绍信息返回给客户端。

3.2.6用户阅读进度上传服务

用户阅读进度上传服务主要响应客户端存储用户进度的请求，将用户阅读进度及历史的相关信息存储到数据库中。

3.2.7用户阅读进度查询服务

用户阅读进度查询服务主要响应客户端获取用户阅读进度的请求，从数据库中读取用户最近阅读进度，返回给客户端。

3.2.8收藏图书服务

用户阅读进度上传服务主要响应客户端添加或删除收藏图书的请求，将收藏的图书记录到用户收藏列表或者将该图书从收藏列表中删除。

3.2.9用户收藏图书查询服务

用户收藏图书查询服务主要响应客户端获取用户收藏图书列表的请求，从数据库中获取用户收藏图书列表，返回给客户端。

3.2.10用户兴趣推荐服务

用户兴趣推荐服务主要根据用户阅读历史，收藏数据等信息进行用户行为分析，生成推荐列表，向客户端推送。

3.3 性能需求

（1）数据准确：图书信息，用户信息，用户收藏等API必须保证准备返回，保证良好的用户体验。

（2）响应时间：图书信息，用户信息，用户收藏等API保证在500ms之内返回。

（3）响应信息准确详细：用户请求API时尽可能返回详细的错误信息。

3.4 运行需求

（1）客户端支持主流的Android手机，支持Android4.0以上、iOS 8以上系统。

（2）在用户没有网络或者网络极差的条件下，客户端需保证基本功能可以流畅使用。

1. 数据库设计

4.1数据库的需求分析

通过对项目的需求分析，数据库数据存储的基本信息如下

（1）图书分类信息： 类型ID、类型名称、该类型下书的数目。

（2）图书详细信息：图书ID、图书名称、图书作者、图书介绍、图书封面、图书章节列表（章节ID、章节名称）、图书分类。

（3）图书内容：图书ID，章节ID，图书内容。

（4）用户信息：用户ID、用户昵称、账号、密码、阅读详情（今日读书时间、累计读书时间、累计读书数目）

（5）收藏图书信息：用户ID，收藏图书列表（图书ID，分类标签）。

（6）推荐图书信息：用户ID，推荐图书列表（图书ID）。

（7）历史记录信息：用户ID，图书ID，章节ID，页数，历史创建时间。

4.2 数据库的概念设计

根据数据库需求分析，数据库应该包含八个主要的数据实体：

Tag：tagId, category, booksCnt

Book: bookId, name, author, introduce, picture, catalogueId, tagId

Catalogue: categoryId, bookId, chapter

Content: bookId, categoryId, text

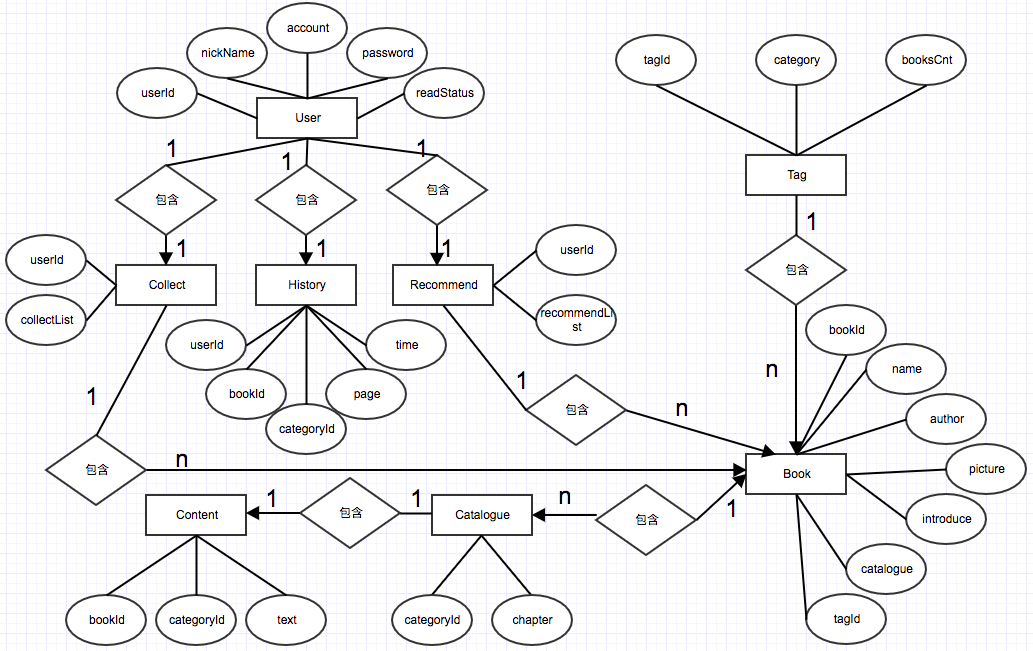
User: userId, nickName, account, password, readStatus

Collect: userId, collectList

Recommend: userId, recommendList

History: userId, bookId,categoryId,page, time

实体之间关系如图4.1.

图4.1数据库实体关系图

4.3 数据库的逻辑设计

表4.3.1 Tag的逻辑设计表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 |  |
| 1 | tagId | string | 主键 |
| 2 | category | string |  |
| 3 | booksCnt | number |  |

表4.3.1 描述图书类别信息的数据表，其中tagId为主键，category为类别名称，bookCnt为当前该类别所有图书的数量，每一次后端更新或者添加图书时，更新数据表。

表4.3.2 Book的逻辑设计表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 |  |
| 1 | bookId | string | 主键 |
| 2 | name | string |  |
| 3 | author | string |  |
| 4  5  6  7 | introduce  picture  catalogue  tagId | string  string  引用数组  string | 外键 |

表4.3.2 描述图书信息的数据表，其中bookId为主键，tagId为所属分类的外键，catalogue为该书所有章节的列表，书籍更新到数据库后信息基本不更新。

表4.3.3 Catalogue的逻辑设计表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 |  |
| 1 | catalogueId | string | 主键 |
| 2  3 | bookId  chapter | string  string | 外键 |

表4.3.3 描述章节具体信息的数据表。其中catalogueId为主键，chapter为章节标题。该数据为图书catalogueList的元素。

表4.3.4 Content的逻辑设计表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 |  |
| 1 | bookId | string | 外键 |
| 2 | categoryId | string | 外键 |
| 3 | text | 引用数组 |  |

表4.3.4 描述书籍详细内容的数据表。其中bookId，categoryId为主键。text为文章内容的数组，按段落分组。

表4.3.5 User的逻辑设计表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 |  |
| 1 | userId | string | 主键 |
| 2 | nickName | string |  |
| 3  4  5 | account  password  readStatus | string  string  引用数组 |  |

表4.3.5 描述用户详细信息的数据表。其中userId为主键，readStatus为用户历史阅读状态的统计，包括今日读书时间，累计读书时间和累计读书的数量，每日读书时间每晚24点清零。

表4.3.6 Collect的逻辑设计表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 |  |
| 1 | userId | string | 主键，外键 |
| 2 | collectList | 引用数组 |  |

表4.3.6用来描述历史信息的数据库，其中单词为在记录时对应的那个单词，创建的时间为Date类型，格式为1970-01-01，记录是否记住为Boolean类型，usertime代表用户在这个单词上停留的时间，为秒数。

表4.3.7 Recommend的逻辑设计表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 |  |
| 1 | userId | string | 主键，外键 |
| 2 | recommendList | 引用数组 |  |

表4.3.7 描述用户推荐列表的数据表，其中userId为主键，系统会定时进行用户行为分析，得出可能用户感兴趣的图书列表，将图书列表的id存储到recommendList中。

表4.3.8 History的逻辑设计表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 |  |
| 1 | userId | string | 主键，外键 |
| 2 | bookId | string | 外键 |
| 3 | categoryId | string | 外键 |
| 4 | page | string |  |

表4.3.8 描述用户阅读历史的数据表。其中userId为主键，用户阅读图书时会不断上传当前读书的状态，将这些状态记录到该表中，用于分析用户行为和提醒用户快速回到上次阅读的状态。

。

1. 详细设计

本章将在需求分析和数据库设计的基础上介绍该项目的详细设计过程。

5.1 爬虫模块设计

作为一个背单词软件，各种常见的单词词库和单词释义发音等信息肯定是必须提供的，而不能让用户手动去录入。

市面上主流的单词软件甚至有专门的单词数据编辑团队来做这些事，但是我们资源有限，只能选择以爬虫的方式把别人网站的数据爬下来作为已用。

首先我们需要四六级这样的考试词库的词的列表，这可以从其它的背单词网站上爬取得到。然后我们需要单词释义，例句，发音这些数据，这可以从一些在线词典网站上爬取得到。

爬虫模块数据处理流程图如下：



图5.1 爬虫模块数据处理流程图

5.1.1词库数据采集模块

词库数据的采集是项目进行和页面显示的基础，大致分为以下步骤

词库的选择。当前这方面的应用词库比较多，通过多方面筛选，扇贝网的词库比较全面，适合使用。

脚本获取词库。通过脚本，将所需词库获取到本地，保存为JSON文件。

5.1.2 单词信息采集模块

获取到对应词库后，需要采集每个单词对应的句子，释义，短语，真人发音等信息，并将采集数据存储到数据库中。

单词信息的选择。软件对于单词信息的需求比较多，通过比较有道，必应词典，百度词典，海词词典等多个在线词典，海词词典的信息比较全面规范，适合使用。

脚本获取单词信息。脚本从先前采集的词库数据中读出单词列表，并通过单词列表在海词词典网站上查询相关信息，并保存到数据库。

5.1.3 数据清洗模块

上一步获取到的信息是HTML源码，无法直接使用，需要将源码进行分析，取出其中有用的信息。具体步骤如下。

从数据库中取出上一步获取的HTML源码。

遍历DOM树，提取有用信息，保存为JSON格式，存入数据库。

5.1.4 数据验证模块

为了使应用可以更好的运行，用户也不会获取到一些比较奇怪的的数据，需要对数据进行验证，验证过程如下：

（1）遍历每一条记录，对获取的信息进行判断。

（2）当对应信息不存在或者不符合所需格式时，重新获取对应信息，如果还不存在，赋值为新的提示。

5.2 后端API接口设计

下面是后端API接口请求和返回数据结构的设计。以便在后端API未就绪的时候就可以按照API格式约定生成MOCK数据把前端跑起来。

5.2.1 获取用户信息的API

用户进入软件，首先获取对应用户的信息，这就需要获取用户信息的API。

请求包

GET /user

返回包

200 OK

**{**

name**:** **<**Name**>**

nickname**:** **<**NickName**>**

wordbook**:** **{**

name**:<**WordbookName**>**

desc**:** **<**WordbookDesc**>**

**}**

**}**

注意：如果用户没有选择过单词书，则wordbook项的值为null

5.2.2 修改用户信息的API

软件需要满足用户修改其昵称，重新选择单词书的功能，这就需要修改信息的API

请求包

PUT /user

{

wordbooknow: <NewWordBookName>

nickname: <NewNickName>

}

返回包

200 OK

{}

注意：参数是可选的，它会根据你传递的参数进行对应的修改，空值会被忽略掉。

5.2.3 获取单词书列表API

用户在选择单词书时，需要先获取到单词书的列表供用户选择，因此需要获取单词书列表的API

请求包

GET /wordbook

返回包

200 OK

[

{

name:<WordbookName>

desc: <WordbookDesc>

}

]

5.2.4 获取某本单词书单词列表的API

用户在背诵所选单词书时，需要获取该单词书中的单词列表，因此需要获取某本单词书单词列表的API

请求包

GET /wordbook/:name

返回包

200 OK

[

{

word:<word>

explain:<explain>

}

]

5.2.5 获取用户单词分组列表的API

对于每一本单词书，用户都会在选择时给自己设定一个每天背诵的单词数，后端会根据用户的设置进行智能分组，并返回分组的列表，因此需要获取分组列表的API

请求包

GET /wordgroup

返回包

200 OK

{

user: <UserID>

groups: [

<GroupID>,

<GroupID>,

]

config: {

everyday: <Everyday:Number>

disturb: <Disturb:Boolean>

}

}

5.2.6 获取某个单词分组的API

点击分组时，需要获取当前分组中的单词列表，所以需要获取某个单词分组的API

请求包

GET /wordgroup/:id

返回包

200 OK

{

history:[

userTime:<UserTime:Number>

createDate:<CreateDate:Date>

word:<WordID>

remember:<Rembmber:Boolean>

],

words:[

word:<Word>

explain:<Explain>

]

}

5.2.7 上传背诵记录的API

当用户点击背诵按钮时，需要将用户背会的信息传入数据库中，因此需要上传背诵记录的API

请求包

POST /wordgroup/:gid

{

remember:true

word:<Word>

createDate:<CreateDate:Date>

useTime:<UseTime:Number>

}

返回包

200 OK

{}

5.2.8 生成分组信息的API

用户第一次选择单词书进行个性化配置每天单词书和时候选择乱序或者用户选择重新分组的时候，需将分组信息上传到数据库，所以需要生成分组信息的API

请求包

POST /wordgroup

{

everyDay:<EveryDay>

disturb:<Disturb>

}

返回包

200 OK

{}

5.2.9 删除分组信息的API

用户选择重新分组时，首先先删除用户当前分组信息，然后再重新分组，避免因为失误点击造成重新分组而使用户原先数据被覆盖。

请求包

DELETE /wordgroup

返回包

200 OK

{}

5.3 前端交互设计

前端采用React框架，实现单页面组件化管理应用。

5.3.1 路由设计

**表5.2 项目路由设计表**

|  |  |  |
| --- | --- | --- |
| URL | 模块 | 描述 |
| / | Index | 进入应用的主界面 |
| /wordgroup | WordGroup | 进入应用的分组列表界面 |
| /wordgroup/:id | WordGroupOne | 进入应用的某一分组单词列表界面 |
| /config/wordbook | WordBookChoose | 进入应用的选择单词书并进行基本设置页面 |
| /login | Login | 用户登录系统 |

5.3.2主界面布局设计

当用户进入软件登录之后首先进入主界面模块。

主界面主要分为三部分

（1）用户交互趣图：通过获取用户昵称，实现一个吸引背单词的趣图，来增加软件的趣味感。

（2）当日单词数目提醒：获取到当天分组的背诵单词情况，以醒目的提醒用户当天应该完成的任务和具体的进度。

（3）每日一句：动态给用户提供每日一句，中英对照，在背单词之余每天翻译一个句子也是不错的选择。



**图 5.3 主界面布局原型设计**

5.3.3 主界面交互设计

主界面的交互主要体现在当日单词提醒这一部分，用户可以通过这里的数据了解自己当天的任务，完成的进度，点击开始按钮后，进入单词分组列表模块。

5.3.4 单词分组列表界面布局设计

单词分组列表界面分两个部分

（1）所选单词书的背景图和描述

（2）分组信息，显示所有分组及分组中的单词书和所背单词的进度。



**图 5.4 单词列表界面布局原型设计**

5.3.5 单词分组列表界面的交互设计

单词分组列表的交互设计主要体现在：

（1）获取用户当前正在背诵的单词书，显示其封面和描述，对用户起提示作用。

（2）显示所有分组，并显示分组的具体单词数目及当前用户的进度，有助于用户把握自己的整体进度。

（3）用户点击分组进入卡片背单词页面。

5.3.6卡片背单词界面的布局设计

卡片背单词页面布局中是一张卡片，中间显示所背诵的单词和英文释义。右上角添加卡片背单词和列表背单词的切换按钮。



**图 5.5 卡片背单词布局设计**

5.3.7 卡片背单词界面的交互设计

卡片背单词界面的交互设计主要体现在：

（1）卡片上显示用户当前分组上一次背诵单词的位置。

（2）当用户向左滑动界面时，显示下一个单词。

（3）当用户向右滑动界面时，显示上一个单词。

（4）当用户向上滑动界面时，代表该单词已经记住。

（5）当用户向下滑动界面时，标记当前单词为难词。

（6）当用户点击切换到列表背单词按钮时，进入列表背单词页面。

5.3.8 列表背单词界面布局设计

列表背单词界面是把所有单词以列表的形式呈现，让用户快速查看当前分组的单词的背诵情况。

****

**图 5.6 列表背单词布局原型设计**

5.3.9列表背单词页面交互设计

列表背单词页面的交互设计主要体现在：

（1）当用户向左滑动单词列表项时，代表用户标记该单词为难词，单词颜色变为红色。

（2）当用户向右滑动单词列表项时，代表用户已经记住该单词，单词颜色变为绿色。

（3）当点击右上角切换按钮到卡片背单词按钮时，进入卡片背单词页面。

1. 系统实现与测试

6.1爬虫模块实现

6.1.1词库数据采集模块实现

在词库数据采集方面我使用了Koa，采集的步骤如下：

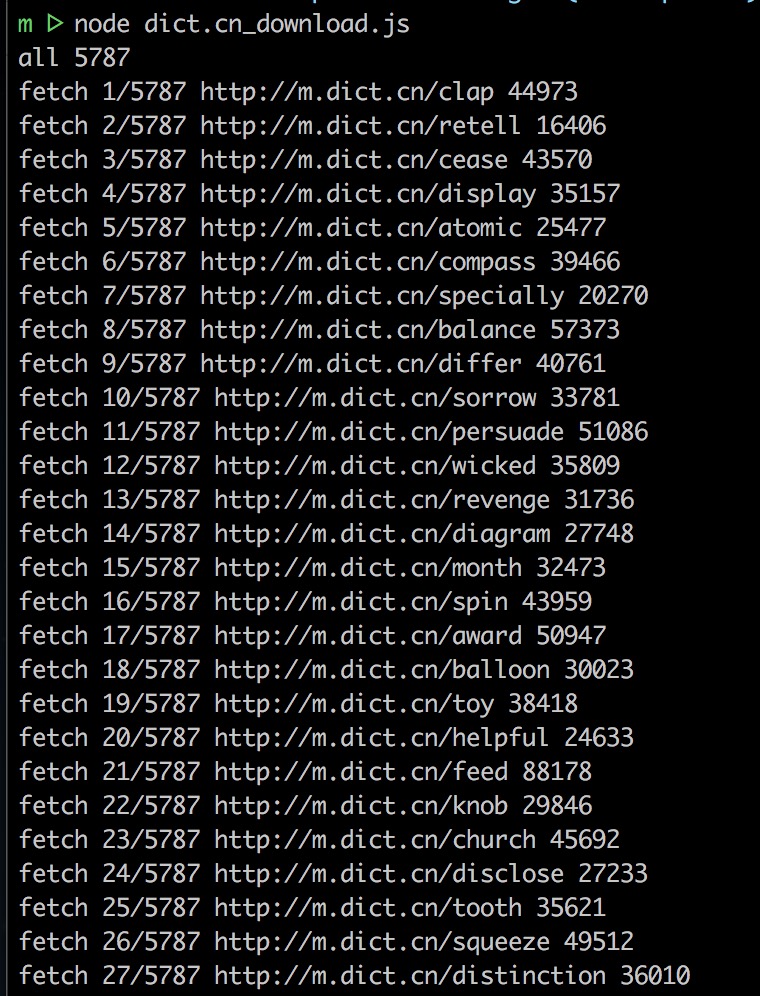
1. 请求扇贝网单词书页面单词的列表分组信息。
2. 分析其分组列表，取出对应标签下的索引信息（$('.wordbook-wordlist-name a', $(menuList[i])).attr('href')）。
3. 循环获取每一个分组的单词列表页面。
4. 分析单词列表页面，获取到单词信息（$('.span2 strong', $(pageWordList[k])).text().trim()）。
5. 将获取到的单词列表一次push到初始定义的空数组中，并利用Web形式将数组输出。

(6) 将输出文件保存为JSON格式即为所需单词列表。

6.1.2 单词信息采集模块实现

单词信息采集过程我使用了Node.js中的Crawler模块，获取的具体步骤如下：

1. 导入先前保存的词库文件。
2. 连接数据库。
3. 对于每一个单词请求海词对应单词的界面信息（ http://m.dict.cn/${word}）。
4. 将每一个请求加入到请求队列中。
5. 定义一个爬虫，完成队列中的每一个请求，将获取的HTML源代码保存到数据库。

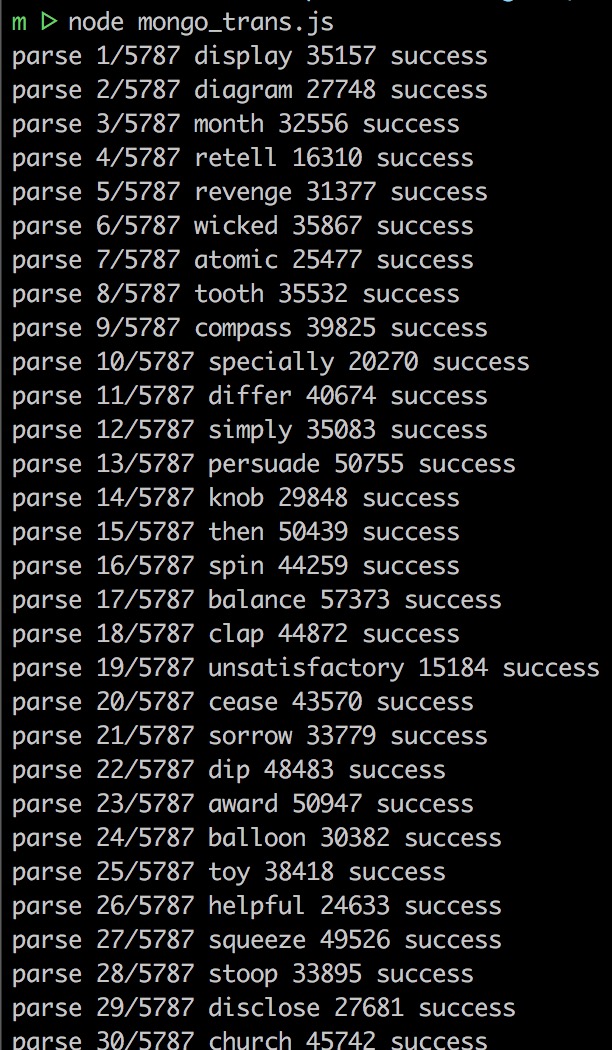


**图 6.1 单词信息采集爬虫运行截图**

6.1.3 数据清洗模块实现

数据清洗模块使用的是Node.js中的Cheerio模块实现，清洗步骤如下：

1. 依次从数据库中取出保存的HTML源代码。
2. 利用模块加载对应源代码，并进行分析。
3. 取出对应标签中的信息，转换为JSON格式。
4. 将获取到的信息存入数据库。



**图6.2 数据清洗模块运行截图**

6.1.4 数据验证模块实现

为了使应用可以更好的运行，用户也不会获取到一些比较奇怪的的数据，需要对数据进行验证，验证过程如下：

1. 遍历每一条记录，对获取的信息进行判断。这里使用Node.js中的tcomb模块来验证数据
2. 对于验证失败的数据，用JSON格式输出到一个文件夹，手动检查数据哪些字段存在问题。
3. 对于可以自动修复的数据，修改验证模块重新验证

(4) 对于部分不能自动修复的数据，手动订正数据并更新数据库

6.2 后端API接口实现

6.2.1 获取用户信息的API实现

查找数据库中该用户的信息

如果该用户存在

查找该用户是否选择了单词书

如果用户选择了单词书

返回带有单词书信息的用户信息

否则

返回不带有单词书信息的用户信息

否则

返回提示用户注册的错误信息



**图6.3 获取用户信息的API实现流程图**

6.2.2 修改用户信息的API实现

查找请求的用户是否存在

如果用户存在

查找参数中时候存在wordbooknow且不为空

如果存在

修改对应的wordbooknow为新值

查找参数中时候存在nickname且不为空

如果存在

修改对应的nickname为新值

否则

返回错误信息

注意：参数是可选的，它会根据你传递的参数进行对应的修改，空值会被忽略掉。



**图6.4 获取用户信息的API实现流程图**

6.2.3 获取单词书列表API实现

6.2.4 获取某本单词书单词列表的API实现

读取数据库中单词书信息并返回其名称和描述的列表

根据单词书名称在数据库中查找该单词书

如果该单词书存在

返回该单词书的所有单词信息

否则

返回单词书不存在的错误信息



**图6.5获取单词书列表API实现流程图**

6.2.5 获取用户单词分组列表的API实现

如果存在

根据用户ID和单词书的ID找到用户的单词分组信息

如果存在

返回对应信息

否则

返回分组信息不存在的提示信息

否则

返回用户不存在或者用户需选择单词书的提示信息



**图6.6 获取用户单词分组列表的API实现流程图**

6.2.6 获取某个单词分组的API实现

根据用户名获取到用户的当前单词书

如果存在

根据用户ID和单词书的ID找到用户的单词分组信息

如果存在

根据传入的id参数，获取id对应的分组，并返回该分组的所有信息

否则

返回分组信息不存在的提示信息

否则

返回用户不存在或者用户需选择单词书的提示信息



**图6.7 获取某个单词分组的API实现流程图**

6.2.7 上传背诵记录的API实现

根据用户名获取到用户的当前单词书

如果存在

根据用户ID和单词书的ID找到用户的单词分组信息

如果存在

根据传入的id参数，获取id对应的分组，将背诵信息添加到数据库

否则

返回分组信息不存在的提示信息

否则

返回用户不存在或者用户需选择单词书的提示信息



**图6.8 上传背诵记录的API实现流程图**

6.2.8 生成分组信息的API实现

根据用户名获取到用户的当前单词书

如果存在

根据用户ID和单词书的ID找到用户的单词分组信息

如果存在

返回分组已经存在的提示信息

否则

根据用户设置，计算出分组情况，并写入数据库

否则

返回用户不存在或者用户需选择单词书的提示信息



**图6.9 生成分组信息的API实现流程图**

6.2.9 删除分组信息的API实现

根据用户名获取到用户的当前单词书

如果存在

根据用户ID和单词书的ID找到用户的单词分组信息

如果存在

删除分组信息

否则

直接返回成功

否则

返回用户不存在或者用户需选择单词书的提示信息



**图6.10 删除分组信息的API实现流程图**

6.3 前端逻辑实现

6.3.1前端ACTION定义

**表6.11 前端ACTION定义表**

|  |  |
| --- | --- |
| 常量定义 | 描述 |
| USER\_API\_CALL | 用户信息API调用 |
| USER\_API\_SUCCESS | 用户信息API调用成功 |
| USER\_API\_ERROR | 用户信息API调用失败 |
| WORDGROUP\_LIST\_API\_CALL | 单词组列表API调用 |
| WORDGROUP\_LIST\_API\_SUCCESS | 单词组列表API调用成功 |
| WORDGROUP\_LIST\_API\_ERROR | 单词组列表API调用失败 |
| WORDGROUP\_API\_CALL | 单词组API调用 |
| WORDGROUP\_API\_SUCCESS | 单词组API调用成功 |
| WORDGROUP\_API\_ERROR | 单词组API调用失败 |
| WORDGROUP\_LEARN\_SET\_VIEW | 设置当前单词背诵方式 |
| WORDGROUP\_LEARN\_SET\_INDEX | 设置当前背诵单词的索引 |
| GLOBAL\_TOAST | 发起全局通知 |
| SET\_WORD\_REMEMBER | 标记某个单词为已经记住 |
| SET\_WORD\_DIFFICULT | 标记某个单词为难词 |
| SET\_LIST\_VIEW\_SHOWALL | 切换列表页面显示全部单词和已经记住 |

6.3.2 前端ACTION提供的方法

**表6.12 前端ACTION定义表**

|  |  |
| --- | --- |
| 方法 | 描述 |
| loadUser() | 加载用户信息 |
| loadWordGroupList() | 加载单词组列表 |
| loadWordGroup(groupID) | 加载某个单词组 |
| wordbookLearnSetView(view) | 设置单词背诵方式 |
| wordbookLearnSetIndex(index) | 设置单词背诵单词的索引 |
| globalToast(msg) | 发起全局通知 |
| wordSetRemember(groupID, wordID) | 标记某组的某个单词已经记住 |
| wordSetDifficult(groupID, wordID) | 标记某组的某个单词为难词 |
| setListViewShowALL(showAll) | 切换列表页面显示全部单词和已经记住 |

6.3.3 前端Middleware实现

前端使用了 redux-thunk、redux-localstorage这两个开源中间件，同时还自定义了一个API中间件。

1. redux-thunk 用于支持异步ACTION。
2. redux-localstorage 支持把STORAGE保存到浏览器localStorage，具体的表现是：用户进行任何操作之后立刻按下浏览器的刷新按钮或者关闭浏览器再进入，应用的界面会和之前的一模一样。而不存在传统Web应用会丢失数据的可能。
3. API 用户发送API请求，同时对于服务器错误的情况，默认对用户给出友好的提示信息。

6.3.4 前端Reducer和Store实现

* Root
  + User

保存用户信息，当用户信息API成功（USER\_API\_SUCCESS）的时候被更新。

* + WordGroupList
    - Loaded

分组信息加载成功的标志。

* + - GroupNum

用户目前的所有的分组的数目，当分组列表加载成功（WORDGROUP\_LIST\_API\_SUCCESS）的时候被更新。

* + WordGroup

分组数据，是一个键为组ID、值为分组数据的Dict，包含了背诵所需的单词列表、释义等信息，当分组加载成功（WORDGROUP\_API\_SUCCESS）的时候被更新。

* + WordGroupLearn
    - View

当前是卡片视图（Card）还是列表视图（List），当收到WORDGROUP\_LEARN\_SET\_VIEW的ACTION的时候被更新。

* + - Index

当前背诵单词的索引，当收到WORDGROUP\_LEARN\_SET\_INDEX的ACTION的时候被更新。

* + - ShowAll

当前列表页面是显示全部单词还是未记住单词，当收到SET\_LIST\_VIEW\_SHOWALL的ACTION的时候被更新。

* + WordGroupStat

单词背诵记录，是一个键为组ID、值为背诵数据的Dict，背诵数据是一个键为单词ID、值为单词状态的Dict。当收到SET\_WORD\_REMEMBER、SET\_WORD\_DIFFICULT的ACTION的时候被更新。

* + Toast

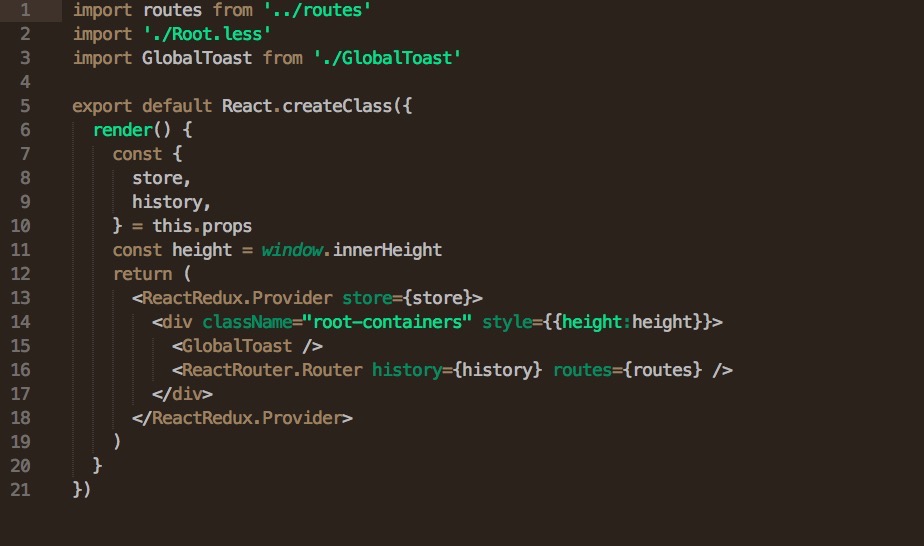
最后一条通知的内容和触发时间，当收到GLOBAL\_TOAST的ACTION的时候被更新。

6.4 前端组件实现

6.4.1 Root组件实现

Root组件是整个应用的根组件，主要为下层组件提供了STORE的context，

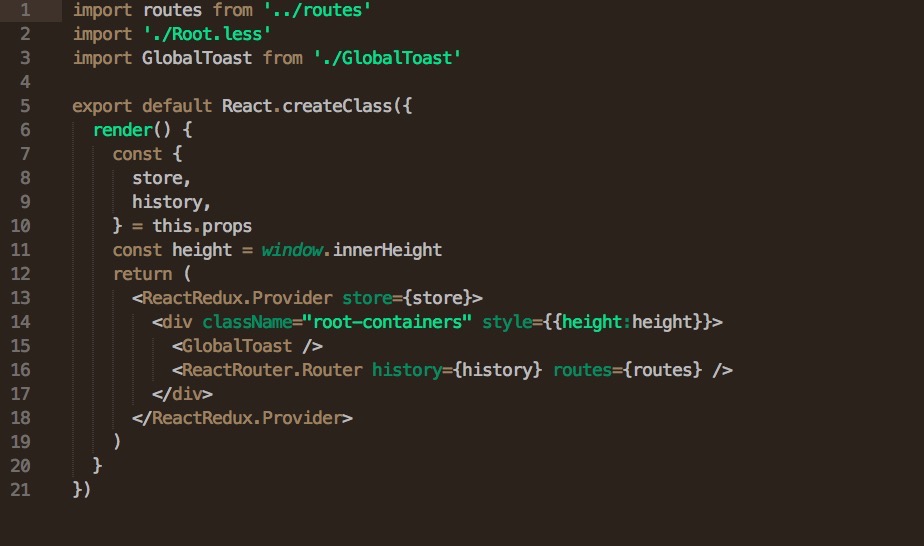
其子组件为Router组件和GlobalToast组件



**图6.13 Root组件实现部分代码**

6.4.2 Router组件实现

定义了整个应用的路由，路由使用了react-router这个路由模块。其子组件为IndexPage、WordGroup、WordGroupLearn、Config。



**图6.14 Router组件实现部分代码**

6.4.3 GlobalToast组件实现

提供类似安卓系统显示在屏幕下方的全局通知的功能，从STORE的toast字段中获取通知并显示。

主要逻辑：全局存放了一个通知计数器，如果STORE中有新值，并且通知长度大于0，则把通知计数器加1，用CSS3实现的淡入效果显示通知，并按通知里面的显示时间设定一个定时器。在定时器的回调里面，检测目前的通知计数器的值和设定计数器的时候的值是否一致，如果一致则认为这段时间里面没有新的通知要显示，然后用CSS3实现的淡出效果隐藏掉通知。如果不一致则不进行任何操作。



**图6.15 GlobalToast组件部分代码**



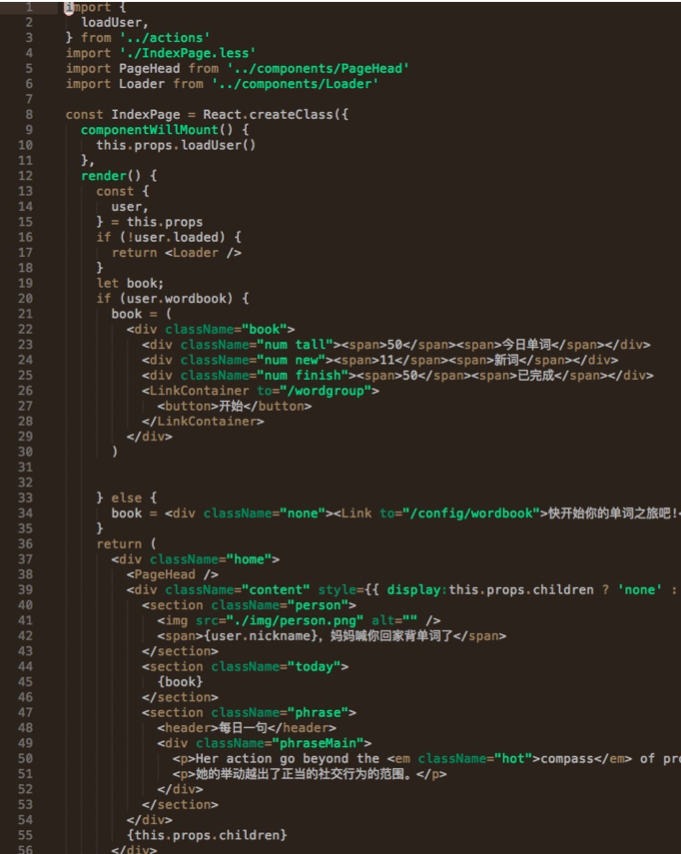
**图6.16 GlobalToast组件运行效果**

6.4.4 IndexPage组件实现

IndexPage用于显示用户进入App之后看到的首页，其逻辑比较简单。如果用户数据没有加载则显示加载动画，否则显示首页。首页提供开始按钮让用户跳转到分组列表页面。

在组件加载的时候会触发一次获取用户信息的操作，这样实际上用户是先看到上次进去的时候缓存的数据，同时程序开始试图刷新数据，如果新数据和旧数据一致则不进行页面更新，否则对数据计算差异并更新页面。

相比传统Web应用每次用户进入都必须等待数据加载完成之后才能看到界面的方式用户体验有了很大提升。

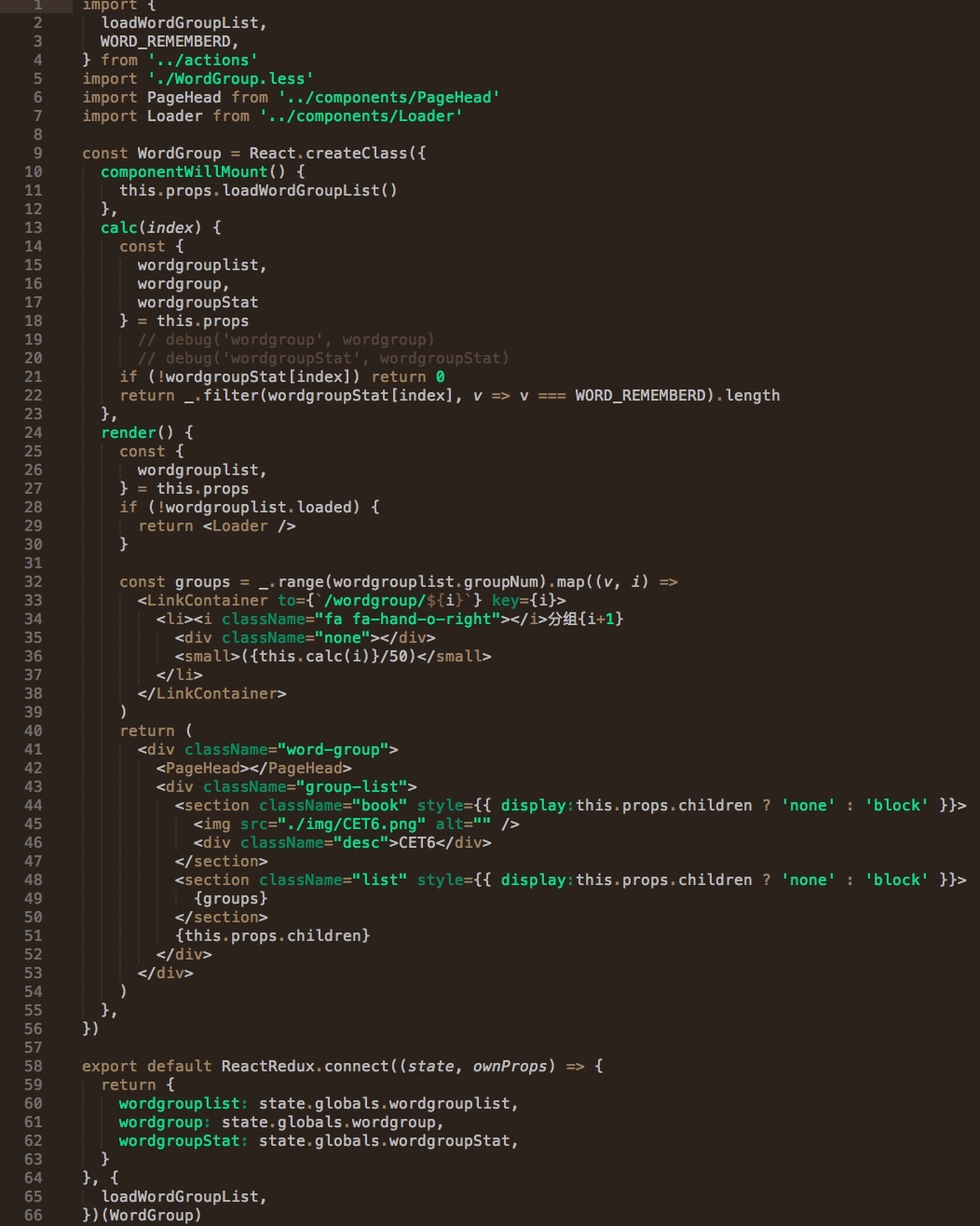


**图6.17 IndexPage 组件部分代码实现**

6.4.5 WordGroup组件实现

WordGroup组件用于显示用户的分组列表，逻辑也较为简单。如果分组数据没有加载则显示加载动画，否则显示分组列表。分组列表里面的背诵统计信息则是根据STORE里面的WordGroupStat字段计算出来。

WordGroup组件加载的时候也和IndexPage一样会去刷新数据。



**图6.18 WordGroup组件部分代码实现**

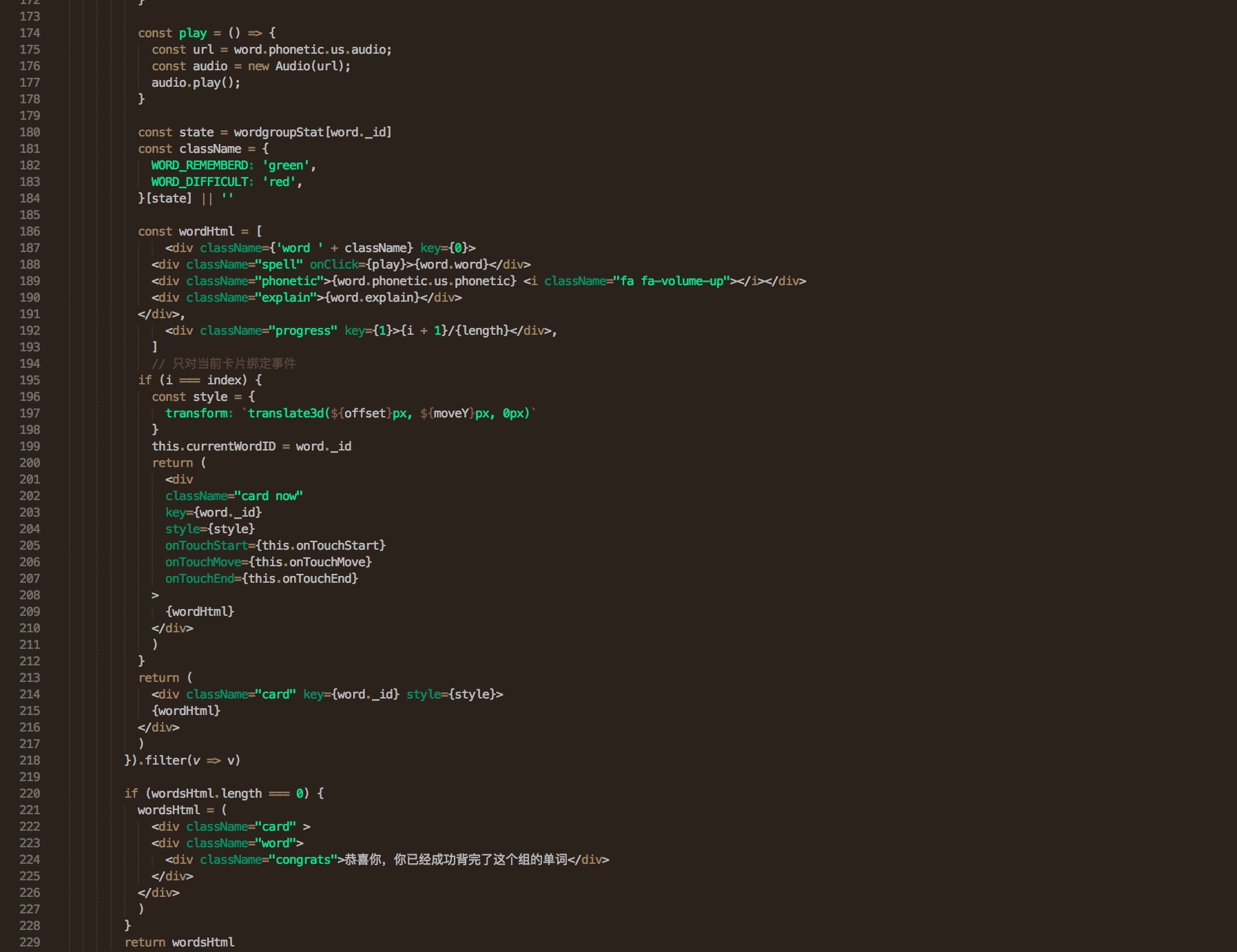
6.4.6 WordGroupLearn组件实现

这个组件是用于显示用户背诵单词的时候看到的界面，其包含两个子组件，如果STORE中当前背诵方式为Card则显示WordGroupLearnCard组件，否则显示WordGroupLearnList组件。

6.4.7 WordGroupLearnCard组件实现

这个组件是用于显示用户卡片背诵单词的时候看到的界面，因为处理了较多的手势和动画，逻辑非常复杂。

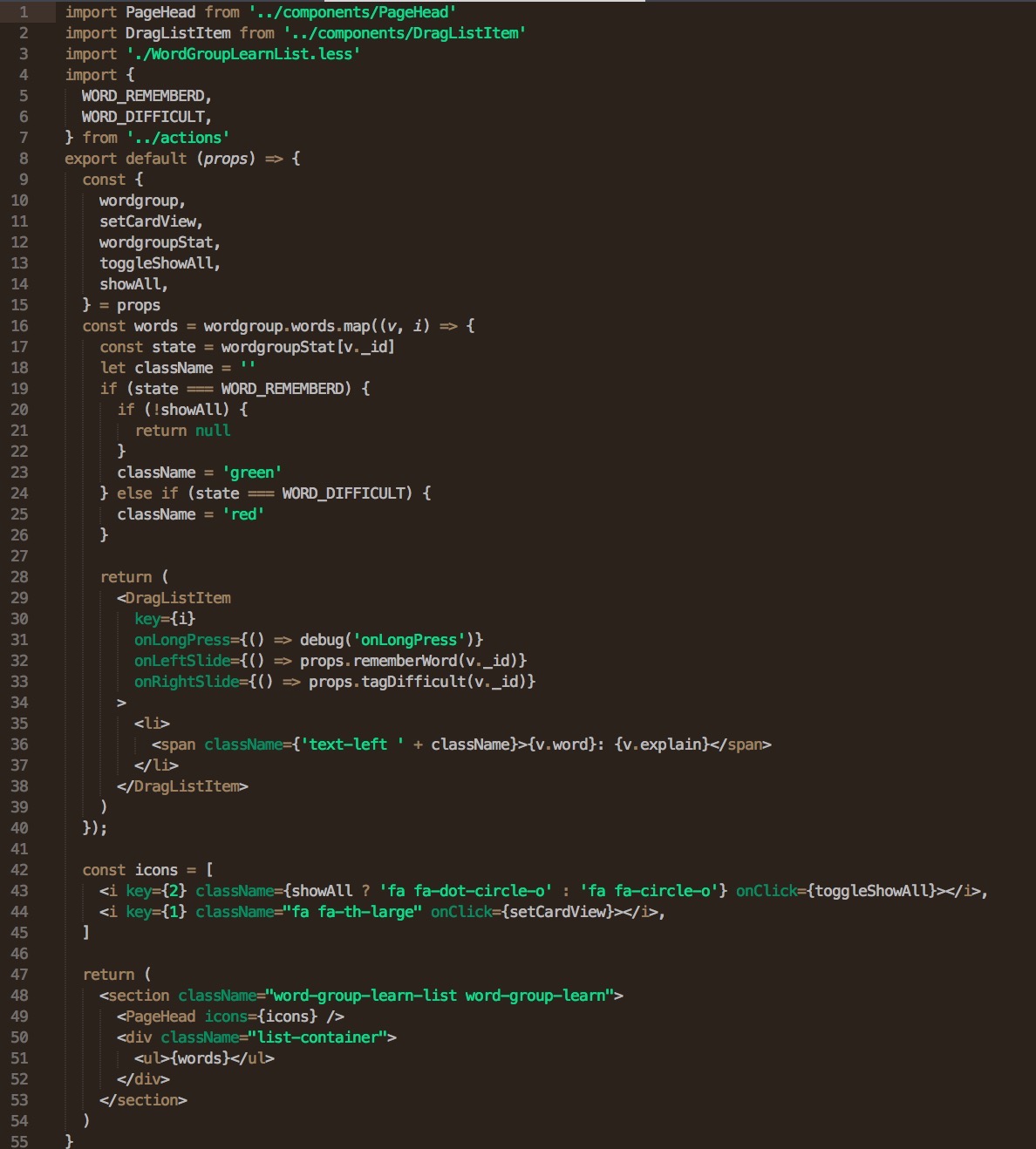
1. 单词卡片左右滑动效果的实现
   1. 首先把每个单词渲染成一张单词卡片用CSS的相对定位使其在屏幕的同一位置重合。
   2. 然后根据每个单词的索引计算X轴偏移值使其从屏幕开始一张张向右排开，卡片之间保留一定的间隙是屏幕刚好可以看到一张卡片和左右卡片的边缘。
   3. 然后根据当前单词的索引计算X轴偏移值，是所有单词卡片向左移动到合适位置让当前卡片刚好显示在屏幕正中间。
   4. 如果当前单词有变化则重新计算偏移值，然后以当前偏移值和新的偏移值为参数计算出缓动动画。
   5. 然后就是优化性能，实际上用户可以看到的只有当前卡片、左边一个卡片、右边一个卡片这三个卡片，其它卡片实际上是在屏幕之外看不到的，所以实际上我们计算的时候可以跳过这些卡片，只渲染中间的三张卡片。然后在每一次翻页结束之后重新计算出三张卡片。
   6. 因为这三张卡片的定位都是独立的，所以像翻页的时候在最右边添加一张卡片然后移除左边的卡片这样的DOM操作不会造成动画闪烁。这样就在兼顾内存占用、动画性能、用户体验的前提下实现了翻页动画。
2. 缓动动画的实现
   1. 计算出物体当前位置和目标点的位置的距离
   2. 把运动物体当做一个被拉伸的弹簧，为运动确定一个劲度系数和阻尼系数，由物理定律可知，物体受到的空气阻力和物体的速度正相关，受到的弹力与弹簧拉伸的距离即运动物体与目标位置的距离正相关。
   3. 每过一段极小的时间，通过以当前速度和当前位移用物理公式计算下一个间隔时间的速度和位移。虽然这样的计算是不精确的，但是还是足够骗过人的眼睛。
   4. 上述这段时间间隔的选取可以用浏览器的requestAnimationFrame API来自动选取（一般每秒60次），这样恰好浏览器渲染一帧就重新计算一次位置，不会有多余的计算。
3. 手势的实现
   1. 在当前卡片监听了触摸开始、触摸移动、触摸结束这三个浏览器事件。
   2. 内部保存了当前坐标、动画起点坐标、动画当前坐标、终止状态这几个状态值。
   3. 在触摸开始事件被触发的时候，修改当前状态为动画准备。之前默认值为正常。
   4. 在触摸移动事件被触发的时候，如果当前状态为动画准备，则设定动画坐标状态、动画当前坐标状态为当前触摸坐标。   
      如果当前X轴的偏移或者Y轴的偏移大于某一个阈值，并且另一个坐标轴的偏移小于这个阈值的一半，则认为用户在进行这个方向的手势操作，修改终止状态为对应的方向。
   5. 在触摸结束事件被触发的时候，检查终止状态，如果不为空，则触发对应的事件如记住当前单词，然后重置各个状态的值回默认值。手势操作结束。之后利用缓动动画移动卡片到新的位置或者原来的位置。
   6. 触摸的时候对应的界面更新：如果有左右偏移（动画起点坐标X轴与动画当前坐标X轴之差），则把所有的卡片都加上左右的卡片。如果有上下方向的偏移，则仅对当前卡片加上上下方向的偏移，效果就是当前卡片跟着用户手指上下左右移动。其它卡片在左右方向上都跟着用户的手指移动。



**图6.19 WordGroupLearnCard 组件部分代码实现**

6.4.8 WordGroupLearnList组件实现

WordGroupLearnList用于显示用户在卡片下背诵单词的时候看到的页面。主要逻辑：根据单词数据渲染出单词列表，然后对于每一行，加上和上面的WordGroupLearnCard类似的动画效果。表现为用户左滑标记当前单词已经记住，右滑标记当前单词为难词。



**图 6.20 WordGroupLearnList组件部分代码实现**

6.5 系统测试

6.5.1 后端API功能测试

后端开发过程中采用TDD（测试驱动开发），开发流程：

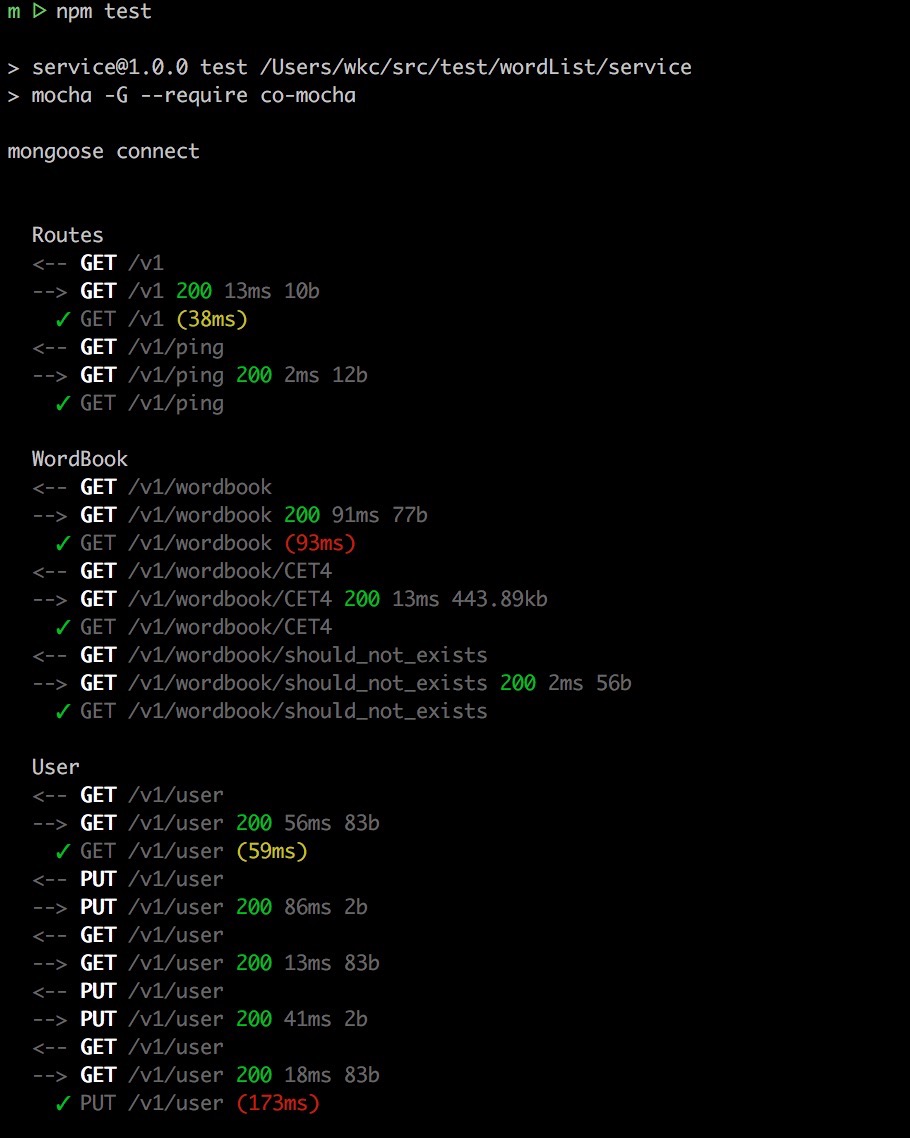
1. 会先写测试用例造成测试失败
2. 写业务代码使其测试通过
3. 在保证测试还能通过的前提下重构上面写的业务代码，然后继续从第一步循环

开发过程中还使用了istanbul来为代码生成了测试覆盖率报告，如果发现某些代码覆盖率过低就为这个地方加上测试。

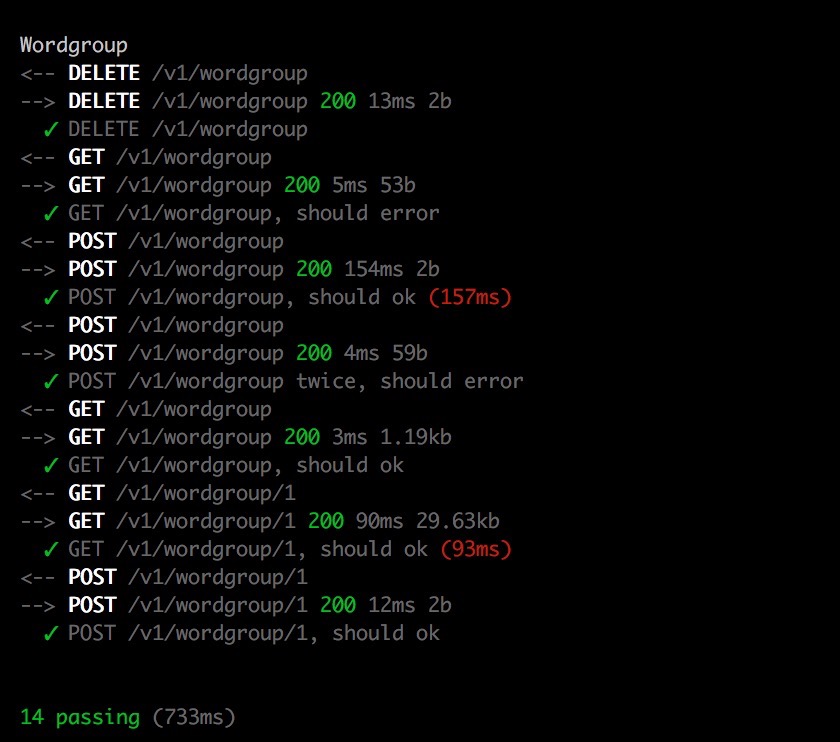
代码托管在GitHub，同时启动了Travis CI在每次代码提交之后自动运行测试，如果测试出现问题就会发邮件通知开发者来修复。

**表6.21后端API部分测试用例**

|  |  |  |  |
| --- | --- | --- | --- |
| 操作 | 描述 | 期望结果 | 实际结果 |
| GET /v1/ping | Ping后端服务器 | 返回200 | 同期望结果 |
| GET /v1/wordbook | 获取单词书列表 | 返回结果条数大于1 | 同期望结果 |
| GET /v1/wordbook/CET4 | 获取某本单词书 | 返回结果长度大于2K | 同期望结果 |
| GET /v1/wordbook/should\_not\_exists | 获取不存在的单词书 | 返回单词书不存在的错误信息 | 同期望结果 |
| GET /v1/user | 未登陆状态下获取用户信息 | 返回用户未登陆的错误信息 | 同期望结果 |
| POST /v1/user | 登陆用户 | 返回200 | 同期望结果 |
| GET /v1/user | 获取用户信息 | 返回正确的用户信息 | 同期望结果 |
| PUT /v1/user | 修改用户信息 | 返回200 | 同期望结果 |
| GET /v1/user | 获取新的用户信息 | 用户信息已经成功被更新 | 同期望结果 |
| DELETE /v1/wordgroup | 删除分组信息 | 返回200 | 同期望结果 |
| GET /v1/wordgroup | 获取分组信息 | 返回分组不存在的错误信息 | 同期望结果 |
| POST /v1/wordgroup | 创建分组 | 返回200 | 同期望结果 |
| POST /v1/wordgroup | 重新创建分组 | 返回分组已经存在的错误信息 | 同期望结果 |
| GET /v1/wordgroup | 获取分组信息 | 返回200并且分组数目大于5 | 同期望结果 |
| GET /v1/wordgroup/1 | 获取第一组 | 返回200并且单词数目大于10 | 同期望结果 |
| GET /v1/wordgroup/-1 | 获取错误的分组 | 返回分组不存在的错误 | 同期望结果 |
| DELETE /v1/user | 退出登录 | 返回200 |  |



**图 6.22 后端API测试截图**



**6.23 后端API测试截图二**

6.5.2 前端功能测试

前端开发由于采用了React，逻辑代码和界面代码得以较好的分离开来。逻辑部分采用了TDD方式开发。这部分的测试主要是单元测试，测试用例比较琐碎就不一一列举。

视图部分由于涉及到动画和手势，难以自动化测试，这部分使用手动测试。

**表 6.24 前端功能测试手动测试部分用例列表**

|  |  |  |  |
| --- | --- | --- | --- |
| 操作 | 描述 | 期望结果 | 实际结果 |
| 打开应用进入登录界面，点击微信登录 | 测试登录功能 | 登录成功 | 同期望结果 |
| 进入选择单词书页面，选择四级单词书 | 测试单词书功能 | 选择成功 | 同期望结果 |
| 进入分组创建页面，选择每日单词书，点击创建 | 测试创建单词分组 | 创建成功 | 同期望结果 |
| 返回首页，点击开始，进入单词分组列表 | 测试选择单词分组 | 进入成功 | 同期望结果 |
| 点击任意分组进入背诵页面 | 测试背诵功能是否正常 | 进入成功 | 同期望结果 |
| 上下左右滑动操作 | 测试各种手势能否正常使用 | 可以正常使用 | 同期望结果 |
| 点击切换到列表视图按钮 | 测试列表背诵方式是否正常 | 可以正常使用 | 同期望结果 |
| 点击切换全部单词和未记住单词 | 测试切换全部单词和未记住单词功能是否正常 | 可以正常使用 | 同期望结果 |
| 列表页面上下左右手势操作 | 测试各种手势能否正常使用 | 可以正常使用 | 同期望结果 |
| 返回分组页面，查看分组统计数据 | 测试分组统计功能是否正常 | 可以正常使用 | 同期望结果 |

部分截图如下：

|  |  |  |  |
| --- | --- | --- | --- |
| C:\Users\hi\Documents\Tencent Files\1057275848\Image\C2C\XN3HZPS4P[%G]N9E62FFYX1.jpg | | C:\Users\hi\Documents\Tencent Files\1057275848\Image\C2C\{$MS9YP3%RJMJ}MV8})%P46.jpg | |
| C:\Users\hi\Documents\Tencent Files\1057275848\Image\C2C\D)Y~EEQRRT`2N(E46)_T)}3.jpg | | C:\Users\hi\Documents\Tencent Files\1057275848\Image\C2C\Z[IDL3KX`~@QU9IM00O91EA.jpg | |
| C:\Users\hi\Documents\Tencent Files\1057275848\Image\C2C\%MXLRA}9QX)N_RAY4_WCFB4.jpg |  | | C:\Users\hi\Documents\Tencent Files\1057275848\Image\C2C\0BOHKF2~5S~(3_L)37M1V5G.jpg |
| C:\Users\hi\Documents\Tencent Files\1057275848\Image\C2C\SA721ET3@CYT7TZN`D9ISVA.jpg |  | | C:\Users\hi\Documents\Tencent Files\1057275848\Image\C2C\MNKW`(ZR~$@%O4E_]5%LC52.jpg |

**图6.25 前端功能测试截图**

1. 总结

本项目是以更好的方便学生背单词为目的，集合了市面上已有背单词软件的诸多优势，针对现有软件的不足（如冗余功能多，操作过于繁琐，广告多，用户体验差等）而开发的一个简洁、精美、小巧的基于WEB平台的背单词手机应用。

项目中使用了Webstorm为主要开发工具，Koa作为后端开发框架，Mongodb作为后端数据库，React+Redux作为前端页面的主要框架，Webpack作为前端资源的构建打包工具。数据存储使用了浏览器localStorage，使用react-router实现了页面跳转无刷新技术，使用了浏览器的touch事件和CSS3的transition实现了手势操作和复杂的动画效果，在Web平台上实现了媲美原生应用的体验。

设计开发本项目的过程中，充分研究了市场现有软件的设计思想，并对软件的功能和使用进行了研究分析，并从中找出了各种软件的优势以及它们存在的需要改进的问题，针对这些不足，我分别采用了不同的解决方案逐一解决优化。

下面我对项目中的特色做详细介绍：

（1）养眼颜色搭配

项目没有使用传统的黑白两色，也没有使用非常亮眼的红色，黄色之类的颜色，而是使用了养眼的浅绿色，这种颜色更给眼睛一种舒适感，尽可能的避免用户在使用过程中产生眼疲劳，保持更加愉悦的心情，从而更好的提高效率。

（2）简洁的界面设计

项目在界面设计过程中，去除了很多繁杂的按钮和模块，背单词页面单纯由背单词模块组成，这种设计，简洁大方，更好的突出项目的主题而没有冗余的功能，用户在使用过程中也不会因为页面中的其他模块的存在而分心，从而更加专注。

（3）全方位的手势操作

手势操作相对于如今很多市场上流行软件的按钮操作，更加能凸显出智能手机的优势，手势操作更加快捷，既节省了屏幕空间，又提升了用户体验。通过手势来进行单词的跳转和标记单词的背诵情况，提高了用户使用软件的兴趣，从而使效率得以提高。

（4）流畅的动画实现

在实现动画的过程中，充分考虑到了操作过程中可能出现的各种情况，读单词时的单词放大和缩小的变化，手势动作的操作页面的动作等，动画模拟了物理学中的弹簧运动，让用户感觉更自然。

（5）巨量的词库提供

词库的数量和质量是评价一个背单词软件优劣的重要依据，词库不仅包含大学生所需的CET4和CET6的词汇，还包含了从小学到研究生博士所需的词库以及像医用，军用等不同分类的词库，为不同的用户提供了选择。

（6）权威的真人发音

单词的学习，不仅仅是拼写和解释的学习，更重要的是发音的学习，因此，权威的真人发音就成为了必然需求。在选取发音的过程中，我爬取了不同网站的发音文件，并且做了细致的筛选，使每一个单词的发音清晰标准。

（7）人性化的流量优化

流量的使用会是很多用户非常关心的问题，所以尽可能的节省流量也成了项目中必不可少的需求。系统会检测用户当前使用的网络类型，如果是WiFi网络，系统会尽可能的下载用户可能用到的数据，如果是移动网络，则只会下载当前用户所需的数据，发音的质量，背诵情况的同步也会根据网络的类型有所更改，这样使用户使用过程中不需担心流量使用过多的问题。在数据的传输方面，采用了HTTP2.0，不仅仅节省了流量还省了不少电量。背诵情况的同步方式也采用了基于时间戳的增量传输，大大减低了流量消耗。

（8）智能的用户提醒

项目会根据用户的背诵习惯推测用户当天背单词的时间，如果超过了预计的时间而用户还没有完成当天的计划则会有相关的提醒，这对用户来说，在养成良好习惯的道路上起到了提醒的作用。并且，如果用户绑定了微信或者微博的情况下，当天没有完成任务会自动发送一条朋友圈或者微博来提醒用户。

由于本人的能力十分有限，导致系统还存在着许多的不足之处，如使用了较多的浏览器新特性，在旧版本的安卓系统下可能会出现兼容性问题。动画比较多对机器性能要求比较高，配置比较低的安卓机器可能存在动画帧率低、卡顿的情况。希望在以后的之后的工作与学习中持续提升个人能力，继续完善此项目。

参考文献

[1] Cassio, de, Sousa, Antonio. Pro React[M]. Apress; 1st ed. 2015 edition:Cassio de Sousa Antonio, 2015.

[2] Biography. Learning React Native: Building Native Mobile Apps with JavaScript[M]. O'Reilly Media:Biography, 2016.

[3] Juho, Vepsäläinen. SurviveJS - Webpack and React: From apprentice to master[M]. CreateSpace Independent Publishing Platform:Juho Vepsäläinen, 2016.

[4] Paperback. Developing a React Edge: The JavaScript Library for User Interfaces[M]. Bleeding Edge Press:Paperback, 2014.

[5] Jim, R, Wilson. Node.js the Right Way: Practical, Server-Side JavaScript That Scales[M]. Pragmatic Bookshelf:Jim R. Wilson, 2013.

[6] Ethan, Brown. Web Development with Node and Express: Leveraging the JavaScript Stack[M]. O'Reilly Media:Ethan Brown, 2014.

[7] David, Flanaga. JavaScript: The Definitive Guide: Activate Your Web Pages [M]. O'Reilly Media:David Flanaga, 2015.

[8] Douglas, Crockford. JavaScript: The Good Parts[M]. O'Reilly Media:Douglas Crockford, 2008.

[9] Stoyan, Stefanov. JavaScript Patterns[M]. O'Reilly Media:Stoyan Stefanov, 2010.

[10] Ethan, Brown. Learning JavaScript: JavaScript Essentials for Modern Application Development[M]. O'Reilly Media:Ethan Brown, 2016.

[11] Eric, T, Freeman. Head First JavaScript Programming[M]. O'Reilly Media:Eric T. Freeman, 2014.

[12] Ravi, Kumar, Gupta. Test-Driven JavaScript Development[M]. Packt Publishing:Ravi Kumar Gupta, 2015.

[13] Clement, Nedelcu. Nginx HTTP Server - Third Edition[M]. Packt Publishing:Clement Nedelcu, 2015.

[14] Steve, Corona. nginx: A Practical Guide to High Performance[M]. O'Reilly Media:Steve Corona, 2016.

[15] Lea, Verou. CSS Secrets: Better Solutions to Everyday Web Design Problems[M]. O'Reilly Media:Lea Verou, 2015.

致谢

在老师的悉心指导和自己的不懈努力下，毕业设计终于完成了。总体来说，项目的功能已基本实现，在很多细节方面也做了很多完善，拥有良好的用户体验。当然，由于时间有限，项目在功能上还有待增加，兼容性问题还没有得到良好的解决，这些问题将会在后面的学习生活中继续完善，开发出更加优秀的项目。

通过本次毕业设计的完成，让我对大学四级的学习有了一个系统的认识，从需求分析到最后项目的测试使用，让我真正体会到了一个简单项目实现的不易。在以后的开发过程中我还要多加学习，多去实践，这里面的学问需要用心去体会，去挖掘。

在毕业设计的制作过程中，老师，同学都给了我很大的帮助。在这里我要特别感谢我的指导老师姜磊老师，他是一位和蔼可亲，认真负责的好老师。从大一开始，他就告诉我们学习技术的重要性，给我们提了很多学习方面的建议和方法，让我们在这条追随科学技术发展的路上乘风破浪，更胜一筹。在论文的完成工程中，它给予了我很多的指导，从选题、定题、项目的完成到论文的修改，姜磊老师都很耐心的指导我，给我提了很多建设性的意见。在姜磊老师的耐心帮助下，我的毕业论文才得以顺利完成。

我还要感谢在项目开发过程中给予我很大帮助的李同学。在开发过程中我遇到了很多我没有接触过的技术和工具，是李同学为了解答了很多我不是很能理解的问题以及很多编码过程中前端方面遇到的一些疑惑，都是他给我一一解答。在设计方案方面，他也给我提了很多他的建议和想法，让项目更加的完善。

同时，我还要感谢大学四年中给予我很大帮助的叶同学，它带我加入了学院ACM集训队，我们一起做专题，一起打比赛，一起讨论问题，让我在算法方面的造诣有了很大的提高。