东 华 大 学

软件工程课程设计实验报告

目录

[一、项目开发计划](#_Toc28282_WPSOffice_Level1) [4](#_Toc28282_WPSOffice_Level1)

[1、引言](#_Toc19350_WPSOffice_Level2) [4](#_Toc19350_WPSOffice_Level2)

[1.1 编写目的](#_Toc19350_WPSOffice_Level3) [4](#_Toc19350_WPSOffice_Level3)

[1.2 项目背景](#_Toc31650_WPSOffice_Level3) [4](#_Toc31650_WPSOffice_Level3)

[1.3 定义](#_Toc25048_WPSOffice_Level3) [4](#_Toc25048_WPSOffice_Level3)

[1.4 参考资料](#_Toc29401_WPSOffice_Level3) [5](#_Toc29401_WPSOffice_Level3)

[2、项目概述](#_Toc31650_WPSOffice_Level2) [5](#_Toc31650_WPSOffice_Level2)

[2.1 工作内容](#_Toc16606_WPSOffice_Level3) [5](#_Toc16606_WPSOffice_Level3)

[2.2 主要参加人员](#_Toc12258_WPSOffice_Level3) [6](#_Toc12258_WPSOffice_Level3)

[2.3 产品](#_Toc780_WPSOffice_Level3) [6](#_Toc780_WPSOffice_Level3)

[2.4 验收标准](#_Toc26275_WPSOffice_Level3) [7](#_Toc26275_WPSOffice_Level3)

[2.5 完成项目最迟日期](#_Toc11150_WPSOffice_Level3) [7](#_Toc11150_WPSOffice_Level3)

[2.6 本计划的批准者和批准日期](#_Toc14569_WPSOffice_Level3) [7](#_Toc14569_WPSOffice_Level3)

[3、实施计划](#_Toc25048_WPSOffice_Level2) [7](#_Toc25048_WPSOffice_Level2)

[3.1 工作任务的分解和人员分工](#_Toc15090_WPSOffice_Level3) [7](#_Toc15090_WPSOffice_Level3)

[3.3 进度](#_Toc15164_WPSOffice_Level3) [8](#_Toc15164_WPSOffice_Level3)

[3.4 关键问题](#_Toc22817_WPSOffice_Level3) [8](#_Toc22817_WPSOffice_Level3)

[4、专题计划要点](#_Toc29401_WPSOffice_Level2) [8](#_Toc29401_WPSOffice_Level2)

[二、需求分析](#_Toc19350_WPSOffice_Level1) [8](#_Toc19350_WPSOffice_Level1)

[三、概要设计](#_Toc31650_WPSOffice_Level1) [9](#_Toc31650_WPSOffice_Level1)

[1、总体设计](#_Toc16606_WPSOffice_Level2) [9](#_Toc16606_WPSOffice_Level2)

[1.1 需求规定](#_Toc19795_WPSOffice_Level3) [9](#_Toc19795_WPSOffice_Level3)

[1.2 运行环境](#_Toc18570_WPSOffice_Level3) [9](#_Toc18570_WPSOffice_Level3)

[1.3 基本设计概念和处理流程](#_Toc19250_WPSOffice_Level3) [9](#_Toc19250_WPSOffice_Level3)

[1.4 结构](#_Toc13118_WPSOffice_Level3) [10](#_Toc13118_WPSOffice_Level3)

[1.5 人工处理过程](#_Toc1085_WPSOffice_Level3) [10](#_Toc1085_WPSOffice_Level3)

[2、接口设计](#_Toc12258_WPSOffice_Level2) [11](#_Toc12258_WPSOffice_Level2)

[2.1 用户接口](#_Toc10825_WPSOffice_Level3) [11](#_Toc10825_WPSOffice_Level3)

[2.2 外部接口](#_Toc4026_WPSOffice_Level3) [11](#_Toc4026_WPSOffice_Level3)

[2.3 内部接口](#_Toc12439_WPSOffice_Level3) [11](#_Toc12439_WPSOffice_Level3)

[3、运行设计](#_Toc780_WPSOffice_Level2) [12](#_Toc780_WPSOffice_Level2)

[3.1 运行模块组合](#_Toc4767_WPSOffice_Level3) [12](#_Toc4767_WPSOffice_Level3)

[3.2 运行控制](#_Toc13850_WPSOffice_Level3) [13](#_Toc13850_WPSOffice_Level3)

[3.3 运行时间](#_Toc26786_WPSOffice_Level3) [13](#_Toc26786_WPSOffice_Level3)

[4、系统数据结构设计](#_Toc26275_WPSOffice_Level2) [14](#_Toc26275_WPSOffice_Level2)

[4.1 逻辑结构设计要点](#_Toc19729_WPSOffice_Level3) [14](#_Toc19729_WPSOffice_Level3)

[4.2 物理结构设计要点](#_Toc29405_WPSOffice_Level3) [16](#_Toc29405_WPSOffice_Level3)

[4.3 数据结构与程序的关系](#_Toc29347_WPSOffice_Level3) [18](#_Toc29347_WPSOffice_Level3)

[5、系统出错处理设计](#_Toc11150_WPSOffice_Level2) [18](#_Toc11150_WPSOffice_Level2)

[5.1 出错信息](#_Toc8327_WPSOffice_Level3) [18](#_Toc8327_WPSOffice_Level3)

[5.2 补救措施](#_Toc19749_WPSOffice_Level3) [19](#_Toc19749_WPSOffice_Level3)

[5.3 系统维护设计](#_Toc10904_WPSOffice_Level3) [19](#_Toc10904_WPSOffice_Level3)

[四、详细设计](#_Toc25048_WPSOffice_Level1) [20](#_Toc25048_WPSOffice_Level1)

[1、程序系统的结构](#_Toc14569_WPSOffice_Level2) [20](#_Toc14569_WPSOffice_Level2)

[1.1 创建用户模块设计说明](#_Toc5577_WPSOffice_Level3) [20](#_Toc5577_WPSOffice_Level3)

[1.2 推荐文章模块设计说明](#_Toc328_WPSOffice_Level3) [21](#_Toc328_WPSOffice_Level3)

[1.3 词汇测试模块设计说明](#_Toc20614_WPSOffice_Level3) [24](#_Toc20614_WPSOffice_Level3)

[1.4 获取新闻内容模块设计说明](#_Toc8906_WPSOffice_Level3) [27](#_Toc8906_WPSOffice_Level3)

[1.5收藏文章模块设计说明](#_Toc19574_WPSOffice_Level3) [29](#_Toc19574_WPSOffice_Level3)

[1.6 最近浏览模块设计说明](#_Toc3258_WPSOffice_Level3) [31](#_Toc3258_WPSOffice_Level3)

[1.7 背单词模块设计说明](#_Toc30098_WPSOffice_Level3) [34](#_Toc30098_WPSOffice_Level3)

[1.8 生词本模块说明分析](#_Toc16746_WPSOffice_Level3) [36](#_Toc16746_WPSOffice_Level3)

[添加生词业务逻辑](#_Toc30098_WPSOffice_Level2) [38](#_Toc30098_WPSOffice_Level2)

[五、测试计划](#_Toc30098_WPSOffice_Level1) [38](#_Toc30098_WPSOffice_Level1)

[六、测试结果分析](#_Toc16746_WPSOffice_Level1) [39](#_Toc16746_WPSOffice_Level1)

[七、 项目总结](#_Toc16343_WPSOffice_Level1) [40](#_Toc16343_WPSOffice_Level1)

[1、 实际开发结果](#_Toc16746_WPSOffice_Level2) [40](#_Toc16746_WPSOffice_Level2)

[1.1 软件产品描述](#_Toc16343_WPSOffice_Level3) [40](#_Toc16343_WPSOffice_Level3)

[2.2 主要功能与性能](#_Toc6947_WPSOffice_Level3) [41](#_Toc6947_WPSOffice_Level3)

[2.3进度](#_Toc20533_WPSOffice_Level3) [42](#_Toc20533_WPSOffice_Level3)

[2、 开发工作评价](#_Toc16343_WPSOffice_Level2) [43](#_Toc16343_WPSOffice_Level2)

[3.1 对项目开发过程的评价](#_Toc15370_WPSOffice_Level3) [43](#_Toc15370_WPSOffice_Level3)

[3.2 对技术方法的评价](#_Toc30620_WPSOffice_Level3) [44](#_Toc30620_WPSOffice_Level3)

[3.3 对产品质量的评价](#_Toc10018_WPSOffice_Level3) [44](#_Toc10018_WPSOffice_Level3)

[3、 试运行工作评价](#_Toc6947_WPSOffice_Level2) [45](#_Toc6947_WPSOffice_Level2)

[4、 技术积累总结](#_Toc20533_WPSOffice_Level2) [45](#_Toc20533_WPSOffice_Level2)

[5、 经验与教训](#_Toc15370_WPSOffice_Level2) [45](#_Toc15370_WPSOffice_Level2)

[5.1 邓晏湘](#_Toc5002_WPSOffice_Level3) [45](#_Toc5002_WPSOffice_Level3)

[5.2 陈彩婷](#_Toc14421_WPSOffice_Level3) [47](#_Toc14421_WPSOffice_Level3)

[5.3 汪伦](#_Toc15134_WPSOffice_Level3) [47](#_Toc15134_WPSOffice_Level3)

# 一、项目开发计划

## 1、引言

### 1.1 编写目的

为了保证项目的顺利进行，更好的规划团队的任务分配。 把对于在项目生命周期内的工作任务范围、各项工作的任务分解、项目团队组织结构、各团队成员的工作责任、团队内外沟通协作方式、开发进度、经费预算、项目内外环境条件、风险对策等内容做出的安排以书面的方式，作为项目团队成员以及项目干系人之间的共识与约定，项目生命周期内的所有项目活动的行动基础，项目团队开展和检查项目工作的依据。

### 1.2 项目背景

开发软件名称：《轻易在线阅读》

项目任务提出者：邓晏湘，陈彩婷，汪伦

项目开发者：邓晏湘，陈彩婷，汪伦

用户：喜欢英语新闻阅读，希望通过阅读提高英语水平的群体

### 1.3 定义

* 微信小程序：微信小程序，简称小程序，英文名Mini Program，是一种不需要下载安装即可使用的应用、它实现了应用“触手可及”的梦想，用户扫一扫或搜一下即可打开应用。
* django：Django是一个开放源代码的Web应用框架，由Python写成。采用了MVT的框架模式，即模型M，视图V和模版T。它最初是被开发来用于管理劳伦斯出版集团旗下的一些以新闻内容为主的网站的，即是CMS（内容管理系统）软件。并于2005年7月在BSD许可证下发布。
* WXML语言：微信小程序开发所使用的语言。WXML（WeiXin Markup Language）是框架设计的一套标签语言，结合基础组件，事件系统，可以构建出页面的结构。
* GitHub：GitHub是一个面向[开源及私有软件项目的托管平台，因为只支持git 作为唯一的版本库格式进行托管，故名GitHub。

### 1.4 参考资料

《软件工程导论》，张海藩，清华大学出版社。

《实用软件工程》，郑人杰等，清华大学出版社。

《软件工程导论》，张海藩（第五版），清华大学出版社；

[微信小程序开发文档](https://developers.weixin.qq.com/miniprogram/dev/framework)

[django开发文档](https://www.djangoproject.com/)

## 2、项目概述

### 2.1 工作内容

系统规划：根据对于问题的研究和调查，进行相关可行性研究，写出项目计划书和可行性研究报告

系统需求分析：分析目标和任务，构思基本业务流程，画出数据流图，ER图等，并编写数据字典

系统总体设计：画出系统结构图，找出系统模块，并设计数据库，编写概要设计说明书。

系统详细设计：画出基本逻辑结构图，代码设计，用户界面设计，数据输入与显示，控制界面的设计，系统安全控制设计，编写详细设计文档。

系统测试：系统实施与维护。

使用说明：针对普通用户群体，编写系统使用手册，安装手册，以及编码清单。

项目报告：小组成员对整个开发过程进行分析总结，并编写项目总结报告。

### 2.2 主要参加人员

前端/UI设计

* 陈彩婷：擅长UI设计以及vue框架的使用和开发，综合能力较强，能够编写鲁棒性强，交互性好的前端界面，熟悉各种资源的使用。
* 汪伦：自主开发独立性强，创造性思维强，能够在前人的基础上构思新的方法，不仅熟知前端开发同时精通后端的数据处理。

后端

* 邓晏湘：能力一般，比较擅长面向搜索引擎的代码理解和移植。

### 2.3 产品

#### 2.3.1 程序

系统名称：轻易在线阅读

编程语言：Python，Javascript

功能

* 个性化推荐最适合用户阅读的文章
* 测试词汇量
* 阅读最新的英文时事要闻
* 背单词

#### 2.3.2 文件

应交付的文档：可行性研究文档，项目开发计划， 需求分析，系统设计，测试计划，结果分析，项目总结，可独立运行的程序，使用手册

### 2.4 验收标准

文件：说明具体详尽完整，能够体现小组的工作量。格式满足国家标准文件规范，易读性强，并且结合项目实际情况，无夸大。

程序：鲁棒性强，实现了要求的功能，用户体验感好。

### 2.5 完成项目最迟日期

2019年6月17日（周日）之前

### 2.6 本计划的批准者和批准日期

软件工程课程设计授课老师：王鹏伟

## 3、实施计划

### 3.1 工作任务的分解和人员分工

项目开发计划 负责人：邓晏湘

需求分析 负责人：陈彩婷

（系统）概要设计 负责人：邓晏湘

（系统）详细设计 负责人：邓晏湘

测试计划 负责人：陈彩婷，汪伦

测试结果分析 负责人：陈彩婷，汪伦

项目总结 负责人：邓晏湘，陈彩婷，汪伦

使用手册：汪伦

安装手册：汪伦

编码清单：邓晏湘

### 3.3 进度

开发进度规划以及每个阶段需要完成的任务由甘特图给出。

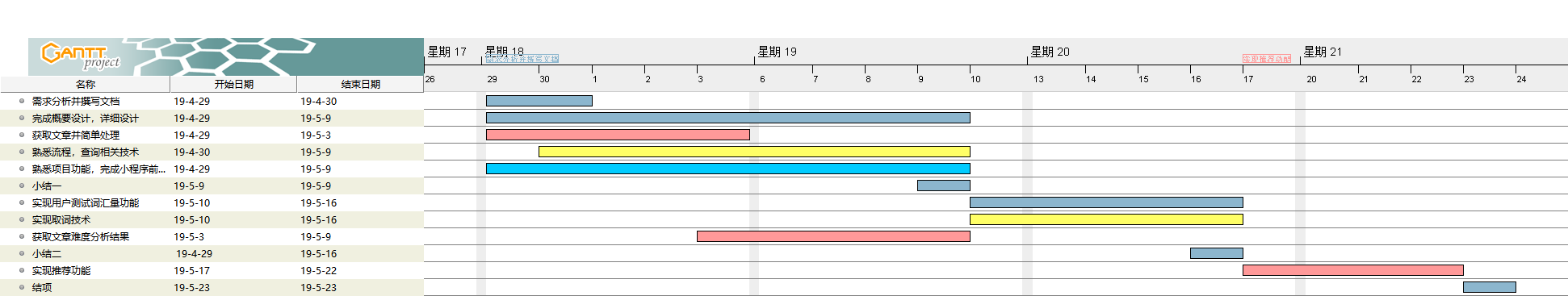


表 1开发进度规划（甘特图）

### 3.4 关键问题

* 测试词汇量：测试词汇量的可靠程度直接影响推荐的文章能否满足用户需求，提升用户的阅读体验。
* 根据词汇量个性化推荐文章：个性化的定制程度直接影响用户是否愿意一直使用该软件。
* 爬取各大外国新闻网站文章：有些国外网站不允许爬取文章以及爬取文章每天更新背后的逻辑会对系统的稳健性造成一定的影响。
* 点击取词翻译：如何识别词汇，并且显示词汇含义对于实现添加生词，背单词都起到了决定性的作用。如果不能实现直接影响了后续功能。

## 4、专题计划要点

测试计划：后端边编写边自测，前端编写完并且和后端的接口联系起来后再统一进行测试。

质量保证计划：在开发的过程中，小组成员随时保持交流，并且对彼此的代码进行了解，提出疑问，相互监督和检查。

配置管理计划：利用现有设备对整个系统进行开发。

系统安装计划：在每个小组成员的机器上进行安装调试，保证设备都能正常工作。

# 二、需求分析

# 三、概要设计

## 1、总体设计

### 1.1 需求规定

功能性要求：构建一个完整的阅读小程序，实现阅读，推荐，背单词，查 询单词，收藏文章等一系列功能。

输入项目为用户的一系列操作请求

输出项目为针对不同请求的响应。

### 1.2 运行环境

* 安卓4.5及以上
* ios设备11.0及以上

### 1.3 基本设计概念和处理流程

系统整体架构如下。用户在微信小程序界面进行操作，前端处理用户各种操作请求，针对需要涉及到数据交互以及复杂操作的命令，则交由后端处理。后端对于前端的http请求，首先从映射的url列表中一一匹配，找到合适的则进行处理并返回处理结果。

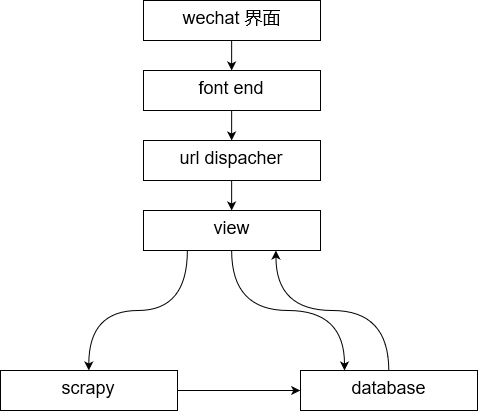


表 2系统架构

### 1.4 结构

小程序用户角色只有阅读者，无管理员等角色。功能主要分为四大模板，用户登陆小程序，创建身份，并获取唯一标识。阅读文章功能包含四个子系统，分别是浏览所有文章模板，查看最近浏览文章，收藏某文章以及获取个性化推荐的文章。测试词汇量下属两个子模板，分别是获取测试的词汇数据，以及查看历史测试结果。用户可以通过生词本背单词，并且对单词进行删除添加等操作。

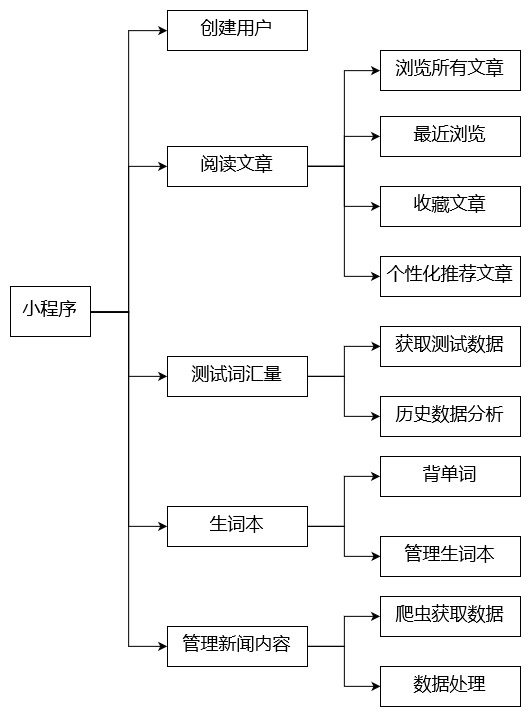


表 3系统功能点

### 1.5 人工处理过程

从新闻网站爬取数据以及将数据移植到服务器的数据库中需要手动操作。

## 2、接口设计

### 2.1 用户接口

用户接口主要功能为用户对文章，生词进行一系列相关处理，并且对个人的生词数据进行操作。

| **接口** | **功能** |
| --- | --- |
| 浏览推荐文章 | 根据用户词汇测试数据返回推荐的文章 |
| 登陆 | 获取用户登陆状态以及唯一标识 |
| 按分类浏览文章 | 获取某分类所有文章 |
| 管理生词本 | 删除/添加单词 |
| 背单词 | 获取生词本所有单词开始记忆 |
| 测试词汇量 | 获取待测试词汇，处理测试结果 |
| 收藏文章 | 收藏文章信息，可供再次查看 |
| 最近浏览 | 查看最近阅读的文章 |

表 4用户接口

### 2.2 外部接口

外部接口主要用于与数据库进行信息交互。

| **接口类型** | **接口** | **功能** |
| --- | --- | --- |
| 软件接口 | 与数据库接口 | 连接数据库并进行信息交互 |
| 服务器接口 | 微信服务器接口 | 从微信服务器解码code，获取用户openid |

表 5外部接口

### 2.3 内部接口

内部接口主要是系统内部功能，例如利用爬虫获取新闻资源，对文章内容进行一定的处理，不同模块之间的调用。

| **接口** | **功能** |
| --- | --- |
| 添加测试记录 | 获取用户测试词汇结果并存入数据库 |
| 查询测试记录 | 返回用户历史测试记录 |
| 添加单词 | 向服务器生词本添加单词 |
| 删除单词 | 删除生词本中指定单词 |
| 查询单词 | 返回所有生词 |
| 添加收藏记录 | 将收藏文章的信息录入数据库 |
| 生成测试词汇 | 随机生成用户测试词汇数据 |
| 录入文章信息 | 将服务器本地文章信息录入数据库 |
| 爬取新闻资源 | 调用scrapy框架获取新闻资源 |

表 6内部接口

## 3、运行设计

该部分主要描述针对系统施加不同外界运行控制时所引起的各种不同的运行模块组合。

### 3.1 运行模块组合

#### 3.1.1 阅读文章模块组合

* 用户浏览所有文章，点击查看文章，添加生词，添加收藏功能模块组合。
* 用户测试词汇，查看个性化推荐文章功能模块组合。
* 用户查看单词本，删除单词，背单词功能模块组合。
* 用户查看最近浏览文章，点击阅读，添加收藏功能模块组合
* 用户背单词，选择单词释义，查看测试结果功能模块组合
* 用户点击分类，阅读文章功能模块组合。

#### 3.1.2 处理新闻资源模块组合

* 后台运行爬虫程序，获取新闻资源，录入数据库功能模块组合
* 后台录入新闻资源，进行数据整理，查询数据功能模块组合
* 后台根据用户信息，获取数据库资源，进行数据比较，返回文章信息功能模块组合
* 后台根据文章图片地址，修改图片名，下载图片并修改数据库功能模块组合

#### 3.2 运行控制

#### 3.2.1 阅读文章模块组合

* 用户需要授权小程序获取用户id才能阅读文章
* 用户需要至少测试过一次词汇量才可以获取个性化推荐文章

### 3.3 运行时间

#### 3.3.1 阅读模块运行时间

系统主要的运行时间消耗在对数据库进行增删改查的操作以及前后端交互过程中大量数据传输的时间。当后台文章未及时清理，导致文章条目不断变多，会致使用户浏览全部文章时，后台一次性返回的数据量过多，加载缓慢。

#### 3.3.2 处理新闻资源运行时间

主要时间消耗在获取文章资源，由于外文网站服务器在国外，相对时延较大，并且在爬取的过程中，框架本身的特性也会造成一定的时间消耗。所以时间的开销不能忽略。

但是好在文章爬取一天仅做一次，并且并非实时获取。所以对于用户体验影响不大。

#### 3.3.3 耗时少的功能模块：

直接在前端实现的所有功能都耗时较少。例如，对用户测试词汇的数据进行分析，按照分类展示文章等等。

#### 3.3.4 耗时大的功能模块：

与后端进行大量数据传输的功能模块，以及与数据库进行交互的模块。

## 4、系统数据结构设计

### 4.1 逻辑结构设计要点

* 系统涉及到的实体主要有如下几种，文章，用户，词汇测试记录，收藏文章信息。

用户（编号，小程序唯一标识，生词本）

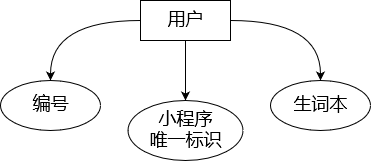
* 

表 7用户逻辑设计图

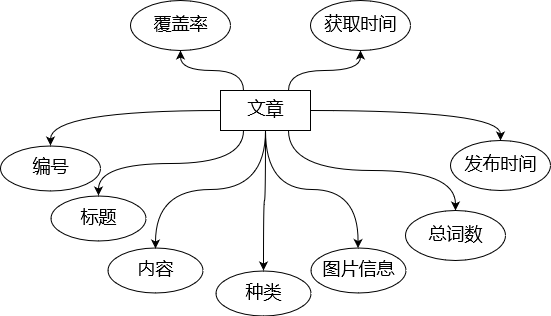
* 文章（编号、标题、内容、作者、种类、图片信息、总词数、发布时间、覆盖率、获取时间）
* 

表 8文章逻辑设计

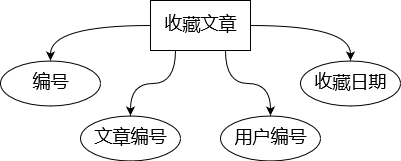
* 收藏文章（编号、文章编号、用户编号、收藏日期）
* 

表 9收藏文章逻辑设计

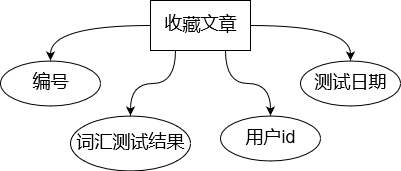
* 词汇测试记录（编号、词汇测试结果、用户id、测试日期）
* 

表 10测试结果逻辑设计

* ER图
* 

表 11 ER图

### 4.2 物理结构设计要点

用户编号是用户唯一标识加密过后的结果，为了保证传输过程的安全性，在一般传输过程中都使用用户编号进行操作。生词本记录了用户生词本在服务器中的存储位置。

| **字段名** | **类型** | **宽度** | **主键/索引** | **约束** | **是否可为null** |
| --- | --- | --- | --- | --- | --- |
| 用户编号 | varchar | 500 | 主键 |  | 否 |
| 用户唯一标识 | varchar | 500 |  |  | 否 |
| 生词本 | varchar | 500 |  |  | 否 |

表 12 用户物理结构设计

文章通过文章编号唯一标识，该属性是自增的。考虑到文章内容有长有短，内容过长在数据库会占据较大空间，因此文章内容存储的是文章存储在服务器上的位置，是一个json文件。图片信息也是如此。只有一点特别的就是图片存储在服务器文件的指定文件夹下，可以通过url直接访问。

| **字段名** | **类型** | **宽度** | **主键/索引** | **约束** | **是否可为null** |
| --- | --- | --- | --- | --- | --- |
| 文章编号 | integer |  | 主键 |  | 否 |
| 标题 | varchar | 500 |  |  | 否 |
| 文章内容 | varchar | 500 |  |  | 否 |
| 作者 | varchar | 500 |  |  | 否 |
| 类型 | varchar | 500 |  |  | 否 |
| 图片信息 | varchar | 500 |  |  | 否 |
| 总词数 | integer |  |  |  | 否 |
| 出版时间 | varchar | 500 |  |  | 否 |
| 覆盖率 | varchar | 500 |  |  | 否 |
| 获取时间 | date |  |  |  | 否 |

表 13 文章表物理设计

测试内容由编号唯一标识，该属性自增。外键为用户编号。一个用户可以有零到多条测试记录。

| **字段名** | **类型** | **宽度** | **主键/索引** | **约束** | **是否可为null** |
| --- | --- | --- | --- | --- | --- |
| 测试编号 | integer |  | 主键 |  | 否 |
| 覆盖率 | varchar | 500 |  |  | 否 |
| 用户编号 | varchar | 500 |  |  | 否 |
| 日期 | date |  |  |  | 否 |

表 14 测试结果物理设计

收藏条目由编号唯一标识，属性自增。外键为文章编号与用户编号。

| **字段名** | **类型** | **宽度** | **主键/索引** | **约束** | **是否可为null** |
| --- | --- | --- | --- | --- | --- |
| 收藏编号 | integer |  | 主键 |  | 否 |
| 文章编号 | integer |  |  |  | 否 |
| 用户编号 | varchar | 500 |  |  | 否 |
| 日期 | date |  |  |  | 否 |

表 15 收藏条目物理设计

### 4.3 数据结构与程序的关系

后端在获取用户测试结果，记录用户操作信息时需要对数据库中的表进行修改，即增删改查操作。

物理数据结构主要用于各模块之间函数的信息传递。接口传递的信息是数据结构封装了的数据，以参数或者返回值的形式在各模块之间传输。

## 5、系统出错处理设计

### 5.1 出错信息

| **出错类型** | **出错形式** | **处理办法** |
| --- | --- | --- |
| 404 | 从后端获取数据，显示内容不存在 | 检查发送请求的url地址以及http请求的格式，逐项排查。 |
| 500 | 从后端获取数据，显示服务器内部错误 | 检查服务器部分代码，重点查看负责该部分功能模块代码的业务逻辑以及返回数据的格式。 |
| 登陆 | 无法获取小程序用户的唯一标识 | 排查后端处理code部分关于小程序的开发者信息是否正确以及检查用户是否过长时间未登陆导致信息失效。 |
| 无法获取文章信息 | 用户查看收藏文章，无法正确显示 | 查看后端是否对数据库进行了误操作，删除了文章内容和记录。 |
| 无法爬取文章资源 | 启动爬虫程序，无法获取任何文章信息 | 可能是访问过于频繁被网站禁止访问了，可以通过更换ip池的ip来访问。 |

表 16 出错信息

### 5.2 补救措施

通过对以上的错误信息进行分析，我们可以针对系统进行一些优化：

1、数据库数据交互错误：这类错误主要是因为开发人员对数据库进行了误操作或者是功能模块内部对数据进行了操作但是开发人员没有意识到，对此， 我们应该定期对数据库进行备份，对进行了数据库相关操作的模块进行严谨的测试。

2、请求错误：主要是前后端的开发人员之间对于接口的定义以及数据形式没有协商好，需要细化接口设计。

3、其他操作错误：针对用户因为不熟悉系统而造成的错误，我们可以通过撰写详尽的使用手册来规避。

4、其他不可知的程序错误：比如，原本网站突然不服务了，或者被禁掉了。这种不可预知的错误可以通过顶起维护后端来规避，或者在程序中设置提示，在发生异常的时候，设置自动消息提示。

### 5.3 系统维护设计

随着文章爬取数量的不断增加，数据库势必会越来越庞大，在爬取的过程中，可能会产生一些无效文章信息，以及脏数据。人工检查成本比较高，因此采取后端维护人员定期对数据进行维护备份的方案。

前端由开发人员不断根据技术更新进行升级完善。

# 四、详细设计

## **1、程序系统的结构**

针对概要设计部分的说明，针对每个模块进一步细化，明确内部结构和实现细节。方便后期测试时进行调试和复查。

### **1.1 创建用户模块设计说明**

#### **1.1.1 程序描述**

用户初次登陆小程序需要获取唯一标识以记录登陆状态，以及在数据库中生成条目用以记录用户后续在小程序中一系列操作。

#### **1.1.2 功能**

由于微信对小程序开发的安全性有较高要求，因此用户信息的解码无法在前端直接进行，只能在前端获取code，然后在后端，附带开发者的基本信息调用微信的api接口来获取解码信息。

用户的基本标识不能直接明文传输，因此加密以后作为数据库用户表的主键并传回前端。此后涉及用户的数据操作，前端都需要附带用户的唯一标识。

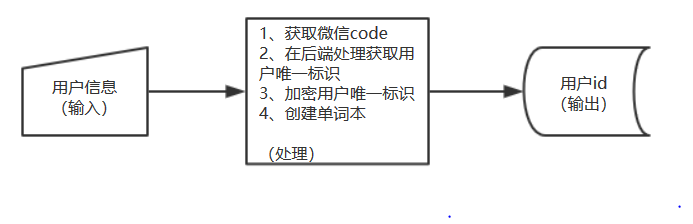


表 17创建用户IPO图

#### **1.1.3 性能**

时间：由于用户登陆授权后，需要及时得到唯一标识并允许用户进行后续操作，因此对于服务器获取解码信息的响应要求较高。

#### **1.1.4 输入项**

| **名称** | **标识** | **类型** | **输入方式** |
| --- | --- | --- | --- |
| 用户编码 | code | json | 附在http请求中 |

#### **1.1.5 输出项**

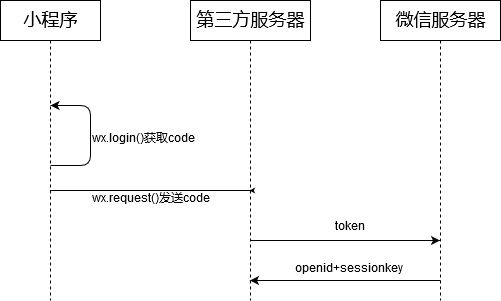
| **名称** | **标识** | **类型** | **输出方式** |
| --- | --- | --- | --- |
| **用户id** | **uid** | **string** | **附在httpResponse中** |

#### **1.1.6 算法**

使用了python自带的md5加密算法对所有用户的唯一标识进行加密。

#### **1.1.7 流程逻辑**

整个获取流程如下图所示。



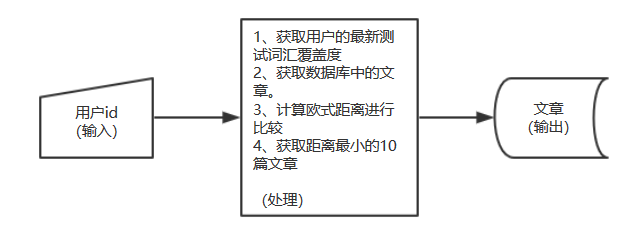
### **1.2 推荐文章模块设计说明**

#### **1.2.1 程序描述**

该模块主要实现了针对用户的词汇量以及对不同词库（四六级等）词汇掌握情况推荐最适合阅读的文章的功能。

#### **1.2.2 功能**

用户获取推荐文章的前提是已经做过词汇测试，如果做过多轮词汇测试则取最新的结果。而文章根据后台管理者的需求，可以设置只取近3天（或者任意）的文章进行推荐。



#### **1.2.3 性能**

时间特性：用户做完词汇测试后应当立即就能看到推荐文章，等待时间过长会影响用户体验，因此要求后端进行数据处理时，不能消耗过长时间，如果数据库中的文章条目过多，后台管理员应该设计一个取数据范围。

#### **1.2.4 输入项**

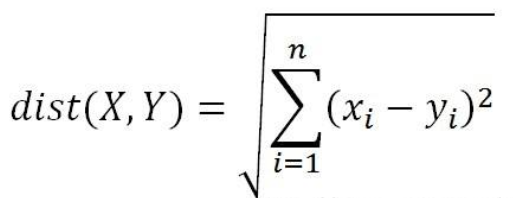
| **名称** | **标识** | **类型** | **输入方式** |
| --- | --- | --- | --- |
| 用户id | uid | string | 附在http请求中 |

#### **1.2.5 输出项**

| **名称** | **标识** | **类型** | **输出方式** |
| --- | --- | --- | --- |
| 推荐文章 | articles | dict | httpresponse中 |

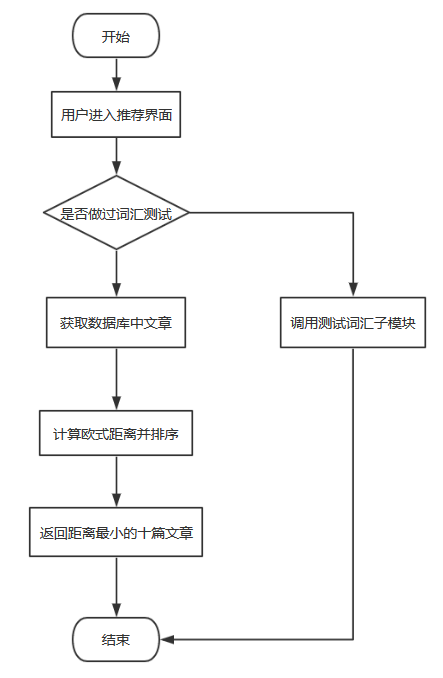
#### **1.2.6 算法**

主要用到了欧式距离来计算用户词汇覆盖度和文章词汇覆盖度之间近似程度。



#### **1.2.7 流程逻辑**

当用户进入推荐模块才会判断是否做过词汇测试，若用户刚进入小程序，并且没有用到推荐功能，则不会要求进行词汇测试。



#### **1.2.8 接口**

若用户未做过词汇测试并想使用推荐功能，则调用词汇测试模块。

### **1.3 词汇测试模块设计说明**

#### **1.3.1 程序描述**

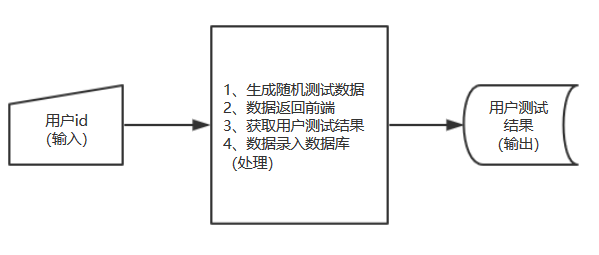
不同的用户掌握的词汇不尽相同，有的用户可能掌握的四级词汇较多，有的可能掌握的六级词汇较多， 该模块主要用于对用户的词汇覆盖程度做一定的度量（度量的标准有四级，六级，托福，雅思等），得出的结果是系统个性化推荐文章的重要依据。

#### **1.3.2 功能**

首先获取用户id，因为测试的结果需要和用户绑定。用户使用该功能后，后端在每个词库中分别随机化抽取词汇，返回前端。

用户对每个单词进行判断，如果认识则选择“认识”，不认识则选择“不认识”，最后将认识的词汇数/该词库抽取的总词数，该比例可以指代用户对于该词库的掌握程度。该数据由前端统计后返回后端并存入数据库。

同时，用户在测试完以后可以查看不认识的词汇进行复习巩固。



#### **1.3.3 性能**

精度：用户词汇掌握程度的结果取浮点数，小数点后两位即可。

灵活性：由于随机生成的测试词汇比较多，因此，应当允许用户随时停止测试，并对未测试的数 据做出随机化处理。

#### **1.3.4 输入项**

| **名称** | **标识** | **数据类型** | **输入方式** |
| --- | --- | --- | --- |
| 用户id | uid | string | http请求 |

#### **1.3.5 输出项**

| **名称** | **标识** | **数据类型** | **输出方式** |
| --- | --- | --- | --- |
| 词汇覆盖度 | cover\_rate\_list | list | http请求 |
| 测试词汇 | test\_words | dict | httpResponse |

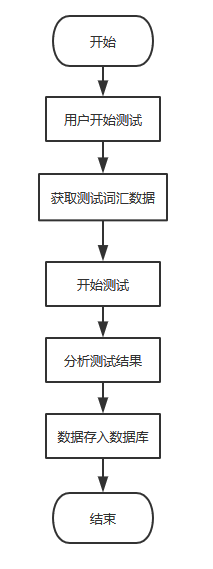
#### **1.3.6 算法**

获取的词库以及词库词汇数量如下：

| **词库** | **cet4** | **cet6** | **高中** | **GRE** | **IELTS** | **考研** | **托福** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 词数 | 4616 | 2088 | 3933 | 6495 | 6555 | 5325 | 5395 |

按照一定的比例从每个词库中随机抽取单词（该比例可有后台开发者自定），若从六级词库中抽取10个词，用户进行测试后，认识6个词，那么可以估计用户对于四级词汇的掌握程度为0.6。其他的词库掌握情况以此类推。

#### **1.3.7 流程逻辑**



#### **1.3.8尚未解决的问题**

有的词库中抽取的词汇可能会和其他词库重合（但是概览较低），因此最保险的做法是在随机抽取词汇时进行判定，若抽到了重复的数据则重新抽取。

### **1.4 获取新闻内容模块设计说明**

#### **1.4.1程序描述**

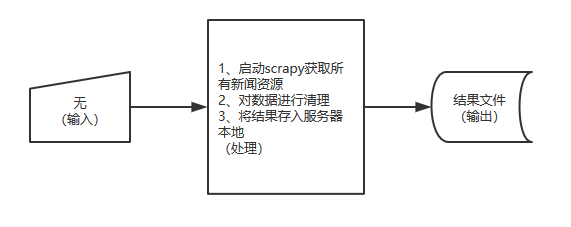
系统最重要的部分是对于数据的获取，我们使用scrapy框架来获取新闻内容以及其他相关信息。scrapy框架功能较为完整，只需要明确需要获取的页面元素即可自动化的爬取内容。

调用该程序后，模块会自动爬取当日所有条目下的新闻文章，并对数据做一定的处理，最后形成json文件存入本地。

工作人员手工将获取的新闻数据再存入数据库。

#### **1.4.2功能**

获取了所有新闻数据后，需要清洗数据，将文本内容从string格式变成以单独词为单位的list列表，并且获得去重的文章单词信息，再将去重单词列表与词库相对比，查看文章内容针对每个词库的覆盖情况。



#### **1.4.3性能**

时间特性：由于数据每日获取一次，对于获取的时间没有要求

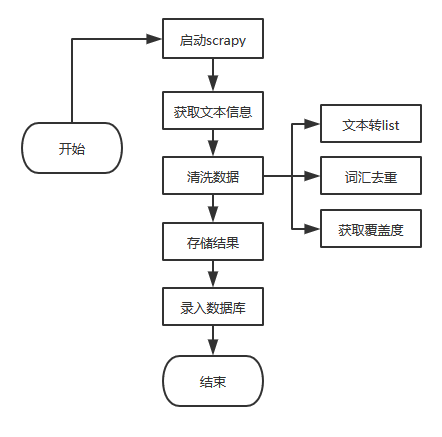
灵活性：模块应当能够处理一定的错误，例如在请求过于频繁，ip地址被禁止以后能够切换ip。

精度：精度要求不高，但是要确保文章内容是准确的。

#### **1.4.4 输出项**

| **名称** | **标识** | **数据类型** | **输出形式** |
| --- | --- | --- | --- |
| 文章信息汇总 | news | dict | 本地文件 |
| 文章内容 | content | list | 本地文件 |

#### **1.4.5 流程逻辑**

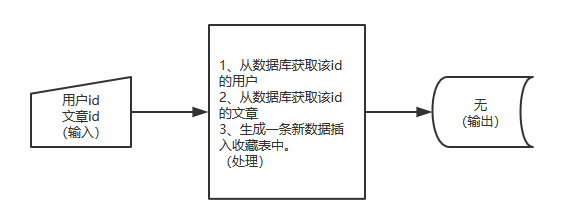


### **1.5收藏文章模块设计说明**

#### **1.5.1程序描述**

该模块主要实现了用户收藏文章的功能，用户与文章是一对多的关系，一个用户可以收藏多篇文章，并且可以在”我的收藏“模块查看历史收藏。

#### **1.5.2功能**



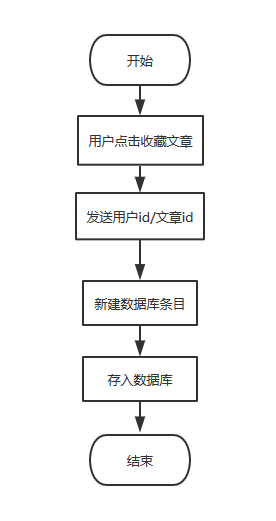
#### **1.5.3性能**

精度：需要准确的记录用户收藏的文章信息以及收藏的日期。

#### **1.5.4输入项**

| **名称** | **标识** | **类型** | **输入方式** |
| --- | --- | --- | --- |
| 文章id | article\_id | integer | http请求 |
| 用户id | user\_id | string | http请求 |

#### **1.5.5流程逻辑**



### **1.6 最近浏览模块设计说明**

#### **1.6.1 程序描述**

该模块主要实现存储用户最近阅读的10篇文章记录的功能。

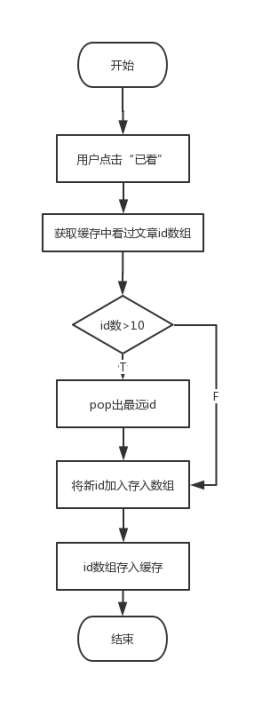
#### **1.6.2 功能**

由于最近浏览不存储过多数据，将用户每篇阅读完成的文章主键存储在数组中，再将该数组同步保存到小程序缓存中：wx.setStorageSync('seenList', temp)；查看浏览记录，则直接从缓存中同步获取wx.getStorageSync('seenList')

#### **1.6.3 输出项**

| **名称** | **标识** | **数据类型** | **输出方式** |
| --- | --- | --- | --- |
| 最近阅读文章列表 | article\_list | list | 从缓存中取 |

#### **1.6.4 流程逻辑**

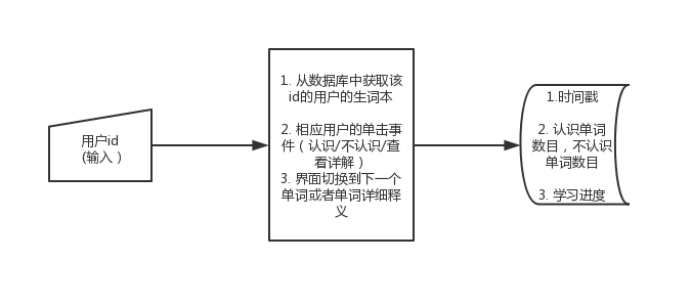


### **1.7 背单词模块设计说明**

#### **1.7.1 程序描述**

该模块主要实现了用户背单词的功能，每一个用户拥有的生词本存放在后台的服务器中。背单词的过程中，程序会随机以卡片的形式展示生词本中的莫一个单词，用户对当前卡片所展示的单词有两个选项，认识和不认识。如果用户不认识该单词可以单击单词卡片上的“查看详细释义”，之后便可以看到单词的详细解释，包括双语例句，发音，以及词性。不认识的单词会多出现几遍。

#### **1.7.2 功能**



#### **1.7.3 输入项**

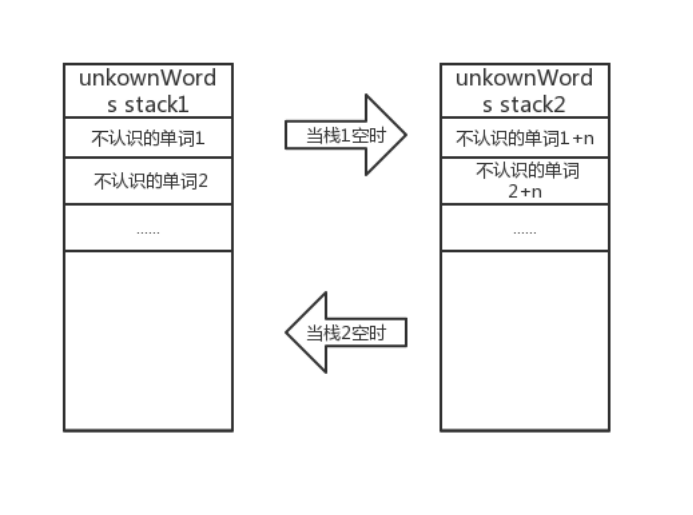
| **名称** | **标识** | **数据类型** | **输入方式** |
| --- | --- | --- | --- |
| 用户id | uid | string | http请求 |
| 认识 | Known | event | 用户触发认识按钮 |
| 不认识 | Unkown | event | 用户触发不认识按钮 |

#### **1.7.4 输出项**

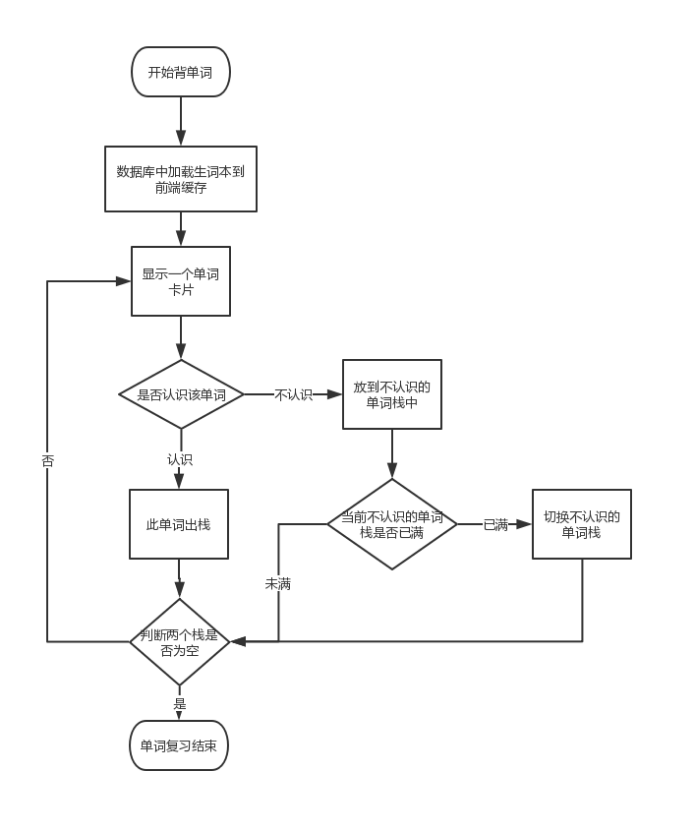
| **名称** | **标识** | **数据类型** | **输出方式** |
| --- | --- | --- | --- |
| 认识单词数 | familiarWords | int | 小程序缓存 |
| 不认识单词数 | UnfamiliarWords | int | 小程序缓存 |
| 背单词日期 | currentTime | Date | 小程序缓存 |

#### **1.7.5 算法**

背单词的过程中，每次用户单击不认识此单词的时候就将此不认识的单词放到一个栈中，当当前的栈空的时候开始背另一个栈，两个栈来回倒来实现背单词的功能，直到用户将单词背完。



#### **1.7.6 流程逻辑**

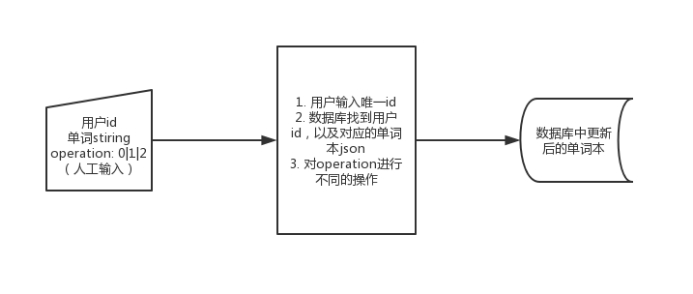


### 1.8生词本模块说明分析

#### 1.8.1程序描述

用户在阅读文章的过程中，可以添加单词到单词本中，单词本中的每一个单词在数据库中的存放形式是一个json的列表，列表里的每一个元素都是一个字典。字典数据的健是单词，值是被添加的次数。用户可以对单词本进行，增和删两种操作。

#### 1.8.2功能



#### 1.8.3输入项

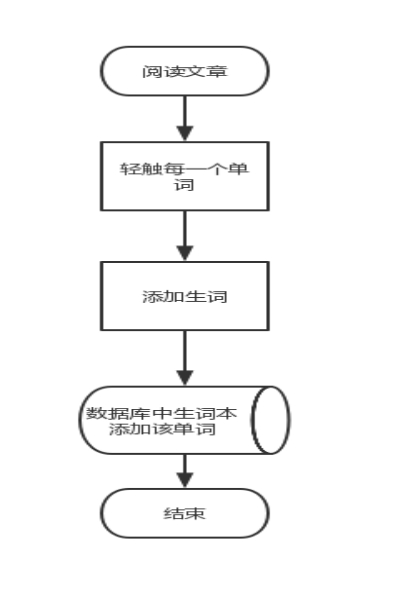
| **名称** | **标识符** | **数据类型** | **输入方式** |
| --- | --- | --- | --- |
| 0 | operation | int | http |
| 1 | operation | int | http |
| 2 | operation | int | http |

#### 1.8.4 输出项

| **名称** | **标识符** | **数据类型** | **输出方式** |
| --- | --- | --- | --- |
| 0 | operation | list | http |

#### 1.8.5 **逻辑流程**

添加生词业务逻辑



# 五、测试计划

# 六、测试结果分析

# 项目总结

## 实际开发结果

### 1.1 软件产品描述

软件模块清单：

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 子系统名称 | 模块名称 | 模块标识 |
| 1 | 创建用户 | 创建用户 | CreateUser |
| 2 | 阅读文章 | 个性化推荐文章 | getArticle |
| 按分类查看文章 | getArticlebyType |
| 收藏文章 | collectArticle |
| 查看最近浏览 | history |
| 查词 | getWord |
| 3 | 生词本 | 添加生词 | AddWord |
| 删除生词 | DeleteWord |
| 背单词 | Recite |
| 4 | 新闻内容 | 获取新闻数据 | GetNews |
| 数据清洗 | clearData |
| 5 | 词汇测试 | 获取测试词汇 | getTestWords |
| 查看历史测试 | showTestHistory |

文档清单：

|  |  |  |
| --- | --- | --- |
| 序号 | 文档名称 | 需提交用户 |
| 1 | 项目开发计划 |  |
| 2 | 需求分析 |  |
| 3 | 概要设计 |  |
| 4 | 详细设计 |  |
| 5 | 测试分析 |  |
| 6 | 测试结果 |  |
| 7 | 项目开发总结报告 |  |
| 8 | 编码清单 |  |
| 9 | 安装手册 | 是 |
| 10 | 使用手册 | 是 |

### 2.2 主要功能与性能

|  |  |  |  |
| --- | --- | --- | --- |
| 功能点 | 功能 | 性能 | 开发目标 |
| 个性化推荐文章 | 根据用户的词汇量推荐适合阅读的文章 | 即时、准确 | 达到了 |
| 测试词汇 | 给予用户词汇测试并得出结果 | 具有一定的可靠性 | 达到了 |
| 阅读最新英文新闻 | 基于用户每日最新英语新闻，并提供查词，按分类查看，收藏文章，浏览最近阅读文章的功能 | 内容准确，查词翻译响应准确即时 | 超过了 |
| 背单词 | 针对添加的生词提供背诵功能，已经背熟的单词可以删除 | 内容无误 | 达到了 |
| 获取文章 | 获取外文网站每日最新文章，进行数据清洗和统计 | 即时，准确，广泛 | 未完全达到 |

### 2.3进度

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 阶段 | 计划工作日 | 实际工作日 | 对比 | 原因 |
| 立项 | 2-3 | 7 | 延迟了4天 | 团队成员意见不一致 |
| 开发计划 | 2 | 2 | 按时完成 |  |
| 需求分析 | 2 | 3 | 延迟一天 | 分析之后进行了多次调整 |
| 概要设计 | 1 | 1 | 按时完成 |  |
| 详细设计 | 5 | 5 | 按时完成 |  |
| 编码 | 30 | 35 | 延迟5天 | 前后端工作人员进行接口对接多花了时间 |
| 测试计划 | 2 | 2 | 按时完成 |  |
| 测试结果分析 | 5 | 5 | 按时完成 |  |
| 项目开发总结报告 | 2 | 5 | 按时完成 | 成员共同完成，进度不一致 |
| 其他文档 | 2 | 2 | 按时完成 |  |

## 开发工作评价

### 3.1 对项目开发过程的评价

整个开发过程按照最开始的甘特图，分成了三个阶段，因此，针对每个阶段工作的完成情况以及文档编写情况和最初计划做对比，得出如下的总结。

阶段一：

* 活动：立项，进行可行性和需求分析，撰写需求分析报告。明确项目的功能点。并进行相关技术和知识的学习和搜集。
* 完成情况：良好
* 文档编写情况：初期文档编写比较简略，有诸多没有敲定的细节，文档成型比较简陋
* 出现的问题：功能点不明确，在确定项目所需技术方面不明确，以及需求分析涵盖不全
* 原因：小组成员经验不足，交流不够，导致想法和问题没有及时反馈。

阶段二：

* 活动：进行概要设计，详细设计以及编码。
* 完成情况：良好
* 文档编写情况：文档中的接口设计不明确，导致前后端编码实现出现偏差以及数据库设计有问题，导致多次返工。
* 出现的问题：闭门造车，导致协同工作的成员实现的功能有部分地方重复，文档由成员完成后未经过其他成员的检查和讨论细节，导致后期出现问题，数据库全部推翻重来。
* 原因：功能分配前后端过于明确，导致成员之间无联系和交流，后期需要将整个系统的前后端联系起来时，出现了较大问题，浪费不少时间。无论各自负责的工作是什么，都要及时了解其他成员的工作以及进度，给彼此提出意见。有问题。勿要耽搁，及时反馈。

阶段三：

* 活动：进行测试调试和项目总结工作。
* 完成情况：良好
* 文档编写情况：进行了测试计划和分析，项目总结报告以及其他附属文档的编写。条例比较清晰，文档编写较为规范。
* 出现的问题：测试上针对测试用例的编写出现了一点分歧，以及测试具体执行上有困惑。其他的问题有的没有了解过，导致动手困难。
* 原因：过去都是对cs结构的系统或者java web端的系统进行测试，这是第一次对小程序进行测试，对于方法和工具都不熟悉。

### 3.2 对技术方法的评价

该系统的开发主要使用到了python django框架用于搭建后端服务器，scrapy框架用于获取网络新闻资源，以及JavaScript语言和微信开发的wxml语言进行前端开发，同时使用github进行协同办公。同时运用了多线程，缓存，欧拉距离，栈等设计方式来实现基础功能。

工具用到了vscode文本编辑器，微信开发组件等。采取敏捷开发流程，不断在开发的过程中进行小结，回顾和迭代。

总的来说，使用的都是比较新的技术，结合最适合的方法用简单的设计实现了多数功能。

在技术方法的选择上比较符合实际要求，虽然没有使用难度较大最时兴的技术，但是精简有用。

### 3.3 对产品质量的评价

开发技术和框架的选择较新，开发人员的素质良好，代码实现和沟通都非常流畅和默契。时间进度虽然中间偶有延迟，但是总体上是保持正常的节奏，并且按时完成了。软件的质量中上，不可避免还会有小的缺陷，但是整体功能是完备的。

## 试运行工作评价

在产品试运行的过程中暴露出了一定的接口设计对接，业务流程逻辑的问题，但是最后都调试修正了。基于该系统自身的性质，产品的维护工作量较小，后期只需要将人工辅助实现的数据录入部分变成自动化流程即可达到阶段性维护的要求。

## 技术积累总结

所使用的新技术和框架知识都是基于官方文档以及开源的项目。大部分技术说明都有编写完备的文档，在这里不再赘述。

## 经验与教训

经验教训因人而异，这里分成三部分，分别是邓晏湘，陈彩婷，汪伦独自撰写的心得体会。

### 5.1 邓晏湘

写在前面，首先感谢老师这一学期的指导，感谢助教的协助，也感谢我的组员汪伦，陈彩婷的付出。虽然最后做出来的小程序还有很多不足的地方，但是基本实现了所有的功能点，也能够在大多数情况下正常运行。这一切都离不开我的组员的努力。

老师在课上说，一个组无论任务如何分配，最后肯定会有人做的多有人做的少，这是事实。但是，回顾整个敏捷开发过程，我觉得，如果组内的成员都有责任心，并且配合默契，自觉承担责任较少的人会相应的去帮助任务重的同学，这也是事实。

这是我们小组第二次在课程设计中合作（前一次是软测课设），我作为组长负责后端而他们二人负责前端，一开始我们对任务分配都没有异议，但是开发到后期，我自己认为后端的任务相较于前端会轻松一些（因为他们是第一次使用微信的wxml语言开发小程序，我原来就会一些python，所以学起django和scrapy来没那么难），所以选择多写一些文档，或者是努力完善后端接口设计，尽可能多做一些事，而他们二人也会对彼此的模块进行相应了解，在其中彼此调试艰难的时候，施以援手。

作为组长，我体会到了很多原来独立完成大作业无法了解的东西。分门别类的梳理下来，有如下几点：

学会倾听和讲道理。一开始我们在选题的问题上有不小的争执，汪伦执意想做java web项目，他的理由非常有道理，现在找工作java岗位比较火。但是我认为应该在课设中去学习新东西，应该做一些真正可以用得上的东西。我们第一节课一直在试图说服对方，最后失败。我脾气火爆，一直认为作为组长就应该掌握主动权，定题应该以我为准，听不进别人的意见。实际上这是非常错误的，作为小组领导人，不能专横，要听得进意见，要虚心，也要在组员的想法和能力更加出众的情况下，去肯定他们，去学习。首先承认自己的不足，然后真诚的去了解别人这是很重要的。

不要总是push别人，每个人开发有自己的习惯，节奏慢点就多体谅，不要过分催促。一开始我担心小组进度缓慢，任务堆积，到最后可能会东拼西凑应付了事。所以一直在催促他们快马加鞭的开发，但是事实证明，这样做除了徒增压力，无任何其他效果。一起合作应该给予对方充分的信任，尊重他们的学习开发速度，只需要在项目初期约定的阶段性汇报点一起验收成果即可。

学习是一个循序渐进的过程，想要掌握一门技术，没有捷径，只能不断地尝试，犯错，然后积累经验再改正。如果害怕莫名其妙的bug，害怕困难，那就永远不会进步。首先肯定客观规律，再心平气和地努力学习。无论是在开发还是在学习理论知识的过程中都是这样，不能指望一口吃成大胖子。我一开始阅读官网文档没有耐心，总是跳字，自以为差不多了就动手开始做，这样固然启动快，但是一旦遇到稍微复杂一点的bug就束手无策了。有的bug实际很简单，但是因为我根本不懂稍微基础一些的知识，以及看不懂报错，就浪费大量时间，这些都是需要改正的。

轮子很多，不必自己造，但是不了解别人造轮子依据的原理，永远也没办法把力气用在刀刃上。一开始想的个性化推荐功能可以用聚类/人工智能啥啥算法实现，一想到网上那么多现成的库可以用，并且一旦用进去就会让整个项目都显得“高大上”，就激动的不得了。但是等到梦想照进现实才发现，确实很多现成的东西可以用，还有极其详尽的教程，可是我不懂怎么用，就算数据摆在那里也套不进去。最后还发现，我自己想实现的功能若是搬进人工智能里简直就是杀鸡焉用宰牛刀，多此一举。

说了这么多还是有困惑，我一开始秉承的目的是做一些真正有用的东西并且投入实际，可是热情不足以支撑，做到后来，就没有持续的动力了，还有很多方面可以优化，也可以做的更好，只是一旦到了课设结项时，就没有动力再去继续了。我觉得这样不好，希望以后可以找到自己真正热爱的并且愿意为之奋斗终生的事业吧。

生命不息，奋斗不止，希望今后在每一次的尝试，犯错中我都能获得进步，跌倒后勇敢的爬起来，为成为一个为祖国做贡献，有价值的人。

### 5.2 陈彩婷

### 5.3 汪伦