单词王网站用户手册

目录

1. 引言

1.1. 编写目的

为帮助用户对本网站有更好的了解，获取基本的使用指南的帮助。用户手册将给出对本网站基本功能的介绍，并描述如何使用本网站进行单词的背诵和复习等操作，以及给出一些基本问题的解决方案。

1.2. 项目背景

本项目是个人开发者开发的，面向所有英语学习者的单词学习网站，旨在为公众提供更有效的单词学习方法和更友好的学习界面。

2. 网站概述

2.1.目标

通过部署网站，让用户能够随时随地进行单词的学习。

2.2.功能概述

1. 用户注册，保留学习进度。
2. 词汇量测试，基于词频和统计学方法的考量。
3. 单词书背诵。

3. 运行环境

Google Chrome 浏览器

4. 使用说明

4.1.主界面

主界面如下图所示：

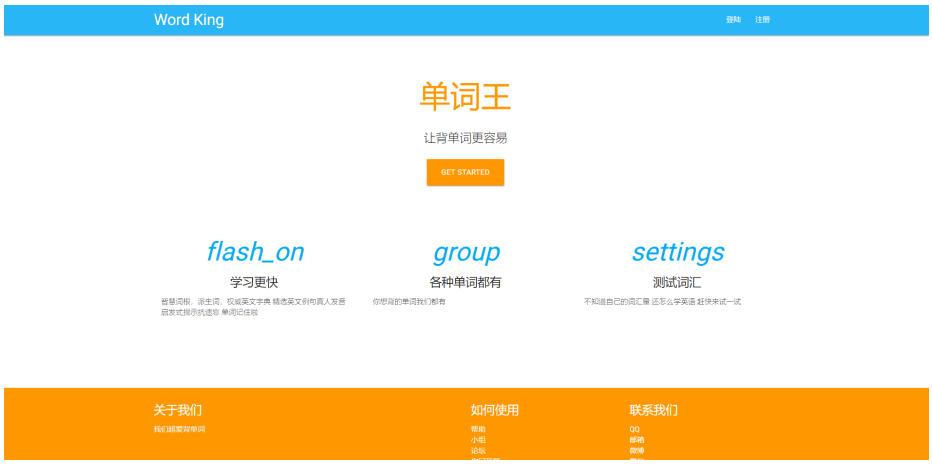

图 4.1-1 网站主界面

主界面中共有三个主要选项，分别是登录、注册和“get started”，即开始背诵。用户通过单击不同的按钮可进入对应的子页面进行操作。值得注意的是，在选择开始背诵之前，用户必须拥有本网站的账户并进行登录。如果没有账户，用户可进入注册页面进行账户注册。

4.2. 注册

注册页面如下图所示：

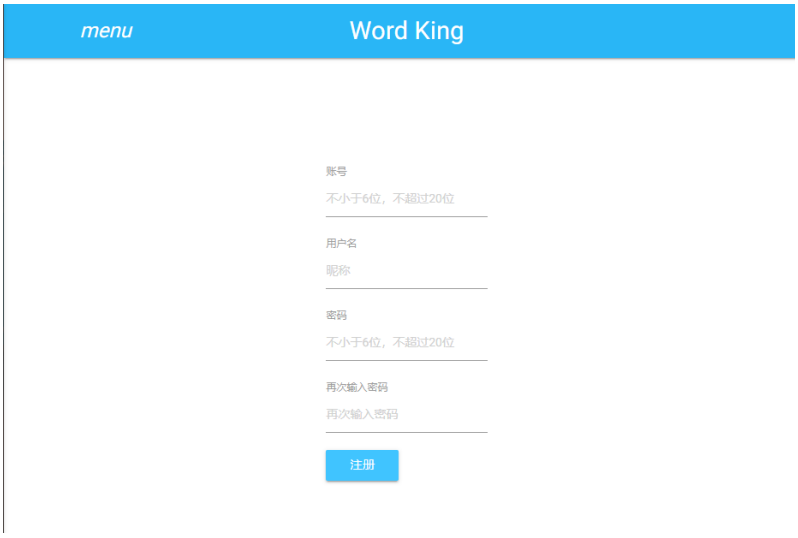

图 4.2-1 注册界面

用户进行注册时，需要输入账号、用户名和密码。

1. 账号由 26 位英文字母（区分大小写）、数字以及下划线组合而成，不可输入其他特殊符号，且长度不小于 6 位，不超过 20 位。
2. 密码的要求与账号相同。
3. 用户名除了英文字母、数字和下划线外，可输入中文。

如果输入错误，网站会对用户进行错误提示并要求重新注册账户。注册成功后，网站将会保存用户的账号和密码，并直接跳转到登录界面。

4.3. 登录

网站登录界面如下图所示：

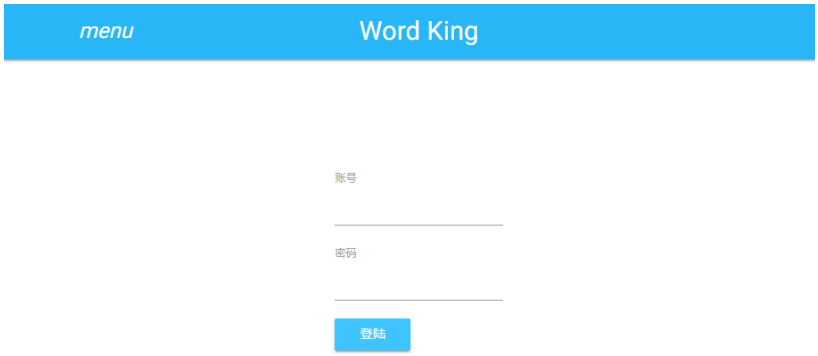

图 4.3-1 登录界面

在此界面，用户输入自己的账号和密码进行登录。如果账号或密码出错，网站将对用户进行错误提示并要求重新登录。如果登录成功，网站将进入用户登录成功模式，在网页右上角显示用户昵称，并跳转到单词书背诵界面，允许用户进行功能的选择和使用。

图 4.3-2 登录失败

图 4.3-3 登录成功

4.4. 主功能界面

在用户单击“get started”或登录成功后，网页将跳转至主功能界面，如下图所示：

图 4.4-1 主功能界面

此时，主功能界面将基于默认的背诵参数设置（用户可自主调整）显示如

下信息：

1. 单词书名称：用户选择背诵的单词书名。
2. 预计完成时间：根据艾兵浩斯记忆法和用户设置的背诵量、复习量计算出的预计完成背诵的时间。
3. 学习进度：显示需要背诵的单词总数和用户已经背诵的单词书，并给出进度条以可视化显示。
4. 今日背诵任务：显示目前需要新背诵的单词数，复习的单词数和新出现的单词数。

主功能界面中共有五个主要功能选择区：

1. 单词学习
   用户单击再次学习后，将重新进入主功能界面。
2. 继续学习
   用户单击继续学习后，将进入单词背诵界面。
3. 词汇量测试
   用户单击词汇量测试后，将进入词汇量测试界面。
4. 设置
   用户单击设置后，将进入背诵参数设置界面。
5. 退出
   用户单击退出后，将退出当前账户且不能继续使用各项功能。

4.5. 词汇量测试

词汇量测试界面如下图所示：

图 4.5-1 词汇量测试界面

进入词汇量测试界面后，网页将显示数十个单词，单词的难度按照从上到下的顺序递增。每个单词前有一个方框，用户单击该方框表示用户认识该单词，并会出现绿色的对勾，表示选中。

用户需要阅读所有单词，并选中其认识的单词。当选择完成后，用户应单击底部的提交按钮，网站将会显示计算得出的用户单词量。

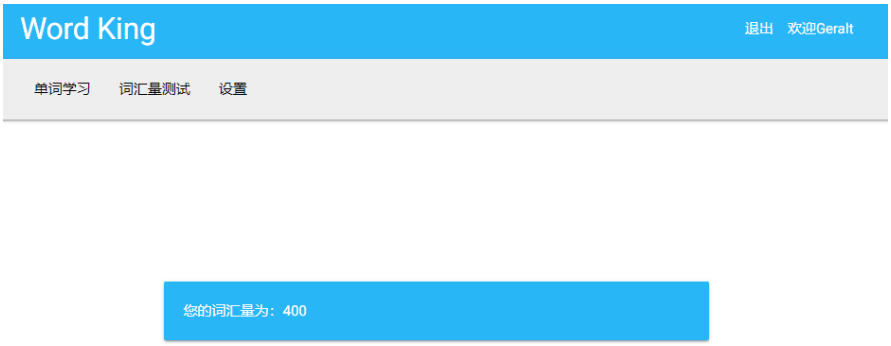

图 4.5-2 词汇量测试结果

词汇量测试完成后，用户可通过单击导航栏中的选项进入别的功能模块，或再次进行词汇量测试。

4.6. 背诵参数设置

用户单击导航栏中的设置条目后，可进入背诵参数设置界面，如下图所示：

图 4.6-1 设置界面

用户在设置界面中可对背诵的参数进行设置，包括每日背诵的新单词数量和每日复习的单词数量。

新词的设置和复习数的设置分别对应两个多选框，用户可在单击后进行选择操作。

4.7. 单词书背诵

用户在主功能界面点击“继续学习”按钮后，即可进入今日的单词背诵界面。单词背诵界面主要由如下几个部分组成：

1. 单词背诵卡
2. 单词内容卡

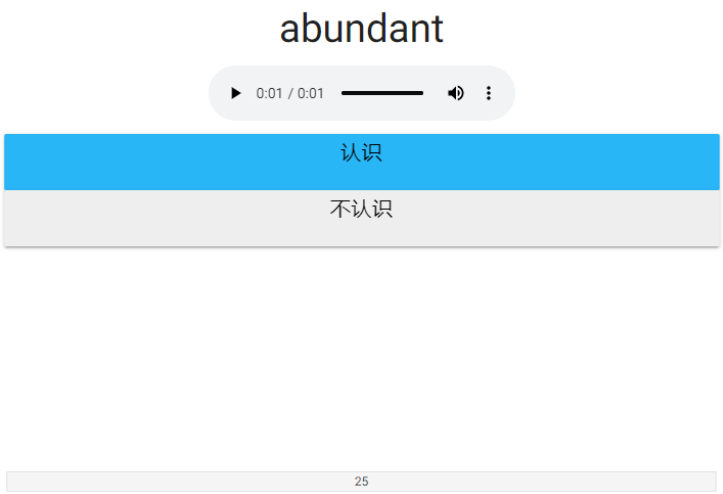

图 4.7-1 单词背诵卡

每当一个需要背诵的新单词出现，网站首先会显示一个单词背诵卡，并在底部显示仍需要背诵的单词数量。在单词背诵卡中，用户可以通过单击不同的按钮进行选择。

1. 单击音频按钮，网站将输出该单词的发音。
2. 单击认识，表示用户认识该单词，并显示单词内容卡。
3. 单击不认识，表示用户不认识该单词，并显示单词内容卡。

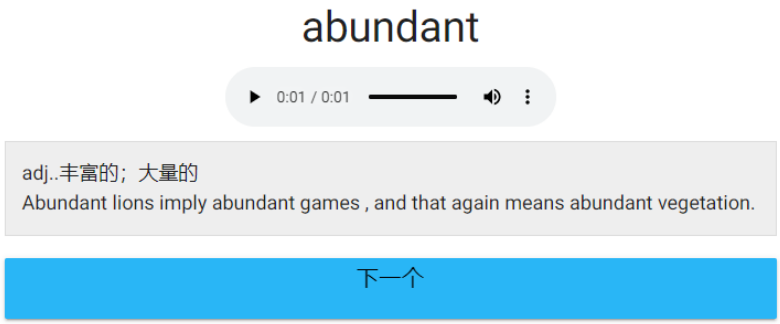

图 4.7-2 单词内容卡

在单词内容卡中，主要有两个部件：

1. 单词的词性、意义和例句。
2. “下一个”按钮，用户单击后将显示下一个单词的单词背诵卡。

4.8. 常见问题

1. 网页反应迟缓

由于服务器的限制，网页的跳转可能出现卡顿、迟缓等现象，用户需要耐心等待网页完成数据的加载和显示。

2. 无法注册或登录

用户应仔细检查是否已经注册账号，或是否输入的非法字符。

单词王网站管理员手册

目录

单词王管理员手册

1. 前端技术介绍

前端采用基本的 html+css+javascript 设置，其中运用了 Materialize CSS 框架优化界面，Materialize 具有内置的响应式设计，以便使用 Materialize 创建的网站将根据设备大小重新设计自己，并且 Materialize 类的创建方式使网站可以适合任何屏幕尺寸。同时在内容更换、表单提交等内容上用到了 javascript 的 jQuery 库。

2. 后端技术介绍

后端开发运用到了 python 的 Django 框架，Django 采用了 MVC 的软件设计模式，即模型 M，视图 V 和控制器 C。Django 已经成为 web 开发者的首选框架，是一个遵循 MVC 设计模式的框架。MVC 是 Model、View、Controller 三个单词的简写，分别代表模型、视图、控制器。

Django 视图不处理用户输入，而仅仅决定要展现哪些数据给用户，而 Django 模板仅仅决定如何展现 Django 视图指定的数据。或者说，Django 将 MVC 中的视图进一步分解为 Django 视图 和 Django 模板两个部分，分别决定“展现哪些数据” 和 “如何展现”，使得 Django 的模板可以根据需要随时替换，而不仅仅限制于内置的模板。

至于 MVC 控制器部分，由 Django 框架的 URLconf 来实现。URLconf 机制是使用正则表达式匹配 URL，然后调用合适的 Python 函数。URLconf 对于 URL 的规则没有任何限制，完全可以设计成任意的 URL 风格，不管是传统的，RESTful 的，

或者是另类的。框架把控制层给封装了，无非与数据交互这层都是数据库表的读，写，删除，更新的操作。在写程序的时候，只要调用相应的方法就行了，感觉很方便。只需要编写非常少的代码完成很多的事情。所以，它比 MVC 框架考虑的问题要深一步，因为我们程序员大都在写控制层的程序。这个工作交给了框架，仅需写很少的调用代码，大大提高了工作效率。

其中在设计时，基本目标是低耦合高内聚。框架里的不同层不应该知道对方的代码，除非它们确实需要。例如，模板系统不需要知道用户的 Web 请求，数据库层不需要了解如果显示数据，而视图并不关心所使用的模板系统。同时每个独特的概念或数据片应该存在且只存在于一个地方。避免冗余，做好标准化。合理的框架应该从尽可能少的信息中推断出尽可能多的需求。

3. 服务器端配置

本程序作为使用 python 开发的 web 应用，必须遵守 WSGI 协议，WSGI 协议是 python web 服务器网关接口，即 python 的 Django 应用程序与 Web 服务器之间的一种通信协议。为了让应用能在任何服务器上运行，就必须遵循这个协议。本应用为了实现 WSGI 协议，采用 gunicorn 作为 Web 服务器，再使用 Nginx 进行负载均衡，采用 Nginx+gunicorn 的部署方式。

为实现部署首先要对 gunicornj 进行配置，在 conf.py 中添加以下内容：

```
# coding:utf-8

import multiprocessing

bind = "127.0.0.1:8001"
workers = multiprocessing.cpu_count() * 2
```

```
workers_class = 'gevent'
```

保存好配置文件后，就可以通过如下的命令来启动应用来：

```
gunicorn --config=conf.py wbProject.wsgi:application
```

接下来需要配置 Nginx

安装完成之后，我们 cd 到/etc/nginx/的目录下,可以看到 Nginx 的所有配置文件。

其中 nginx.conf 文件为主配置文件，可以用来修改其全局配置；sites-available 存放你的配置文件，但是在这里添加配置是不会应用到 Nginx 的配置当中，需要软连接到同目录下的 sites-enabled 当中。但是在我实际操作的过程中，当我在 sites-available 修改好配置文件后，会自动更新到 sites-enabled。

在 nginx.conf 中添加如下配置：

```
    server {
        listen 8080;
            location / {
                    proxy_pass http://127.0.0.1:8001;
            }
            location /static {
                    alias /root/wbProject/static;
            }
    }
```

通过Gunicorn的Nginx配置中，我们只需要通过proxy_pass参数反向代理给运行在http://127.0.0.1:8001/上的Gunicorn

在每次完成对 Nginx 配置文件的修改后，需要通过如下的命令来重启 Nginx

```
    $ nginx -s reload
```

但是还有一个问题，到目前为止，uWSGI 和 gunicorn 都是直接通过命令行运行，并不能够在后台运行，也是当我们关闭了 xShell（或者你使用的是 Putty 及其他 SSH 连接的软件），将无法再访问到你的应用。所以我们需要让 uWSGI 或 gunicorn 在后台运行，也就是所谓的 daemon（守护进程）。

我们最好让我们的后台程序能够监控进程状态，还能在意外结束时自动重启，这就可以使用一个使用 Python 开发的进程管理程序 supervisor。

安装完成后，我们在/etc/supervisor/conf.d/目录下创建我们控制进程的配置文件，并以.conf 结尾，这样将会自动应用到主配置文件当中，创建后添加如下的配置内容：安装完成后，我们在/etc/supervisor/conf.d/目录下创建我们控制进程的配置文件，并以.conf 结尾，这样将会自动应用到主配置文件当中，

创建后添加如下的配置内容：

```
[program:demo]
command=/www/demo/venv/bin/gunicorn                    -c
/pushy/blog/gconfig.py run:app

directory=/www/demo  //项目目录

user=root

autorestart=true //设置自动重启

startretires=3  //重启失败 3 次
```

在上面的配置文件中，[program:demo]设置了进程名，这与之后操作进程的状态名称有关，为 demo;command 为进程运行的命令，必须使用绝对路径，并且使用虚拟环境下的 gunicorn 命令；user 指定了运行进程的用户，这里设置为 root

保存配置文件之后，我们需要通过命令来更新配置文件：

```
$ supervisorctl update
```

然后我们来启动这个进程：

```
$ supervisorctl start demo
```

这样进程就会在后台运行。