多背单词系统

项目开发计划

团队编号：第11组

团队成员：陈冠宇、王逸远、王佳程、翁天佑

完成日期：2024-7-13

目录

[1．引言 3](#_Toc171843699)

[1.1编写目的 3](#_Toc171843700)

[1.2项目背景 3](#_Toc171843701)

[1.3定义 3](#_Toc171843702)

[1.4参考资料 3](#_Toc171843703)

[2．概要设计 4](#_Toc171843704)

[2.1系统模块划分 4](#_Toc171843705)

[2.2系统结构设计 5](#_Toc171843706)

[2.3处理流程设计 6](#_Toc171843707)

[3． 数据库设计 7](#_Toc171843708)

[4．接口设计 11](#_Toc171843709)

[4.1外部接口 11](#_Toc171843710)

[4.2内部接口 12](#_Toc171843711)

[5．模块详细设计 12](#_Toc171843712)

[6. 界面设计 21](#_Toc171843713)

[6.1界面样式设计 21](#_Toc171843714)

[6.2界面交互设计 21](#_Toc171843715)

[7．出错处理设计 23](#_Toc171843716)

[7.1错误类型及出错处理对策 23](#_Toc171843717)

# 1．引言

## 1.1编写目的

本设计说明书旨在详细阐述词汇学习系统的设计和实现，为开发团队、项目管理者以及最终用户提供清晰的指导。本文档的主要读者包括项目的开发人员、测试人员以及项目投资者。

## 1.2项目背景

在当前数字化学习的趋势下，传统的单词记忆方法已难以满足用户的个性化学习需求针对现有背单词方法的局限性，如单词书法的局限性、Excel法的操作繁琐以及 APP法的用户体验限制，我们提出了“多背单词系统”。该系统是一款集成创新在线学习工具，它融合了传统记忆方法与现代技术，旨在提供一个高效、个性化的单词学习平台。

## 1.3定义

1、反馈机制:系统根据用户的学习反馈调整教学策略。

2、艾宾浩斯记忆曲线:用于计算和安排复习频率，优化记忆保持。

3、进度条：表示”记得”与”遗忘”的两个的比例，比单纯数字来的更直观。

4、记忆历史：观察曾经的记忆情况。

5、笔记区：为自己记单词添加笔记。

6、排序：顺序、乱序随机、按记忆率排序、按复习次数排序。

7、例句与关键词高亮：有些单词书做的比较好，会在例句上的单词或者单词所在词组做高亮，这样方便眼睛抓到重点，也符合”**词串记忆法”。**

**8、单词标记：太简单，已掌握，重难词。**

**9、快捷键：”键盘流”**，页面上的高频点击事件都可以用键盘快捷键代替

10、导入词书：网页版导入单词书功能

## 1.4参考资料

1. Django官方文档：[使用 Django | Django 文档 | Django (djangoproject.com)](https://docs.djangoproject.com/zh-hans/4.2/topics/)
2. 前后端语言手册和使用教程：[菜鸟教程 - 学的不仅是技术，更是梦想！ (runoob.com)](https://www.runoob.com/)
3. Django实战与项目部署：[从零开始的Django框架入门到实战教程(内含实战实例) - 01 创建项目与app、加入静态文件、模板语法介绍（学习笔记）\_django实例教程-CSDN博客](https://blog.csdn.net/Hjh1906008151/article/details/125477399?ops_request_misc=%7B%22request%5Fid%22%3A%22172060211616800172568031%22%2C%22scm%22%3A%2220140713.130102334..%22%7D&request_id=172060211616800172568031&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~baidu_landing_v2~default-6-125477399-null-null.142%5ev100%5epc_search_result_base5&utm_term=django%E5%AE%9E%E6%88%98&spm=1018.2226.3001.4187)
4. Django框架：[Python框架——最详细的Django框架入门\_python django教程-CSDN博客](https://blog.csdn.net/Z987421/article/details/125200435?ops_request_misc=%7B%22request%5Fid%22%3A%22172060217716800207074015%22%2C%22scm%22%3A%2220140713.130102334..%22%7D&request_id=172060217716800207074015&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~top_positive~default-2-125200435-null-null.142%5ev100%5epc_search_result_base5&utm_term=django&spm=1018.2226.3001.4187)
5. Pug模板：[pugjs/pug: Pug – robust, elegant, feature rich template engine for Node.js (github.com)](https://github.com/pugjs/pug)
6. Django：[django/django: The Web framework for perfectionists with deadlines. (github.com)](https://github.com/django/django)

# 2．概要设计

## 2.1系统模块划分

1. 用户认证模块

功能：

1、注册和登录

对于任何需要个性化的学习工具来说，用户认证都是基础设施的一部分，确保用户可以访问他们的个人数据并保护用户数据不被未授权访问。

2. 单词管理模块

功能:

1、单词数据库维护：增加、删除、编辑单词

2、单词排序显示：顺序、乱序随机、按记忆率排序、按复习次数排序

3、单词标记：标记为“太简单”，“已掌握”，“重难词”

该模块支持排序和单词标记等需求。单词排序功能提高了复习效率，帮助用户根据不同的学习需求调整学习策略。单词标记功能允许用户根据掌握程度分类复习，提高学习的个性化和效率。

3. 学习跟踪模块

功能:

1、艾宾浩斯记忆曲线的实现：根据记忆曲线调整复习频率

2、进度条显示：展示“记得”与“遗忘”的比例

3、记忆历史查看：查看单词的历史记忆记录

4、笔记功能：为单词添加、编辑笔记

该模块支持反馈机制、艾宾浩斯记忆曲线、进度条、记忆历史、笔记区对应的用户需求。通过艾宾浩斯记忆曲线，系统自动调整复习频率，帮助用户以科学的方式进行复习。进度条和记忆历史功能提供了对学习效果的直观反馈，增加了用户对学习进度的掌控。笔记功能增强了学习体验，帮助用户记录额外信息，加深记忆。

4. 导入模块

功能:

1、从web端导入单词书

该模块对应导入词书需求。用户能够轻松地扩展和自定义自己的词库，特别是教师或高级用户可以导入专业或定制的词汇表。

1. 从网站上爬取对应单词的解释和助记法

根据不同的 URL，从外部字典网站抓取单词的解释、助记法等信息。

5. 内容呈现模块

功能:

1、界面样式设计和页面逻辑切换

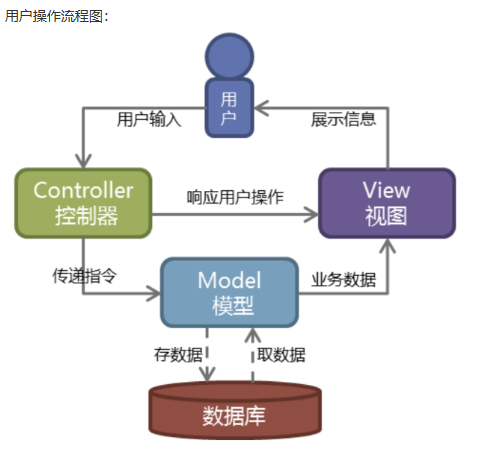
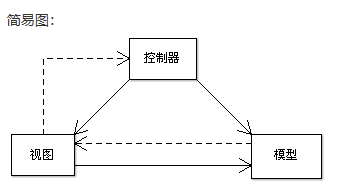
2、显示例句和关键词高亮

3、实现快捷键操作

该模块对应例句与关键词高亮、快捷键和友好的用户界面需求。例句高亮帮助用户理解单词在实际语境中的使用，提高学习的实际应用价值。快捷键支持提高用户交互效率，尤其是在进行大量复习时。

## 2.2系统结构设计

本系统采用客户端-服务器架构，前端使用HTML和JavaScript，后端使用Django框架。



## 2.3处理流程设计

主要处理流程：

1、获取单词详细信息

用户发起GET请求获取特定单词的详细信息，服务器查询数据库，检索单词的详情，如定义、发音、使用频率等，返回单词信息的JSON响应，更新单词标记2、用户通过POST请求更新单词的标记(如"已掌握”、"重难词")

服务器接收数据并验证，更新数据库中的单词标记，返回操作结果的JSON响应。

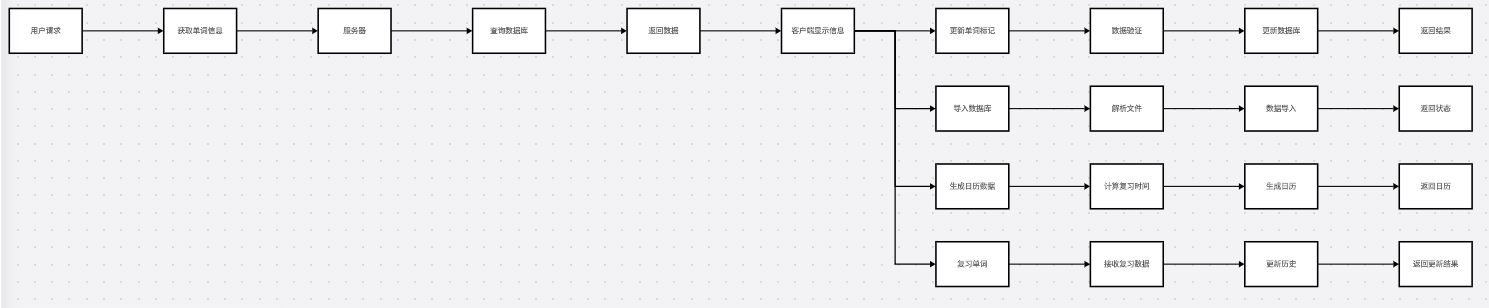
3、导入数据库

用户提交一个包含Excel文件路径的POST请求以导入单词数据，服务器解析Excel文件，读取数据，将数据批量导入数据库，返回操作状态，如成功或错误信息。

4、生成日历数据

用户请求获取基于艾宾浩斯复习计划的复习日历数据。服务器计算复习时间点，生成日历视图数据，返回日历数据的JSON响应。

5、复习单词用户进行单词复习并提交复习结果，服务器接收复习数据，更新用户的复习历史和记忆率，返回更新结果的JSON响应。



模块功能处理流程：

（1）用户认证模块

登录流程：

1、用户访问登录页面，填写用户名和密码。

2、提交表单后，Django 的视图函数接收请求。

3、视图函数使用 Django 的认证系统验证凭据。

4、如果认证成功，用户的登录状态存储在会话中，并重定向到主页。

5、如果失败，返回错误信息，并重新渲染登录页面。

（2）单词管理模块

单词添加：

1、管理员通过后台界面或者导入功能添加单词。

2、如果是导入功能，用户上传 Excel 文件，通过视图函数处理文件，解析数据并存储到数据库。

单词编辑/删除：

1、通过 Django 管理界面，管理员可以编辑或删除单词。

2、对应的 Django ORM 操作执行相应的数据库更新或删除操作。

单词标记和排序：

1、用户界面提供不同的排序和标记选项。

2、根据用户的选择，视图函数查询数据库，应用排序和过滤条件，返回结果到前端。

（3） 学习跟踪模块

艾宾浩斯记忆曲线应用：

1、根据用户的复习活动，后端定期更新复习时间和状态。

2、视图函数计算下次复习时间，并根据艾宾浩斯记忆曲线更新数据库。

进度跟踪：

1、用户完成复习后，前端发送数据到后端。

2、后端处理数据，更新用户的记忆历史和记得/遗忘的比例，存储进度信息。

笔记添加和查看：

1、用户在学习单词时可以添加笔记。

2、笔记数据通过表单提交，后端接收数据并更新数据库。

3、查看笔1、记时，后端查询相关单词的笔记信息并显示。

（4）导入模块

导入词书：

1、用户选择并上传 Excel 文件。

2、视图函数接收文件，使用 pandas 解析 Excel 数据。

3、解析后的数据通过 ORM 存入数据库中相应的表。

爬取解释：

1、根据输入的单词和 URL，选择对应的处理逻辑。

2、对于dict.cn，发送 HTTP 请求，解析返回的 HTML，提取单词释义等信息。

3、使用 BeautifulSoup 和正则表达式处理 HTML。返回抓取结果和状态码。

（5）内容呈现模块

例句显示和高亮：

1、单词学习页面包括例句。

2、前端 JavaScript 或服务器端模板处理，将关键词高亮。

快捷键实现：

1、快捷键功能通过 JavaScript 实现。

2、用户操作（如复习单词、切换单词列表）绑定快捷键事件，提高操作效率。

# 数据库设计

在 Django 中，使用 ORM (Object-Relational Mapping) 来抽象化和管理数据库操作是常见的做法。ORM 允许开发者以编程对象的方式处理数据库数据，而不需要直接编写SQL代码。这种方式提高了开发效率，也使得代码更加易读和可维护。

1. Books 表（单词书清单）

- 字段：

- BOOK: 单词书的英文名。

- BOOK\_zh: 单词书的中文名。

- BOOK\_abbr: 单词书的简写。

- uuid: 唯一标识符。

- create\_time: 创建时间。

- begin\_index: 列表开始的索引，是从0开始还是从1开始。

- hide: 是否隐藏这本书。



2. BookList 表（单词书的列表信息）

- 字段：

- BOOK: 关联的单词书名称。

- LIST: 列表序号。

- review\_dates: 所有复习日期（艾宾浩斯复习曲线时间）。

- review\_dates\_plus: 自愿复习日期。

- list\_uuid: 唯一标识符。

- list\_rate: 表记忆率。

- recent\_list\_rate: 近期表记忆率。

- review\_word\_counts: list 内单词复习次数。

- word\_num: list 内的单词数目。

- ebbinghaus\_counter: 艾宾浩斯复习次数。

- unlearned\_num: 仍需复习单词数。

- modify\_time: 上次修改时间。



3. Review 表（复习单词表）

- 字段：

- word: 英文单词。

- total\_num: 复习总次数。

- forget\_num: 忘记次数。

- rate: 单词遗忘率。

- LIST: list编号。

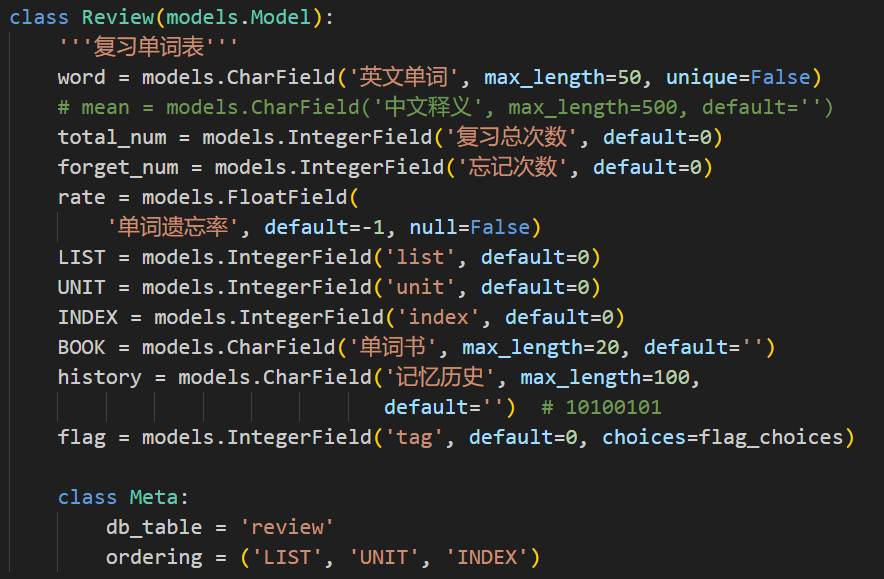
- UNIT: unit编号。

- INDEX: index编号。

- BOOK: 单词书名称。

- history: 记忆历史。

- flag: 标记状态。



4. Words 表（单纯的单词表）

- 字段：

- word: 英文单词。

- mean: 中文释义。

- note: 记忆法。

- total\_num: 复习总次数。

- forget\_num: 忘记次数。

- last\_forget\_num: 上次复习错误次数。

- sentence: 例句。

- rate: 单词遗忘率。

- history: 记忆历史。

- flag: 标记状态。

- webster: 是否被WebsterBuilder收录。

- mnemonic: 助记法。

- phonetic: 音标。

- antonym: 反义词。

- synonym: 近义词。

- derivative: 派生词。

- modify\_time: 上次修改时间。

 外键关系设计

1、BookList 表与 Books 表：

BookList.BOOK 是一个外键，关联到 Books.BOOK。这样可以确保每个 BookList 记录都对应到一个具体的 Books 记录。

2、Review 表与 Books 表：

Review.BOOK 是一个外键，关联到 Books.BOOK。这样每条复习记录都可以追溯到特定的单词书。

# 4．接口设计

## 4.1外部接口

1. 可执行文件(exe)，单机运行，不需要依赖任何环境；
2. 通过在python环境中执行python文件进行运行。

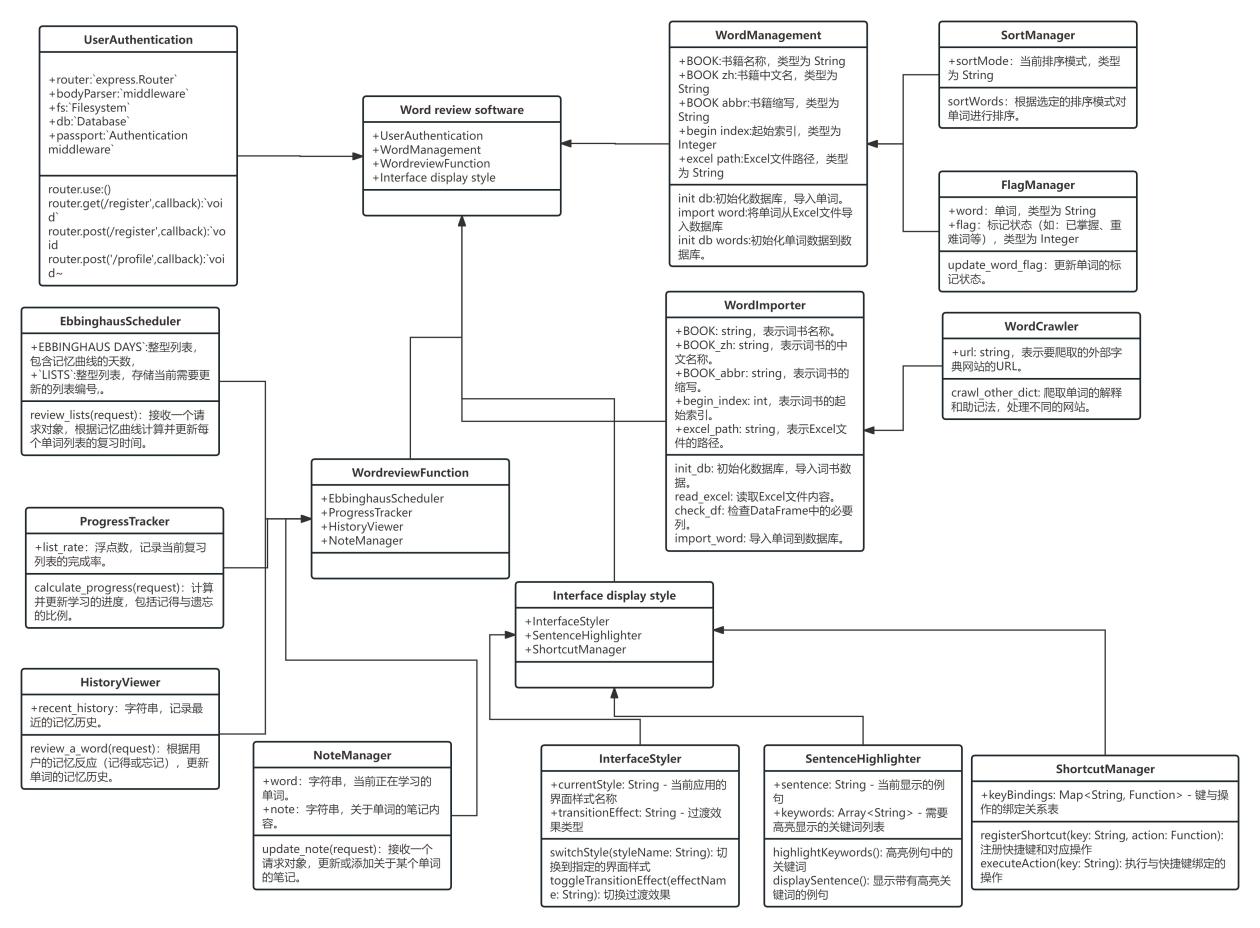
## 4.2内部接口

1、数据访问层与服务层接口:使用Django框架的ORM功能，允许业务逻辑层通过模型直接管理数据库操作，无需直接编写SQL代码，简化开发过程。

2、服务层与表示层接口:前端应用可以通过HTTP请求与后端进行数据交换，确保系统的响应性和可扩展性，在Web应用中实现特定功能，如更新数据库、处理用户提交的表单。

# 5．模块详细设计

总体类图展示

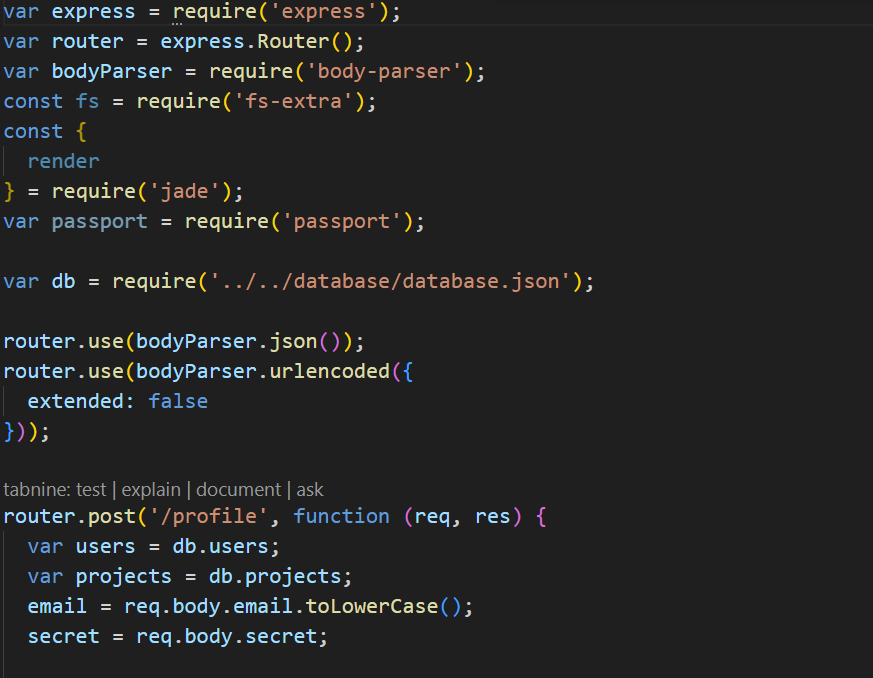
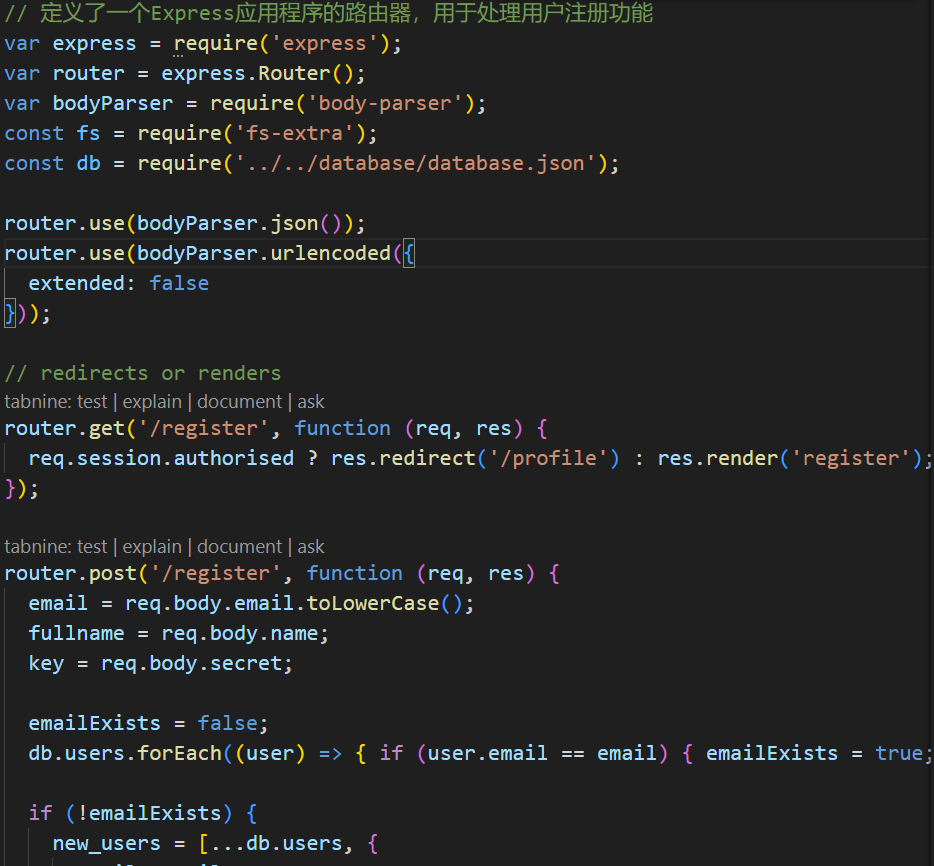


1. 用户认证模块

功能：

1、注册和登录

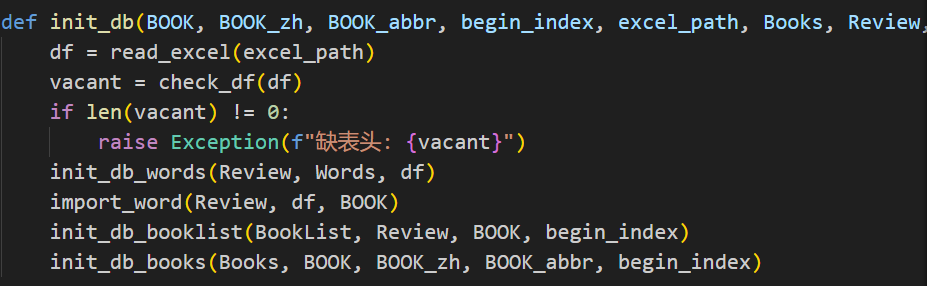
对于任何需要个性化的学习工具来说，用户认证都是基础设施的一部分，确保用户可以访问他们的个人数据并保护用户数据不被未授权访问。



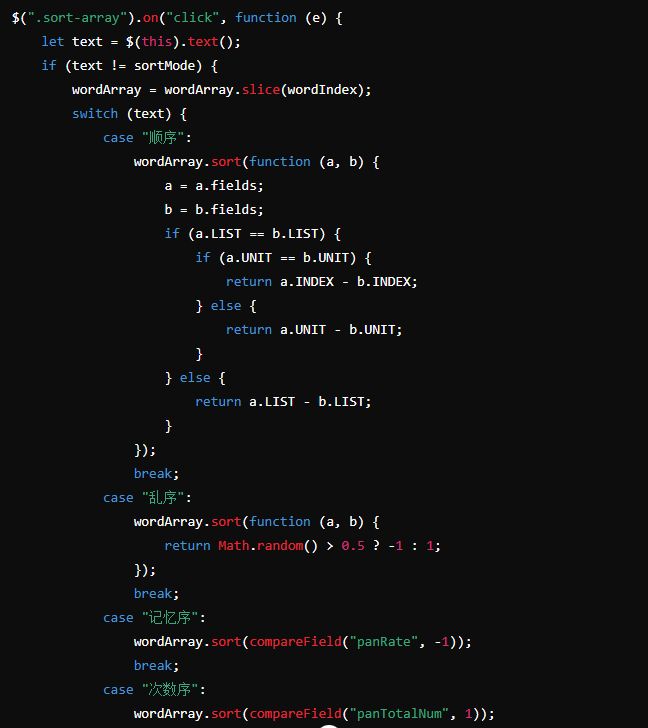
2. 单词管理模块

功能:

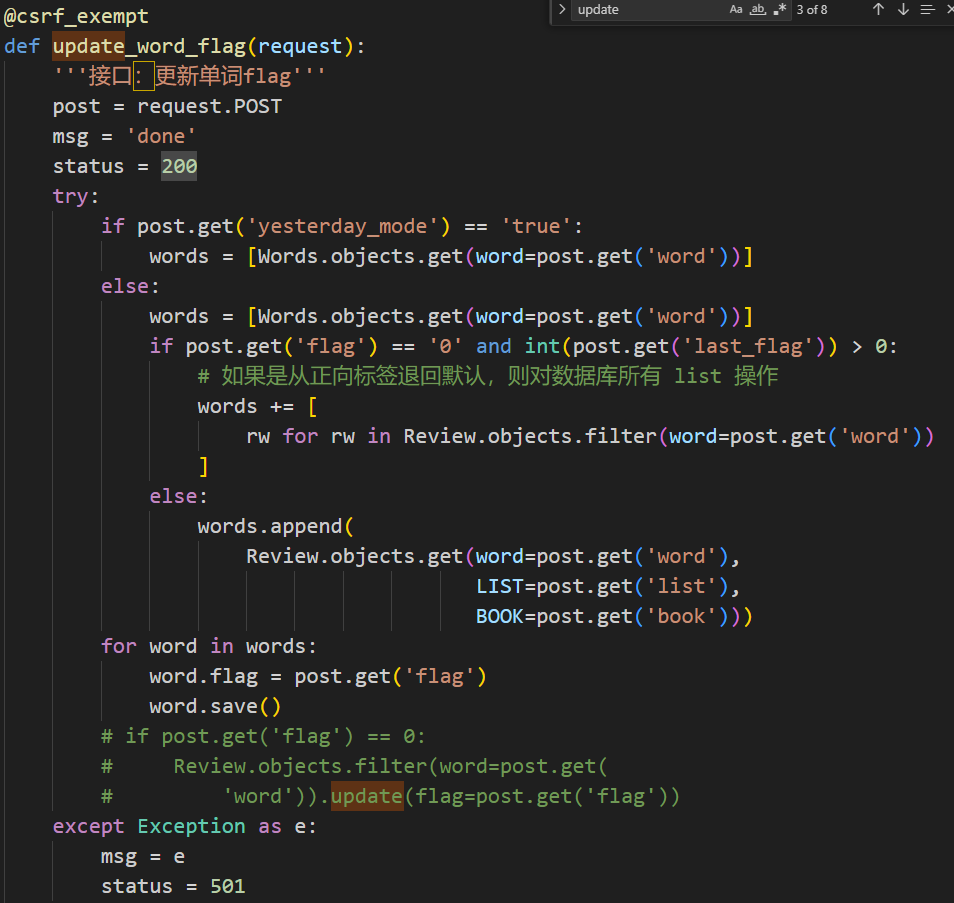
1. 单词数据库维护：增加、删除、编辑单词



1. 单词排序显示：顺序、乱序随机、按记忆率排序、按复习次数排序



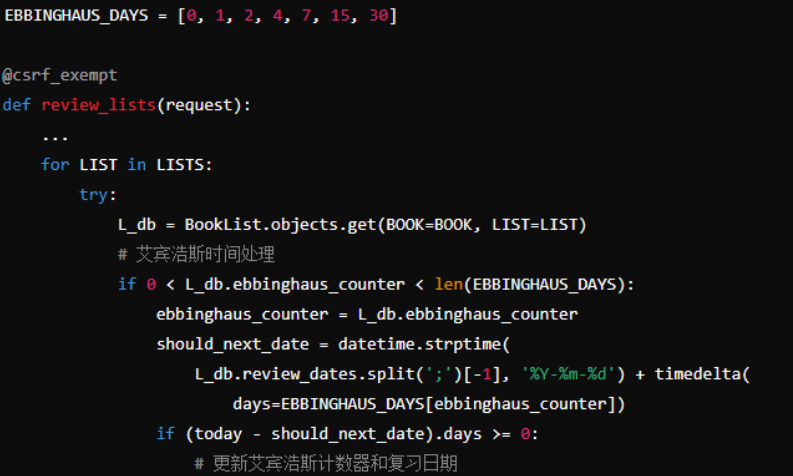
3、单词标记：标记为“太简单”，“已掌握”，“重难词”



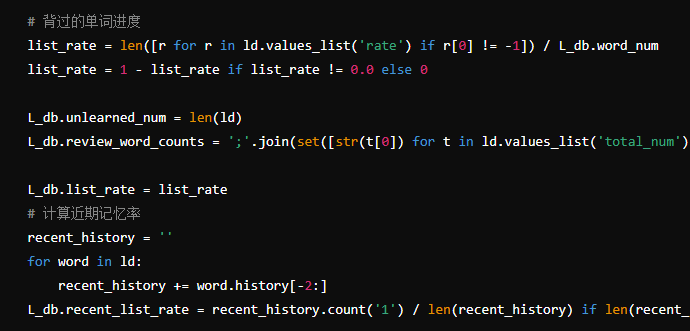
3. 学习跟踪模块

功能:

1. 艾宾浩斯记忆曲线的实现：根据记忆曲线调整复习频率



1. 进度条显示：展示“记得”与“遗忘”的比例



rate(list\_rate变量)计算逻辑:1d'查询获取了标记为未完全记住(flag <1)的单词。使用values 1ist('rate')方法获取所有单词的记忆率，其中’-1’表示未评估的单词，这些被排除在计算之外。计算已学习单词中记忆率非’-1’的比例，用总单词数除以这个比例，然后从、1减去得到的结果，得到整体复习的百分比。

recent rate(recent list rate变量)计算逻辑:遍历'1d”集合，累加每个单词的历史记录中最后两次的记忆结果(这假设history'字段存储的是记忆成功('1')或失败('0')的字符串)。计算近期复习中记忆成功的次数除以近期复习次数的比例。这两个百分比反映了用户对特定复习列表中单词的整体和近期记忆掌握程度，有助于用户了解他们的学习效果，并据此调整复习策略，

3、记忆历史查看：查看单词的历史记忆记录



4、笔记功能：为单词添加、编辑笔记

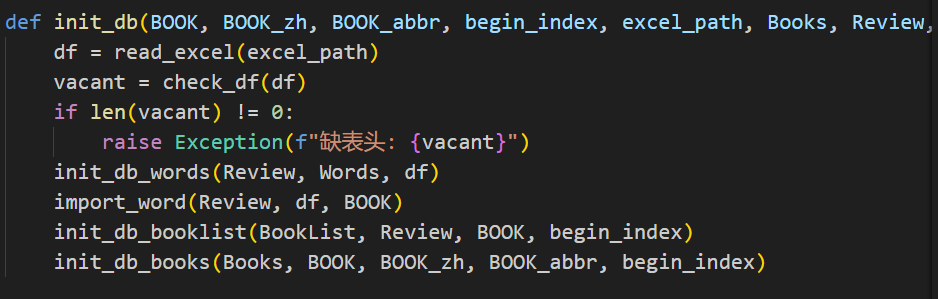


4. 导入模块

功能:

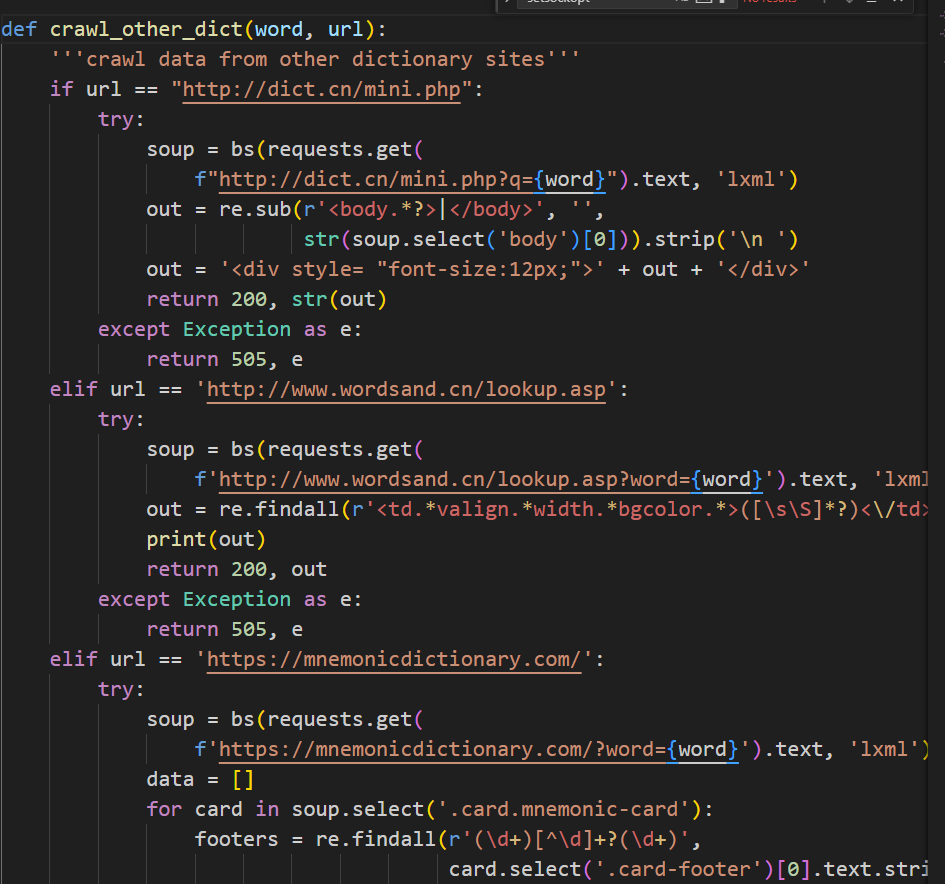
1、从web端导入单词书

该模块对应导入词书需求。用户能够轻松地扩展和自定义自己的词库，特别是教师或高级用户可以导入专业或定制的词汇表。



2、从网站上爬取对应单词的解释和助记法

根据不同的 URL，从外部字典网站抓取单词的解释、助记法等信息。



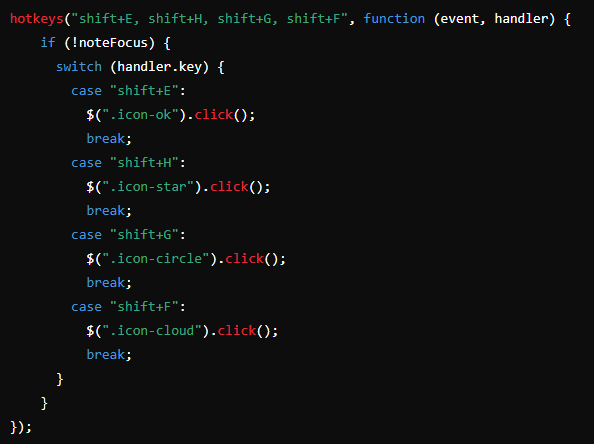
5. 内容呈现模块

功能:

1、界面样式设计和页面逻辑切换

2、显示例句和关键词高亮

3、实现快捷键操作



# 6. 界面设计

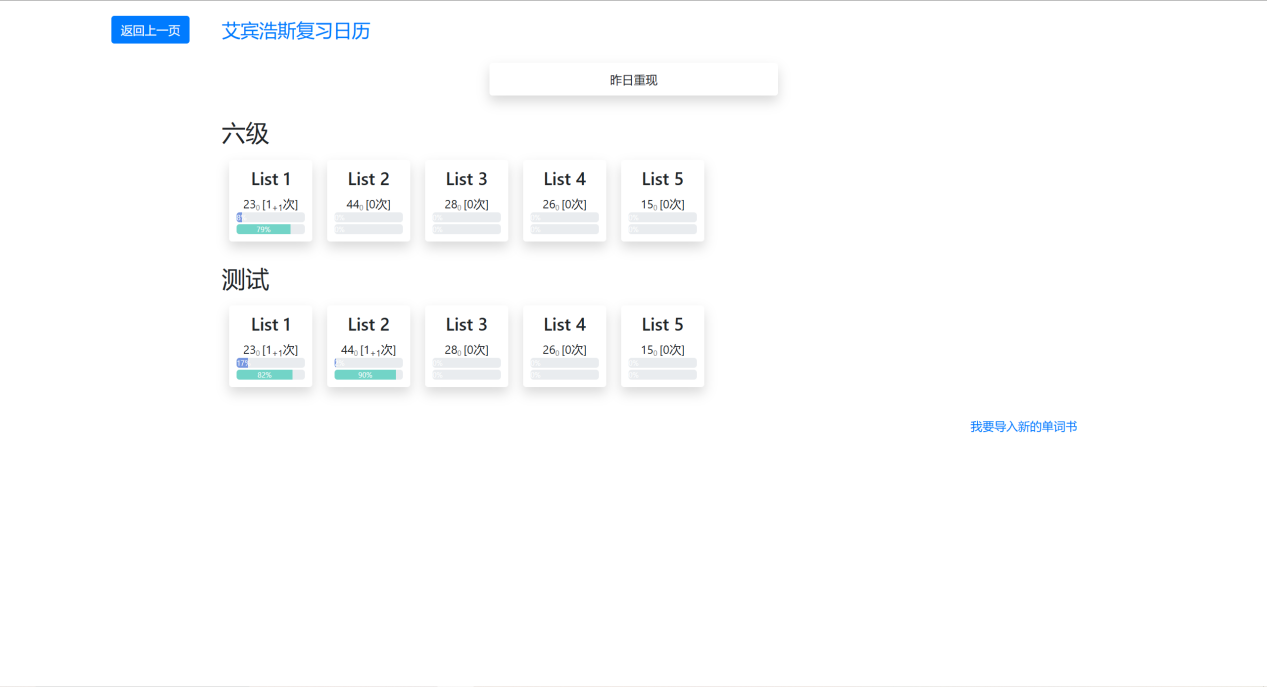
## 6.1界面样式设计

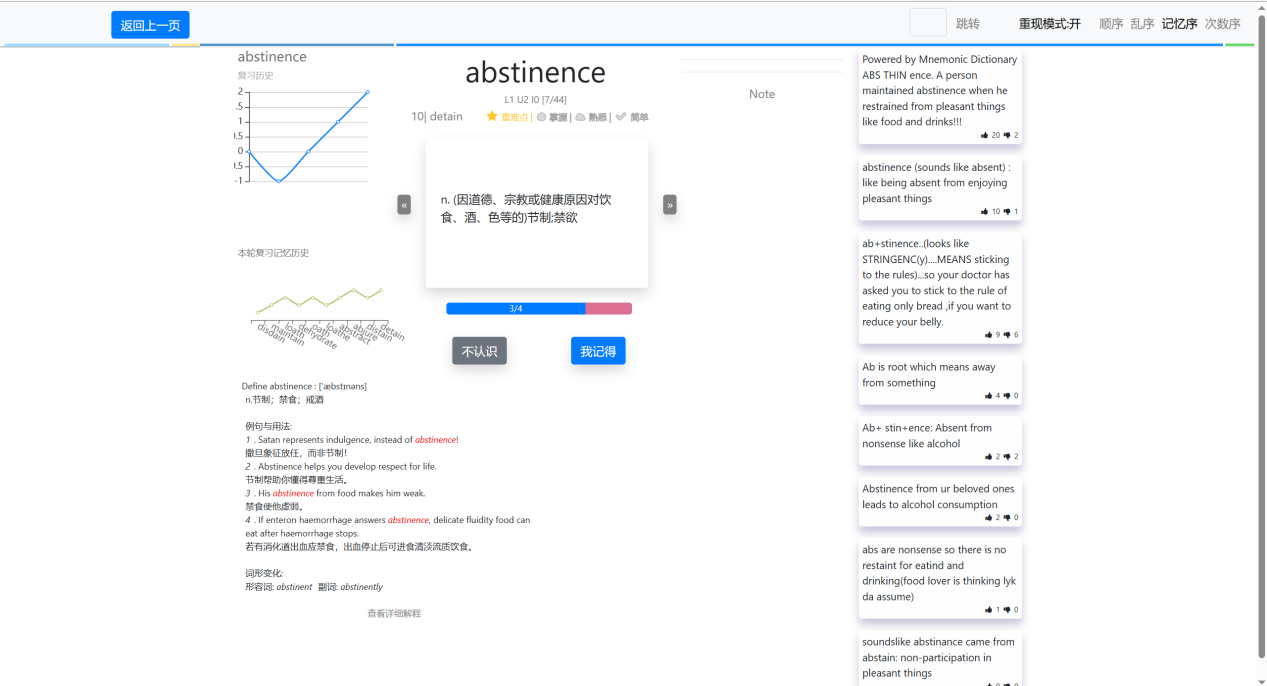
本系统的界面设计采用简洁、直观的布局，以白色和蓝色为主色调，提供清晰的视觉体验。界面上方设有导航栏，方便用户快速切换不同的学习模块。每个词条页面均包含单词、发音、词义以及用例，确保信息全面而易于理解。记忆率和历史记忆曲线直观显示在每个单词旁边，通过图表形式呈现，让用户能够直观地看到学习进度和记忆状态。

## 6.2界面交互设计

1. 学习进度反馈:采用进度条和图表展示学习效果和记忆曲线，增强交互性和用户的学习动力。快捷操作:界面设计包括键盘快捷键支持，允许用户通过快捷键快速记录学习笔记、查看词义和复习单词，有效提升学习效率。







2、动态反馈:用户对单词的记忆操作后，系统即时更新记忆率和记忆曲线，反馈给用户，增加学习的互动性和趣味性。

# 7．出错处理设计

## 7.1错误类型及出错处理对策

在开发复杂的Web应用时，合理地处理错误是确保应用稳定运行和提高用户体验的关键。在实际的应用中，错误处理不仅仅是捕获异常，更涵盖了对各种潜在问题的预防和对错误的适当响应。通过细致地分析代码中可能出现的问题，并设计相应的错误处理机制，可以极大地减少系统故障的发生，同时也便于问题的调试和快速修复。本报告将概述在开发基于Django框架的学习管理系统时遇到的几类典型错误，以及我们所采用的处理对策，包括数据导入错误、数据库查询异常、网络请求问题以及数据格式错误等，旨在展示如何通过有效的错误管理提升整个系统的健壮性和可靠性。

1. 输入验证错误

用户在输入数据时候需要输入符合提示的示例数据，如在 import\_db 函数中，用户必须输入一个有效的 begin\_index（0 或 1），如果输入不符合这些值，需要给出错误信息。



这里使用 try-except 捕捉转换成整数时可能出现的 ValueError 异常，并检查是否在允许的范围内。

2. 模型查询错误

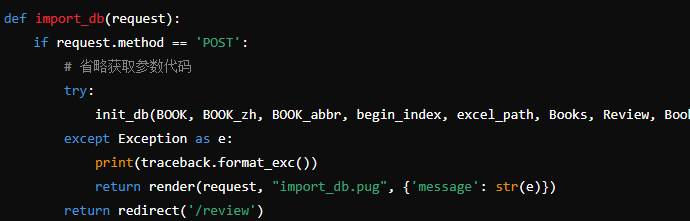
在 review\_lists 函数中，通过 BookList.objects.get() 查询数据库，如果查询不到指定的 BOOK 和 LIST，会抛出 DoesNotExist 异常。



这里使用 try-except 处理查询不到记录时的情况，然后进行适当的初始化或创建操作。

3. 异常捕捉与日志记录

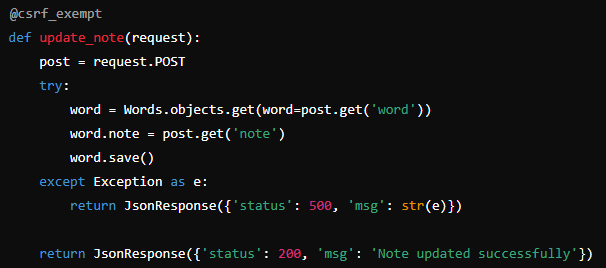
在进行数据库操作或者处理请求时捕捉未知异常，记录错误信息并向用户反馈。



这里使用 try-except 捕捉所有异常，并使用 traceback.format\_exc() 打印详细的异常堆栈信息，同时将异常信息传递给前端显示。

4. 处理 JSON 响应中的错误信息

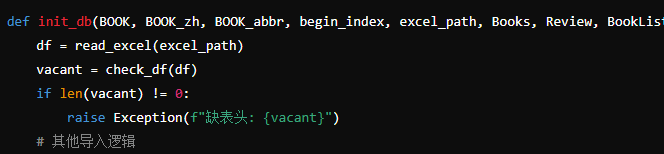
在处理 AJAX 请求或 API 调用时，捕捉异常并通过 JSON 响应返回错误状态和消息。



这里在操作数据库时可能出现异常，通过 JSON 响应向调用者返回错误信息，状态码 500 表示服务器内部错误。

5. 数据导入错误处理

在数据导入过程中，需要确保所有必要的字段都存在于数据源中。如果缺少必要的字段，应抛出异常并停止导入过程。



这段代码检查是否所有必要的列都存在于导入的 DataFrame 中，如果不存在，则抛出异常并输出缺失的列名。

6. 数据库查询错误

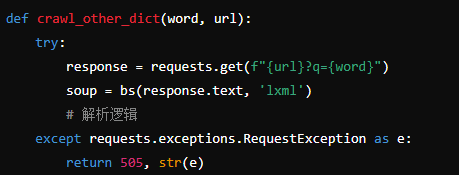
在查找数据库记录时，如果记录不存在，通常会抛出 DoesNotExist 异常。这需要被捕获并进行适当的处理。



这里处理了单词查询失败的情况，如果 Words 表中不存在该单词，会记录失败的单词，并继续处理下一个单词。

7. 网络请求错误

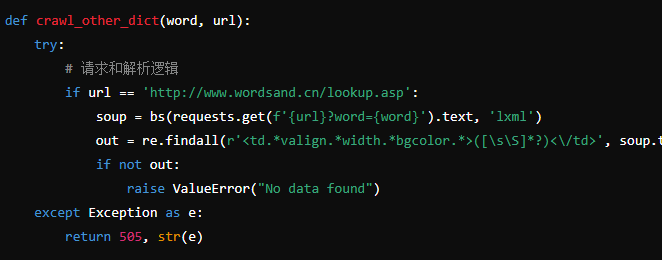
在从外部字典网站抓取数据时，可能会遇到网络错误或数据格式解析错误，这些都需要被妥善处理。



这段代码需要处理来自 requests 库可能抛出的异常，如连接错误、超时等。

8. 数据格式错误

处理外部数据时，如果数据格式不符合预期，也需要进行错误处理。



这里捕获了一般的异常，并且特别处理了数据解析时可能遇到的情况，比如预期的数据未找到。