WordHub - 简洁高效的英语词汇学习软件

北京大学2025程序设计实习第120组大作业

WordHub 旨在通过现代化的交互界面、科学的记忆方法和趣味性的游戏模块，为用户提供一个高效、沉浸式的单词学习环境。本项目不仅实现了完整的单词学习闭环，还包含了用户系统、成就激励、数据可视化和 AI 互动游戏等多种功能。

💻 访问我们的 GitHub 仓库：
https://github.com/stibiums/WordHub

🎬 功能演示

▶ ▶️ 点击查看：程序安装演示

▶ ▶️ 点击查看：注册与用户登录

▶ ▶️ 点击查看：用户界面展示

▶ ▶️ 点击查看：查询单词展示

▶ ▶️ 点击查看：学习模式展示

▶ ▶️ 点击查看：导入词库展示

▶ ▶️ 点击查看：游戏模块展示

✨ 1. 程序功能介绍 (核心功能)

- **科学学习系统**:
  - **学习模式**: 智能推荐新单词进行学习。

    - **复习模式**: 基于自定义算法，自动筛选需要复习的单词。
- **强大的词库管理**:
    - 支持从本地 `TXT` 文件导入自定义词库，格式兼容性强，并可通过在线词典 API 自动补全详细释义。（可见 `词库导入样例.txt` ）
    - 内置常见的英语词库（如 CET4、CET6、GRE 等）。
    - 支持创建多个词库，并可在不同词库间自由切换。
- **快速信息索引**: 提供强大的单词查询功能，可快速检索本地词库或通过在线 API 获取更详细的实时词汇信息。
- **数据可视化**:
    - 通过多种图表，直观展示用户的学习总览、每日学习量、正确率变化等。
    - 清晰追踪每一个单词的复习次数、掌握程度和最后复习时间。
- **游戏化学习**:
    - 内置经典的 **Wordle** 猜词游戏，寓教于乐。
    - 集成基于 **DeepSeek** 的"看描述猜单词"游戏，提供全新的 AI 互动学习体验。
- **完善的用户与成就系统**:
    - 支持用户注册与登录，所有学习数据云端同步（模拟）。
    - 独特的**成就系统**，当用户达成学习里程碑时，会弹出精美的 Toast 提示以示鼓励。
- **现代化桌面体验**:
    - 采用 Qt Widgets 构建，性能卓越，响应迅速。
    - 采用深色主题（AMOLED），并适配 Windows 系统的暗色标题栏，提供统一、沉浸的视觉体验。
    - 提供专业的 `WordHub.exe` 安装包，方便在任何 Windows 电脑上部署。

👥 2. 小组成员分工情况

2.1 小组成员

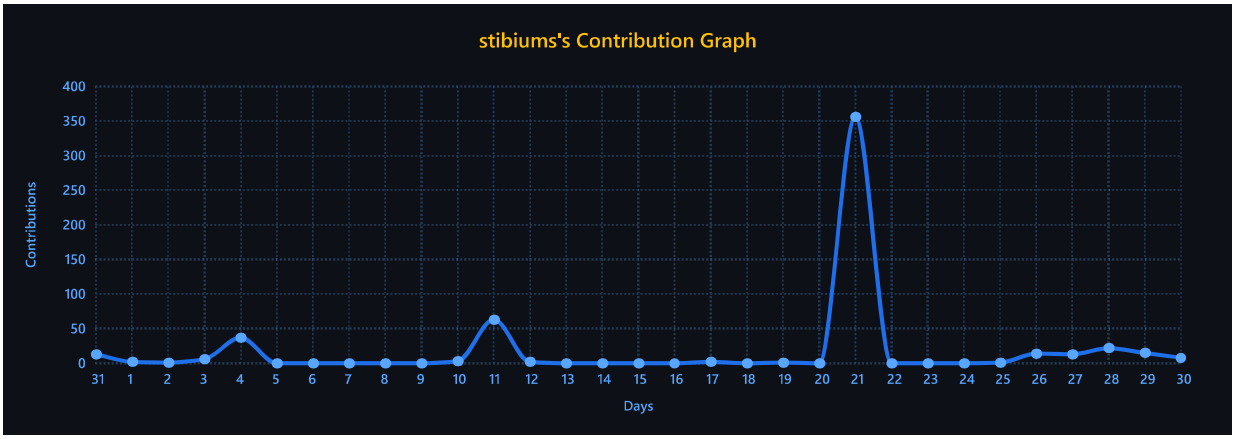
成员	GitHub	主要负责模块	具体分工
刘继轩	stibiums	**后端核心和杂项**	负责整体项目架构设计；数据库模块（SQLite）、数据导入处理、用户系统、学习记录、网络词典调用、成就系统的实现、用户类核心逻辑的编写。用嵌入式python解决游戏模块调用问题。打包发布应用程序完成应用文档。解决杂项问题。
郑星浩	Staaaaaaaaar	**前端界面**	负责全部 UI 界面的设计与实现，包括主窗口、学习、查询、用户中心、数据可视化图表、游戏等界面的开发与美化。 主要提交记录见fork仓库

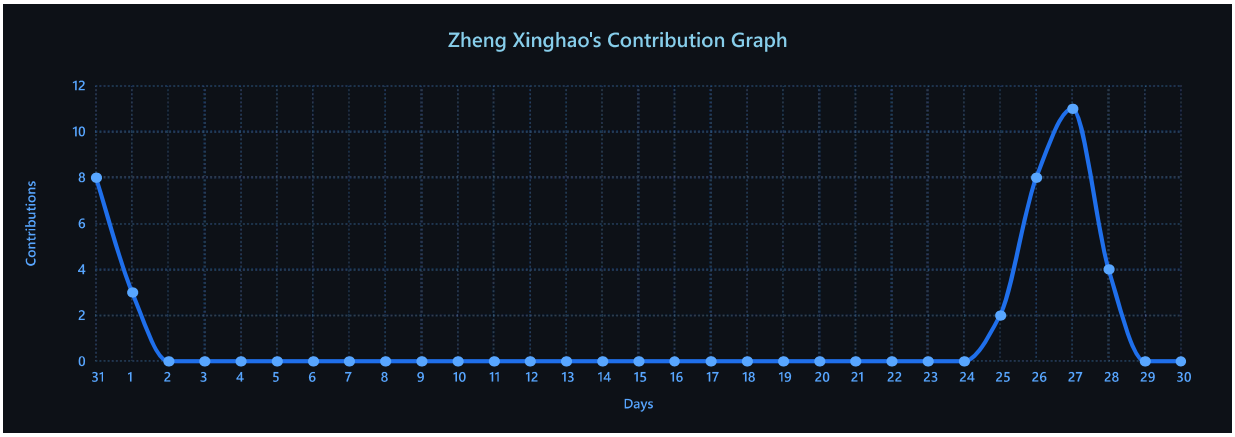
成员	GitHub	主要负责模块	具体分工
郑嘉祺	missswiftie	游戏模块以及其他	负责字典源文件解析（.mdx转.json脚本）；负责 Wordle 和“看描述猜单词”两个游戏的核心逻辑实现，包括 Python 端的 AI 接口调用和 C++ 端的整合；负责用户注册登录界面以及用户界面（包括用户信息、用户登录热力图等）。

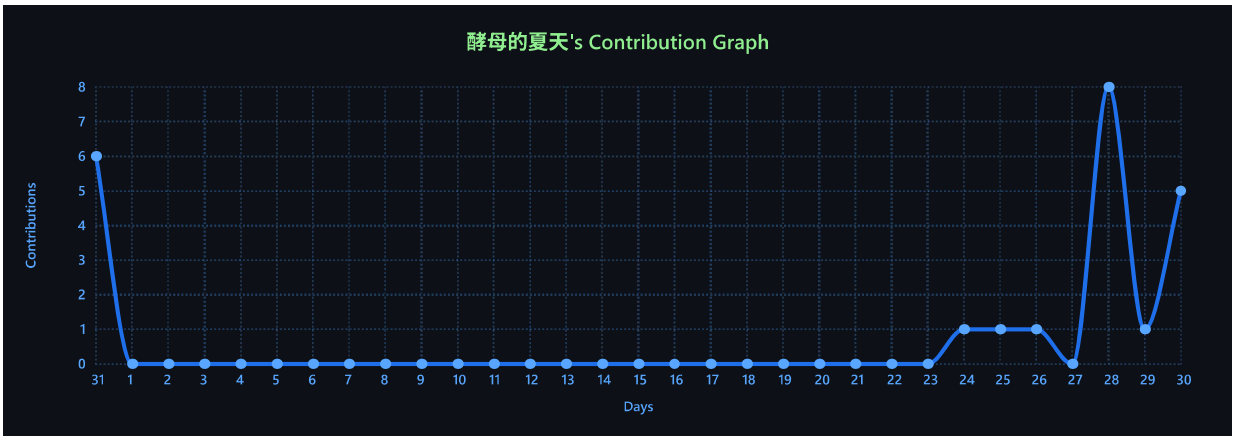
2.2 项目贡献

贡献者

团队提交活动图







🏛️ 3. 项目模块与类设计

3.1 模块结构

```
WordHub_Project_Root/
├── WordHub/                          # Qt C++ 项目源码目录
│   ├── UI/                           # UI 相关文件 (.ui, .cpp, .h, .qrc)
│   ├── back_head/                    # 后端模块头文件
│   ├── back_src/                     # 后端模块源文件
│   ├── API/                          # 外部 API 接口封装
│   ├── guess_according_to_description/ # “看描述猜单词”游戏的 C++ 接口
│   ├── WordHub.pro                   # Qt 项目配置文件
│   └── main.cpp                      # 程序入口
├── python_reasoner/                  # Python 脚本模块 (供游戏调用)
├── 配置文件/                         # 运行所需的配置文件 (e.g., config.zip)
├── docs/                             # 项目相关文档 (设计、规范等)
└── WordHub_Setup.iss                 # Inno Setup 打包脚本
```

3.2 主要类设计

点击类名即可查看每个类的详细介绍文档

后端核心

类名 (Class)	主要职责 (Responsibilities)
`Learner`	核心业务逻辑类（单例），负责管理当前用户状态、学习进度、成就解锁等，是UI与数据之间的桥梁。
`WordDatabase`	数据库操作类，封装所有与 SQLite 相关的操作，包括单词、词库、用户、学习记录的增删改查。
`utils`	定义的数据实体，如 `Word`, `Category` 等。
`Wordloader`	负责从文件导入单词。

类名 (Class)	主要职责 (Responsibilities)
`DictionaryAPI`	负责调用在线词典API。
`guess_word`	包括后端`python_reasoner.py`、后端交互`guess_word`类以及前端界面`guess_word_widget`类。
`wordle`	通过`wordle`类进行wordle游戏。

前端核心

类名 (Class)	主要职责与设计理念
`QueryWidget`	**单词查询的“多面手”**。该类负责为用户提供灵活的单词查询体验。它支持本地数据库和在线API两种查询方式，采用多标签页（Tab）管理查询历史，便于用户多词对比和快速切换。通过信号槽机制与主窗口和成就系统解耦，查询成功时自动触发成就解锁。分类收藏功能采用弹出菜单和多选设计，提升用户管理单词的效率和体验。
`LearnWidget`	**学习流程的“调度中心”**。该类聚合了词库管理、学习、复习、测试等核心学习流程。采用多页面（QStackedWidget）切换，保证流程清晰。通过数据可视化（如柱状图、进度条）直观展示学习进度。词库管理支持本地导入和多词库切换，学习/复习/测试流程高度模块化，便于扩展和维护。
`AchievementWidget`	**成就系统的“展示橱窗”**。该类负责加载、展示所有成就，并根据用户学习行为动态刷新解锁状态。采用滚动区域和分组布局，保证成就展示的美观和可扩展性。通过信号槽与主窗口和UserWidget解耦，成就解锁时自动弹出Toast提示，增强用户激励。
`GameWidget`	**趣味学习的“游戏大厅”**。该类作为游戏模块的入口，集成Wordle和“看描述猜单词”两大游戏。采用QStackedWidget管理不同游戏界面，保证切换流畅。每次进入游戏时动态创建游戏实例，确保每局游戏状态独立。通过信号槽与成就系统联动，游戏达成条件时自动解锁相关成就。

💡 4. 项目总结与反思

4.1 项目总结

本项目成功实现了 WordHub 单词学习软件的预定目标。我们构建了一个功能完善、体验流畅的桌面应用程序。通过 Qt 框架，我们实现了高效的前后端分离；通过 SQLite，

我们保证了数据的可靠存储和快速访问；通过模块化的设计，我们清晰地划分了学习、游戏、用户、成就等多个核心功能，并保证了它们之间的良好协作。

4.2 项目反思

- **技术收获**: 通过本次项目，我们深入实践了 C++/Qt 的大型项目开发流程，对信号与槽机制、UI设计、数据库交互、多进程通信（C++ 与 Python）等技术有了更深刻的理解。
- **遇到的挑战**: 在开发初期，模块间的接口定义不够清晰，导致了部分返工。一开始前端和后端之间的协作不够顺畅，导致了部分功能实现的延误。通过后期的沟通和接口规范化，这些问题得到了有效解决。
- **团队协作**: 团队成员间的分工明确，各自负责不同模块，保证了项目的高效推进。通过定期的代码审查和讨论，我们确保了代码质量和功能实现的一致性。
- **使用git**: 在项目开发过程中，我们使用 Git 进行版本控制，确保了代码的可追溯性和协作的高效性。每个成员都能在自己的分支上独立工作，减少了冲突和重复劳动。
- **可改进之处**:
  i. **多线程优化**: 目前单词导入和网络请求在主线程执行，未来可以将其移至工作线程，避免界面卡顿。
  ii. **代码复用**: 部分UI控件的代码可以进一步封装成自定义Widget，以提高复用性。
  iii. **更多功能**: 未来可以考虑增加更多的学习模式和游戏模块，如拼写测试、听写等，以丰富用户体验。

🚀 附：快速上手指南

普通用户

如果您是普通用户，想要快速体验 WordHub 的功能，请按照以下步骤操作：

1. **下载可执行文件**:
   - 前往 Releases 页面，下载最新版本的 `WordHub.exe` 安装包。
   - 解压缩后，双击 `WordHub.exe` 运行程序。

开发者

如果您是开发者，想要参与项目开发或修改，请按照以下步骤操作：

1. **获取源码**:

```
# 本项目的源代码公开地址即为本仓库地址
git clone https://github.com/stibiums/WordHub.git
```

2. **环境配置**: 使用 **Qt Creator** 打开项目根目录下的 `WordHub/WordHub.pro` 文件。请确保使用 **Qt 6.x** 版本和 **MinGW 64-bit** 编译器。
3. **构建运行**:
    - 直接在 Qt Creator 中构建并运行项目。
    - **重要**: 首次运行前，请将 `配置文件/config.zip` 解压到编译生成的可执行文件（`.exe`）所在的目录下。

开源许可

本项目采用 MIT License