项目文档

需求分析

本项目旨在建立一个百闻牌平衡性管理系统，用于整合手游《阴阳师：百闻牌》自公测以来的平衡性调整信息。具体来说，项目通过使用数据库来记录式神、卡牌等游戏内容在历次平衡性中的数值和效果变化，并通过前端框架实现用户图形界面上的操作，通过后端框架进行操作请求的验证逻辑和数据库交互。

概要设计

身份区分

根据使用者的不同需求，主要区分出两种身份角色：“玩家”和“管理员”。不同身份在使用系统时的权限不同。“玩家”身份主要用于给游戏玩家提供数据查询功能，而“管理员”身份主要用于对新的平衡性调整内容的记录。

用户注册

- 首次使用系统需要进行注册。对于“玩家”身份的注册无限制，而“管理员”身份的注册必须额外提供符合要求的密钥（邀请码）。用户必须登录身份以使用系统功能。系统暂不支持用户信息的更改。

用户登录

- 完成注册的用户需要登录来使用系统功能。一旦完成登录，客户端会持续向服务器发送请求来维持登录状态。客户端关闭时会自动退出登录。登录超时时用户需要重新登录。

系统功能

数据查询：任意用户身份可用。要求必须选择数据表，可以选择数据字段、或通过文本进行搜索。

数据修改：仅“管理员”身份可用。允许新增、修改、删除数据条目，但操作后的数据格式仍须符合各个字段的要求（由服务器对请求及参数进行验证）。暂不支持通过客户端程序修改数据库中数据表和字段的定义。

数据批量导入/导出：导出功能为任意用户身份可用，在选择了数据表并完成查询后，勾选想要导出的条目，点击按钮弹出文件保存窗口。导入功能仅“管理员”身份可用，在选择了数据表后从本地选择要读取的文件，程序将会发送批量更新请求（即根据每个条目的主键，分别进行新增/编辑）至服务器，并返回每个条目的更新结果。

详细设计

数据库设计

以下表中加下划线的字段为主键，*斜体*字段为外键。大写英文字母表明各字段的类型。未说明的字段均不允许空值（或者空字符串）。（实际上，由于本项目使用了`Flask-SQLAlchemy`插件作为ORM，所有类型为CHAR的字段都会注册为VARCHAR类型）

“用户”表

- User(uid INT, user_type VARCHAR(6), username VARCHAR(20), pwd_salt CHAR(32), salted_hash CHAR(64), reg_key CHAR(8), login_status BOOL, last_online_time DATETIME)
- 各字段分别表示：用户id，用户身份类型（玩家/管理员），用户名，用户登录的随机盐值，密码哈希的加盐后再次哈希，注册时所填写的邀请码，用户的当前登录状态，用户最后一次在线的时间。
- 仅允许`last_online_time`字段为空值、`invite_key`字段为空字符串。

“式神”表

- SS(ssid INT, ss_update_time DATE, ss_pack VARCHAR(4), *from_ssid* INT, ss_name VARCHAR(10), *ssvid* INT)
- 各字段分别表示：式神id，式神更新版本数据的时间，式神所属资料片版本（召唤物式神同于其从属式神），召唤物所从属的式神id（仅限召唤物），式神名称，当前采用的式神版本数据。
- 仅允许`from_ssid`字段为空值。
- `from_ssid`字段（非空值时）指向“式神”表中的主键`ssid`（不允许是自身的`ssid`字段值）。`ssvid`字段指向“式神版本数据”表中的主键`ssvid`。

“式神版本数据”表

- SS_Version(ssvid INT, ssv_update_time DATE, ss_type VARCHAR(3), ss_color VARCHAR(2), ss_atk INT, ss_hp INT, ss_desc VARCHAR(200))
- 各字段分别表示：式神版本数据id，该式神版本的更新时间，式神类型（式神/召唤物），式神派系（红莲/苍叶/青岚/紫岩/无），式神基础攻击力，式神基础生命，式神基础能力描述。
- 仅允许`ssv_update_time`字段为空值、`ss_desc`字段为空字符串。

“卡牌”表

- Card(cid INT, card_update_time DATE, *ssid* INT, card_name VARCHAR(10), card_rarity VARCHAR(3), *cvid* INT)
- 各字段分别表示：卡牌id，卡牌更新版本数据的时间，卡牌所属的式神id，卡牌名称，卡牌稀有度（R/SR/SSR），当前采用的卡牌版本数据。
- 仅允许`ssid`字段为空值。
- `ssid`字段（非空值时）指向“式神”表中的主键`ssid`（可以是衍生式神，例如来自鬼使白的“魂狩”）。`cvid`字段指向“卡牌版本数据”表中的主键`cvid`。

“卡牌版本数据”表

- Card_Version(cvid INT, cv_update_time DATE, card_type CHAR(2), card_level INT, card_desc VARCHAR(200), card_has_target INT, **fields)
- 各字段分别表示：卡牌版本数据id，该卡牌版本的更新时间，卡牌基本类型（法术/战斗/形态/幻境），卡牌等级（使用卡牌所需的式神勾玉等级），卡牌描述，卡牌是否要求有目标（0=无目标，1=有目标且无合法目标时不可使用，2=有目标但无合法目标时也可使用）。其中`**fields`实际为多组字段，分别用于记录法术牌、战斗牌、形态牌、幻境牌、觉醒等不同效果。
- 法术牌相关字段：暂无。
- 战斗牌相关字段：(zd_atk INT, zd_shd INT)，分别表示：战斗牌临时攻击力，战斗牌护甲。
- 形态牌相关字段：(xt_atk INT, xt_hp INT)，分别表示：形态牌基础攻击力，形态牌基础生命。
- 幻境牌相关字段：(hj_dur INT)，表示：幻境牌基础耐久。
- 觉醒相关字段：(incl_jx BOOL, jx_atk INT, jx_hp INT)，分别表示：卡牌是否含觉醒效果，觉醒效果（如有）的永久攻击力加成，觉醒效果（如有）的永久生命加成。
- 以上所有字段中，仅允许`cv_update_time`字段为空值、`card_desc`字段为空字符串。非自身卡牌类型的相关字段默认为0值。

功能实现

用户注册

- 注册时向服务器发送用户名和密码，由服务器随机生成16字节的随机盐值并保存为字符串，同时在数据库中记录密码哈希加盐后的再次哈希值，用于登录时进行验证（前端应用程序不会发送原密码，而是发送其哈希值）。
- 注册时可选填邀请码，由服务器验证通过后自动成为“管理员”身份用户，否则为“玩家”身份用户。注册完成后用户身份暂不支持更改。

- 通用邀请码可通过运行项目中的`/backend/reg_key_gen.py`获取，关联到某个已经注册的“管理员”身份用户的邀请码请参考`/backend/app.py --> def register --> def check_reg_key`中的相关逻辑。

用户登录与保活

- 前端不直接明文发送登录密码，而是先将密码哈希后发送。服务器接收用户登录请求并进行参数检查，如果登录成功则向数据库中记录其最近一次登录时间。
- 前端应用程序以一定时间间隔发送保持登录状态（并非再次登录）的请求，该请求同样会向数据库中更新当前用户的最近一次登录时间。
- 用户可以在主界面窗口点击按钮来重新登录。

用户权限控制

- 前端应用程序发出的大部分请求都附带用户id（通过请求头部的`Authentication`字段），并由服务器判断用户登录状态（是否超时）、用户是否有权限进行所请求的数据库操作。对数据库有写入操作的请求（新增、编辑、删除、批量导入）会要求发起请求的登录用户为“管理员”身份。

流程说明

- 前端应用程序首先将按钮控件的点击事件绑定到相应的处理函数，在其中进行必要的参数检查，然后将带参数的请求发送到后端服务器，由服务器进行参数检查并尝试执行，期间有任何问题则会弹出警告窗口（前端直接弹窗、不发送请求，或者发送请求后显示来自服务器的错误提示消息）。
- 对于新增操作，在有且仅有一行数据被选中（使用复选框）的情况下，会自动复制该行的数据到编辑窗口，方便管理员进行条目的复制。
- 对于数据库的写入操作，主要由后端服务器通过`Flask-SQLAlchemy`提供的数据库抽象模型来尝试执行。其中对于新增操作，主要检查非空字段是否均已填写。对于编辑操作，在检查非空的基础上，不允许对主键值进行修改。对于删除操作，主要由后端服务器检查要删除条目的各个字段与数据库中的条目是否一致。
- 对于批量导入操作，前端应用程序只调用一次接口，由后端服务器依次检查并尝试执行其中每一条目的新增或编辑操作，然后记录结果（不论成功与否）并统一返回。

部署方法

Windows 11

从源代码部署

1. 下载安装Python：从官方网站选择对应安装包进行下载安装。（备注：选择Python 3.11是因为`PyQt6-tools`暂时不兼容Python 3.12+；同时虽然Python 3.11的最新版本为3.11.10，但该版本为安全补丁版本、未提供安装包，故选择使用了3.11.9版本。）
2. 下载安装PostgreSQL：从官方网站选择`PostgreSQL 17`版本的对应安装包进行下载安装。（请全选安装选项，且安装过程中的语言选项不要手动更改，可能导致安装报错：`Problem running post-install step. Installation may not complete correctly. The database cluster initialisation failed.`）

3. 配置PostgreSQL服务器和数据库：从开始菜单启动`pgAdmin`应用程序，在左侧“对象浏览器”找到所安装的PostgreSQL 17版本，使用安装时设置的密码登录。选择其下的“数据库”栏，右键选择“创建”->“数据库”，在弹出窗口的“常规”选项卡中“数据库”一栏填写数据库名称，然后点击保存，完成数据库创建。
4. 部署环境：在本项目的根目录下打开终端，输入执行`python -m venv .venv`创建虚拟环境，然后输入执行`.venv\Scripts\activate`激活虚拟环境，最后输入执行`pip install -r requirements.txt`安装依赖库。
5. 设置后端服务器的运行配置：项目`backend`目录下创建或编辑`database_config.json`和`server_config.json`两份配置文件，调整数据库连接和后端服务器的运行配置。
6. 运行后端服务器：项目根目录下使用终端输入执行`.venv\Scripts\activate`激活虚拟环境之后，先输入执行`cd backend`再输入执行`python app.py`即可在本地运行服务器。
7. 运行前端应用程序：项目根目录下使用终端输入执行`.venv\Scripts\activate`激活虚拟环境之后，先输入执行`cd frontend`再输入执行`python qt_app.py`即可启动客户端。

问题报告

目前主要存在的问题：

1. 在Win11系统下，前端应用程序从PyCharm运行时，可能出现`进程已结束，退出代码为 -1073740940 (0xC0000374)`的程序异常中止问题，例如在使用“用户注册”按钮切换到注册窗口，随后关闭窗口回到登录窗口，再次点击“用户注册”按钮的情况下，出现应用程序无响应、随后异常退出。类似地，正常使用同一个按钮达到两次时也可能会产生该异常退出（例如，在成功删除一个条目后，再次点击按钮尝试删除另一条目）。另外，直接关闭登录窗口时也可能出现此情况。该问题可能是由Qt库的C语言部分产生的堆损坏问题，暂时难以通过Python部分的程序编写来解决。
2. 在项目作者的电脑上，后端服务器与前端应用程序同时运行时，前端应用程序的网络通信具有明显的较高延迟，但在电脑开启了VPN的情况下有明显好转，其原理未知。