

PKU ddler 项目功能报告

刘霆煦 黎宣萱 许鲟

项目分工:

刘霆煦: 日历部分、功能报告

黎宣萱: 任务安排部分、功能报告

许鲟: 健康系统部分、功能报告、视频介绍

项目简介: 我们的项目名称叫做 PKU ddler, 旨在帮助各位同学进行每日的日程规划, 在每天提醒同学当天需要完成的日程以及建议开始的日程。同时, 为了同学关注的身体健康, 我们特意加入了健康模块, 帮助同学了解健康状况, 规划锻炼安排。我们的项目功能大概包含三个部分: 日历部分, 任务安排部分, 以及健康系统部分。接下来我们将一一介绍每个模块的具体功能与具体实现细节。

一, 日历部分

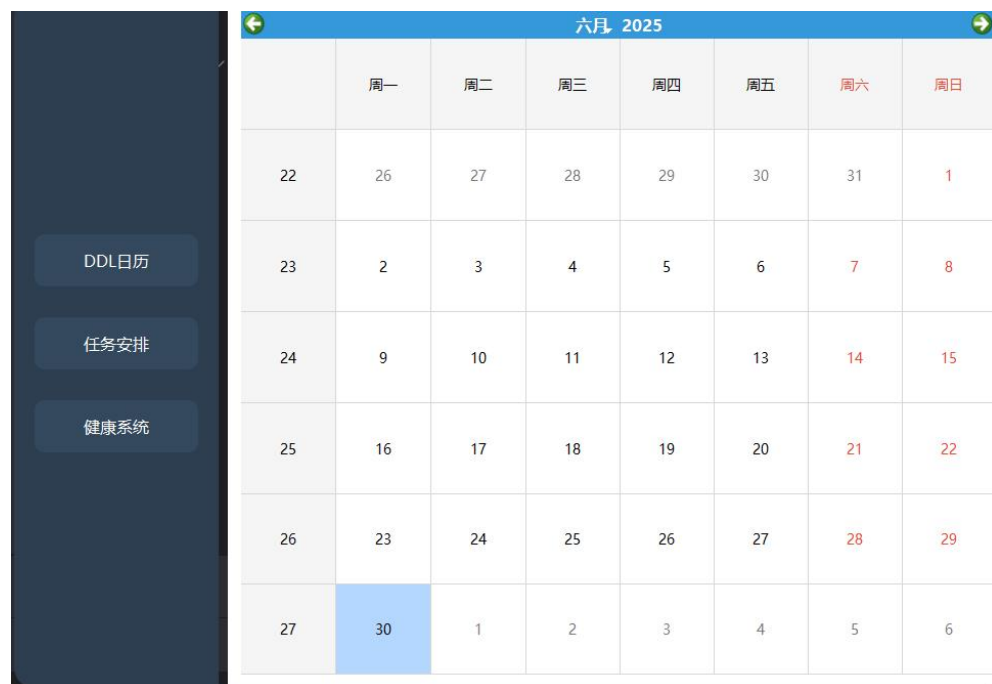
本模块为校园生活管理系统的重要组成部分, 旨在为用户提供可视化的 DDL 日历功能, 便于管理个人事务, 如考试、作业、娱乐、锻炼等事项, 并扩展到健康管理的日程显示整合, 提升时间管理效率与用户体验。

(1) 主要功能

日历视图展示

采用 `QCalendarWidget` 显示全年日历, 支持用户选中某一具体日期。

美化样式, 突出选中日期、周末高亮 (红色) 等。



	周一	周二	周三	周四	周五	周六	周日
22	26	27	28	29	30	31	1
23	2	3	4	5	6	7	8
24	9	10	11	12	13	14	15
25	16	17	18	19	20	21	22
26	23	24	25	26	27	28	29
27	30	1	2	3	4	5	6

事件添加功能

用户可通过“+”按钮在某一天添加事件, 选择时间、类别、填写 20 字以内的事件内容。

分类包括: 考试、作业、学工、玩乐、锻炼。

The image shows a '添加日程' (Add Event) dialog box. It has a title bar with a close button (X) and a help icon (?). The form includes a time selection field with '00' for hours and '00' for minutes, followed by '时' (hours) and '分' (minutes). Below this is a dropdown menu for '事件类别' (Event Category) with '考试' (Exam) selected. A text input field for '事件内容 (不超过20字):' (Event Content, no more than 20 characters) is present. At the bottom is a '保存' (Save) button. The dialog is overlaid on a calendar grid showing dates 26, 27, 28, and 29.

事件查看功能

点击任一日期，可弹出窗口查看该日最多 3 条事件，按时间排序，突出重要内容。

The image shows a '日程安排' (Event Schedule) dialog box. It has a title bar with a close button (X) and a help icon (?). The dialog displays a list of events for a specific date, sorted by time. The events are: '2025年06月04日 00:00 - 考试' (2025年06月04日 00:00 - Exam) and '2025年06月04日 00:00 - 作业' (2025年06月04日 00:00 - Homework). Below the list is a '程设大作业' (Programming Assignment) entry. A '+' button is at the bottom right. The dialog is overlaid on a calendar grid for June 2025, showing dates from 22 to 29.

事件存储

所有事件以 JSON 文件 (calendar_events.json) 本地持久化存储，便于后续读取与管理。

```
[
    {
        "year": 2025,
        "month": 6,
        "day": 3,
        "hour": 16,
        "minute": 30,
        "category": "锻炼",
        "content": "保加利亚分腿蹲12*3组"
    },
    {
        "year": 2025,
        "month": 6,
        "day": 4,
        "hour": 0,
        "minute": 0,
        "category": "考试",
        "content": "高数"
    }
],
```

左侧导航栏

页面左侧采用侧边栏样式，提供模块切换（如“任务安排”、“健康系统”）。在健康模块中更换按钮为“体测系统”、“健康建议”等，增强模块拓展性。

(2) 技术要点

- 界面实现：使用 PyQt5 构建 UI 界面，模块化设计，主日历页面类为 `CalendarPage`，健康模块为 `HealthPage`。
- 样式美化：通过 `QStyleSheet` 及 `QTextCharFormat` 实现高度自定义的 UI 风格（如高亮周末、鼠标悬停提示）。
- 事件对话框：使用 `QDialog` 创建弹窗形式的事件添加与展示交互，提升界面友好度。
- 数据处理：使用 Python 标准库 `json` 实现事件数据存取，并进行排序与筛选。

(4) 代码结构说明

文件/类名	功能描述
<code>AddEventDialog</code>	添加事件对话框，输入时间、分类、内容
<code>DateEventsDialog</code>	展示当前日期已添加事件列表
<code>save_event/load_events</code>	JSON 文件读写操作
<code>Ui_Calendar</code>	主日历页面界面初始化类
<code>CalendarPage</code>	封装主日历功能，处理交互逻辑

(5) 项目特色与创新点

- 支持类别标注与事件内容控制，便于查询与分类管理。
- 统一样式风格，符合现代扁平化设计审美。

- 模块化设计，方便未来拓展更多页面（如打卡系统、学分系统等）。
- 日期与健康系统整合：健康模块中保留插入日程功能，体现“日程一体化管理”理念。

(6) 后续改进方向

- 增加事件分类颜色标识或图标提示。
- 支持重复事件添加与提醒功能。
- 提供事件编辑与删除功能，提升操作灵活性。
- 优化数据存储方式，如改为 SQLite 数据库，提升查询效率。

二、任务安排部分

采用时间管理中的“四象限法则”（艾森豪威尔矩阵）对任务进行分类和可视化。主要功能包括四象限可视化、任务管理功能。实现手动添加任务、自动从 JSON 文件导入任务、任务卡片拖拽功能、任务卡片底色区分。

(1) 继承自 QMainWindow 的主类 TaskPriority 中 initUI () 函数初始化 UI 组件和布局，搭建主窗口和主布局，管理四象限布局 and 任务展示，处理用户输入和任务加载。使用 setRenderHint 设置抗锯齿渲染和图像平滑，让界面更美观。enterOn(0, 0) 将视图中心对齐到坐标原点 (0, 0)，使四象限对称分布居中显示。创建一个单行文本输入框 QLineEdit，用于用户输入任务名。setPlaceholderText() 设置占位符文字，当输入框为空时显示提示语。

```
self.task_input = QLineEdit(self)
self.task_input.setPlaceholderText("请输入任务名称")
self.add_button = QPushButton("添加任务", self)
self.add_button.clicked.connect(self.handle_add_task)
```

创建按钮 add_button，文字是“添加任务”。将按钮的点击事件连接到方法 handle_add_task()，点击后会输入框中的任务添加到界面上。创建一个水平布局 QHBoxLayout()，用于放置输入框和按钮，使它们在同一行。addWidget() 将控件添加到这个布局中。创建垂直布局 QVBoxLayout()：第一部分放 self.view；第二部分放输入栏。创建容器 QWidget() 作为主窗口的内容部件，将刚才设计的垂直布局放入容器，调用 self.setCentralWidget(container) 把它作为主窗口的核心组件。

```
input_layout = QHBoxLayout()
input_layout.addWidget(self.task_input)
input_layout.addWidget(self.add_button)
```

```
main_layout = QVBoxLayout()
main_layout.addWidget(self.view)
main_layout.addLayout(input_layout)
```

```
container = QWidget()
container.setLayout(main_layout)
self.setCentralWidget(container)
```

(2) draw_quadrants () 调用 add_arrow () 和 add_label () 在场景中画出“十字坐标线”并加上象限标签、箭头、标题等元素。add_label () 添加带 HTML 样式的文字标签，支持设置颜色、字体、居中对齐、加粗、字体大小；应用样式并设置文本区域大小和显示位置；将文本项加入场景中进行显示。

`add_arrow()` 函数是一个箭头绘制函数，构造黑色线条和填充函数，构造一个向上和向右的箭头，添加至坐标轴顶点。

在 `draw_quadrant()` 中，`pen = QPen(Qt.black)` 创建一支黑色画笔，用于横纵坐标轴，在纵轴上端添加“↑”箭头（说明越往上越重要）；在横轴右端添加“→”箭头（说明越往右越紧急）。在横轴右侧放上标签“紧急程度”；在纵轴上方放上标签“重要程度”。在每个象限中放置类别标签，分别使用红、绿、蓝、黄表示不同类别。

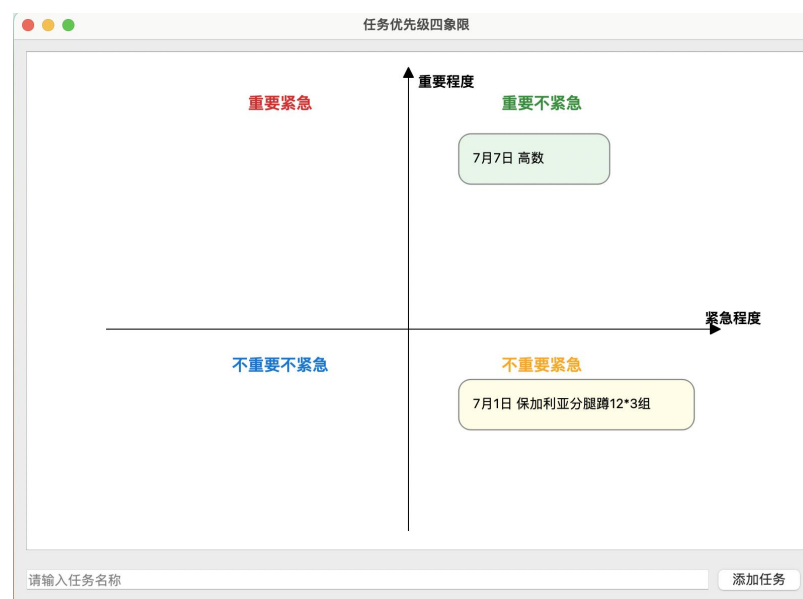
```
self.scene.addLine(0, -250, 0, 200, pen)
self.scene.addLine(-300, 0, 300, 0, pen)

self.add_arrow(0, -260, direction='up')
self.add_arrow(310, 0, direction='right')

self.add_label(250, -25, '<b>紧急程度</b>', ■ '#000000', size=14)
self.add_label(-35, -260, '<b>重要程度</b>', ■ '#000000', size=14)

self.add_label(-200, -240, '<b style="font-size:16px;">重要紧急</b>', ■ '#D32F2F')
self.add_label(60, -240, '<b style="font-size:16px;">重要不紧急</b>', ■ '#388E3C')
self.add_label(-200, 20, '<b style="font-size:16px;">不重要不紧急</b>', ■ '#1976D2')
self.add_label(60, 20, '<b style="font-size:16px;">不重要紧急</b>', ■ '#F9A825')
```

主界面显示为：

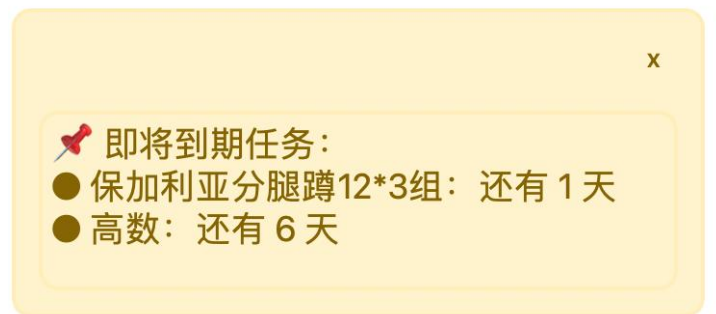


(3) 在任务管理上，通过 `add_task()` 在象限图中添加任务框，显示任务 `ddl` 日期和内容，任务框宽度自适应文本宽度，根据不同象限给任务框设置与标题相对应的颜色，明确清晰，且任务框设置为可拖动。任务根据离 `ddl` 的日期和任务种类的不同，被分在不同的象限。设置 `load_tasks_from_json()` 从日历输入的日程中读入任务；同时 `handle_add_task()` 能够在任务管理界面直接添加新任务。

```
load_tasks_from_json() → add_task() ← handle_add_task()
                        ▲
                        |
                    显示任务框（带DDL + 任务名）
```

(4) 构造函数将任务安排页面初始化，和日历部分保持相同的风格。使用 `QStackedLayout` 实现多个页面的切换，使用 `QLabel` 显示提醒文字，`top_bar` 使关闭按钮所在的横向栏右对

齐。在任务安排界面添加任务提醒窗口，在距离 ddl 还有 7 天时提醒开始任务，在距离 ddl 还有 1 天时提醒尽快提交任务，窗口设置退出键，在用户看到提示之后可选择将其删除方便其他任务的管理。



(5) 反思与总结:

任务管理实现了直观的可视化设计，并且使用颜色心理学原理：红色(紧急)、绿色(重要)、蓝色(平静)、黄色(注意)，设置了自适应大小和颜色的任务卡片；同时拖拽功能让用户自由组织任务，响应式布局自动避免任务重叠。任务管理方便灵活，支持手动添加任务、支持 JSON 文件导入、日期信息自动整合到任务显示。

项目可改进的地方：添加任务完成状态标记，对已完成任务进行可视化处理。可进一步引入日历插件，进行周期任务管理。同时可增加统计任务信息的面板，使用户对自己的任务组成有更多的认识。

三，健康系统部分

健康系统部分主要目标是通过大家的体测结果帮助大家了解自己的身体各个部位的健康状况，同时给出锻炼建议，并且可以一件添加到日历的日程安排之中。主要实现的界面如下：

(1) 健康系统主界面

同日历部分，健康系统部分的主界面采取和日历部分主界面相近的设计，保证大家打开健康系统的时候仍然可以看到日历的安排，可以手动修改日程。主要实现为在 `UI_Health.py` 中的 `Ui_Halendar` 类实现界面 UI，`HealthPage` 类中实现界面页面，在 `HealthWindow` 类中实现健康窗口。同时更改了界面三个按钮的设置：

```
self.btn_calendar = QtWidgets.QPushButton("体测系统")
self.btn_tasks = QtWidgets.QPushButton("健康建议")
self.btn_health = QtWidgets.QPushButton("返回")
```

添加体测系统，健康建议以及返回按钮，以实现后续功能。

(2) 体测系统

体测系统通过 `Ui_health.py` 中的 `TestWindow` 类实现。具体来说，该窗口功能是记录用户输入的体测数据，样式如下：

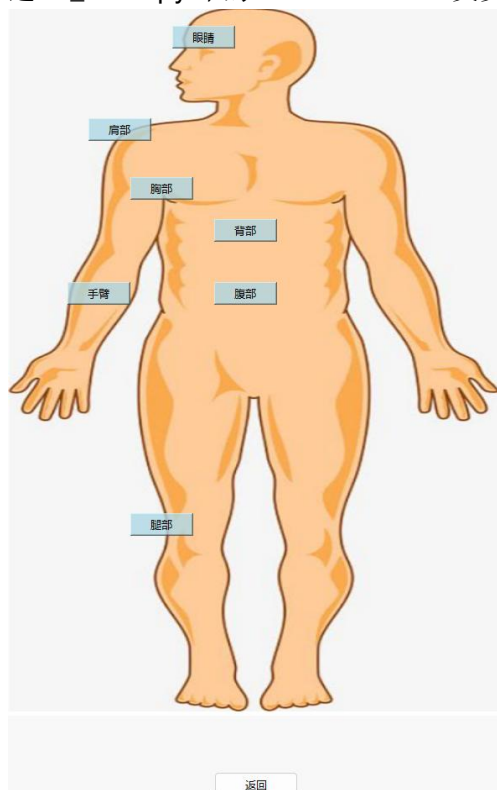
性别	<input checked="" type="radio"/> 男 <input type="radio"/> 女
身高 (cm)	180
体重 (kg)	75
肺活量 (ml)	6000
引体向上 (个)	10
50米跑 (秒)	7
1000米跑 (分:秒)	4:00
坐位体前屈 (cm)	10
左眼视力	1
右眼视力	1

返回
保存数据

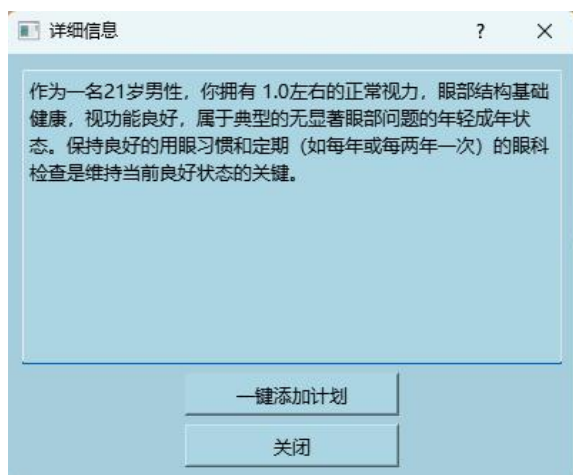
将用户输入的数据保存到 `health_data.csv` 文件中进行存储。如果用户没有输入则创建这个文件等待用户输入。打开或者关闭程序不会影响内容的存储，每次打开界面首先会先从文件中读取用户存储过的历史数据。该数据存储以待健康建议系统使用。该窗口也实现了保存成功提示功能，窗口拖动功能，验证输入数据有效性功能，在代码对应位置有详细注释，不在此一一赘述。

(3) 健康建议系统

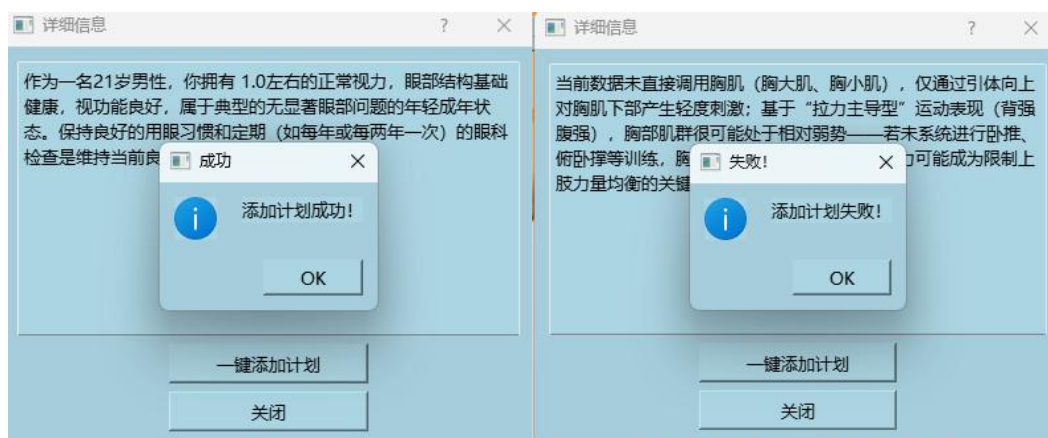
该系统旨在从 `health_data.csv` 中读取用户的体测信息，分析用户的身体各个部位的机能状况，从而给出用户在日常的锻炼建议，并且可以快速添加到日程安排之中。首先，该部分通过 `UI_Health.py` 中的 `AdviceWindow` 类实现了一个人体图为背景的界面，具体样式如下：



每一个部位添加了对应按钮，通过 `CustomButton` 类实现。点开包含对应部位的健康详情分析，通过 `CustomTextWindow` 实现，具体如下：



详细信息记录在 **health** 文件夹的相关文件里。例如眼睛记录在 **eye.txt** 中。一件添加计划按钮则通过 **UI_calendar.py** 中的 **save_event** 函数直接将推荐的锻炼计划添加到日程中，示例如下：



锻炼计划记录在 **advice** 文件夹下，名称同 **health** 文件夹下对各部位的命名。如果没有锻炼计划（AI 无法通过给出的体测信息进行相关部位的分析），则添加失败。在添加计划成功之后，可以在日历界面查到对应日程，该日程也记录在了 **calendar_events.json** 文件中：



(4) 不足之处

我们初始的目标是引入 **AI** 的 **api** 帮我们通过体测信息进行锻炼计划的规划，但是由于给出合适的提示词让 **AI** 能够输出符合我们要求的简介的介绍过于困难，我们在优化提示词上耗费了太多的时间，同时也是没有找到合适的免费 **API**，我们最终选择给出一份示例 **AI** 生成

的示例。同时我们发现，体测数据过于片面，AI 无法给出例如胸部的健康状况与分析，我们可能需要用户提供更多的数据以供分析。