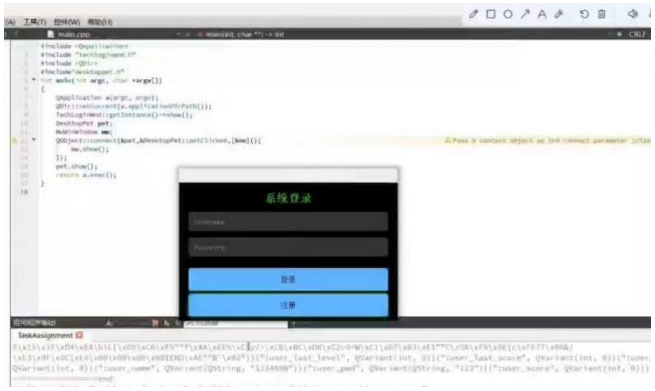
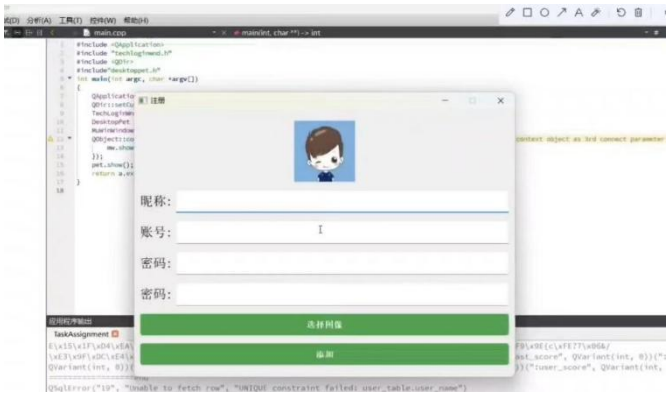
专注项目设计说明文档

组名：QT 快乐小分队　　组员：董钰　周虹好 于越

一、程序功能介绍：

1. 登录界面：输入账号密码可进行登录或退出。不同的账号对应的头像不同。

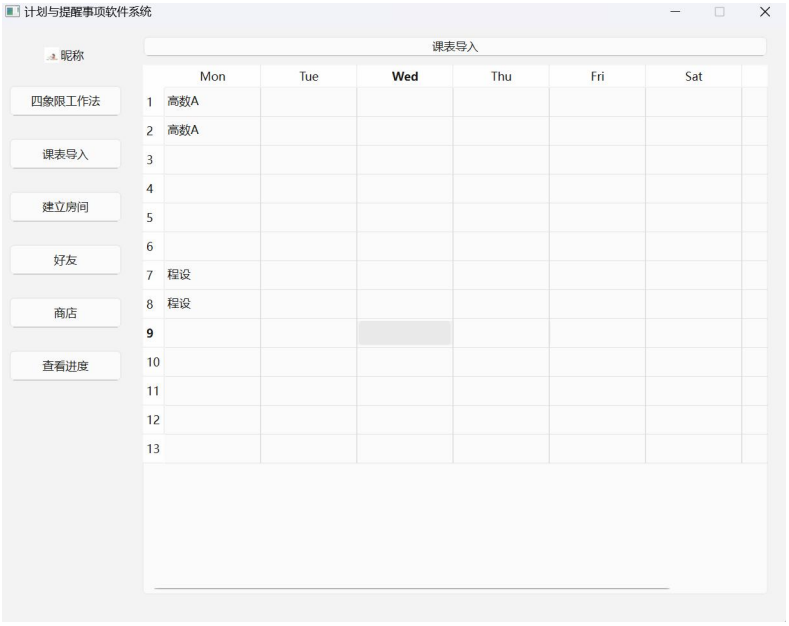




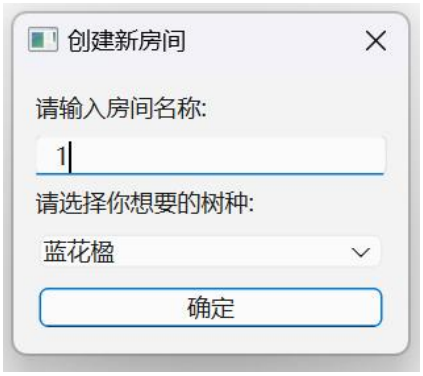
2. 任务四象限工作法：任务分为紧急且重要、紧急但不重要、重要但不紧急、不紧急且不重要四类，可随意添加修改删减，标记是否完成，截止时间，内容和分数设定。

3. 查看任务进度：点击查看进度可以看到四类任务的分别进度和总进度，以及最终积分。

4. 课表导入：可以将课程安排双击格子填入，可以随意修改安排计划。



5. 建立专注房间：要专注计时可以点击右上角的专注计时，设计房间名称和要种的树种，树种分 18 类，可以任意选择。



6. 专注结束后弹出“恭喜完成”的窗口，并且增加相应的积分。
7. 查看好友排行榜。

二、项目各模块与类设计细节:

项目中的类包含以下:

HEADERS += \
GlobalSetting.h \
GlobalTimerTask.h \
ModelManager.h \
MuCursorPosCalculator.h \
MuCustomWindow.h \
MuFramelessHelper.h \
MuFramelessHelperPrivate.h \
MuShadowWidget.h \
MuShadowWindow.h \
MuTitleBar.h \
MuWidgetData.h \
MuWinDWMAPI.h \
MuWinTitlebar.h \
MuWinWindow.h \
Singleholder.h \
SqlObject.h \
abstracttablemodel.h \
task_table.h \
task_tableRepository.h \
tasktablemodel.h \
taskwnd.h \
techloginwnd.h \
user_table.h \
user_tableRepository.h \
userinfownd.h \
userlistwnd.h \
userregwnd.h \
userscorelabel.h \

usertablemodel.h

对应的 sources 文件里各个类实现如下：

1. GlobalSetting.cpp 中包含的类和函数如下：

```
class GlobalSetting : public QObject,public PH::SingleHolder<GlobalSetting>
{
    Q_OBJECT
public:

    ///设备编号
   // Q_INVOKABLE QString getDeviceId(const QString& defValue="F34A4123456");
private:
    friend class PH::SingleHolder<GlobalSetting>;
    explicit GlobalSetting(QObject *parent = nullptr);
    QSharedPointer<QSettings> _ptrSetting;
    template<typename T>
    T getValue(const T& value,const QString& key,const QString& group="");
    void setValue(const QVariant& value,const QString& key,const QString& group="");
    QString getValue(const QString& key,const QString& group="");
    std::mutex _mutex;
};
```

2. GlobalTimerTask.cpp 中包含的类和函数如下，负责实现系统时间更新任务：

```
class GlobalTimerTask : public QThread,public PH::SingleHolder<GlobalTimerTask>
{
    Q_OBJECT
public:
    void run()override;
    void stop();
Q_SIGNALS:
    void updateSystemTime();
    void checkUnJinJiZhongYao();
    void updateUserLevel();
private:
    explicit GlobalTimerTask(QObject *parent = nullptr);
    friend class PH::SingleHolder<GlobalTimerTask>;
    class Private;
    QSharedPointer<Private> _p;
};
```

3. MuCursorPosCalculator.cpp：负责实现鼠标的识别
4. Mucustomwindow.cpp，MuFramelessHelper.cpp：负责实现 Aero 部分特效的窗体
5. MuShadowWidget.cpp：在 Qt 中创建一个带有阴影效果的窗口。通过使用 MuSkin9GridImage 类来处理九宫格图像，并在 MuShadowWidget 类中绘制阴影边框，实现了一个可以调整大小的带有阴影效果的窗口。
6. MuTitleBar.cpp：创建一个自定义的窗口标题栏，包含窗口图标、标题、最小化按钮、最大化/还原按钮和关闭按钮，还可以添加自定义内容。它继承自 QWidget，通过处理相关的事件和信号，实现窗口的最小化、最大化、还原和关闭功能，并支持调整标题栏高度和自定义内容。
7. MuWidgetData.cpp：处理无边框窗口的鼠标事件和窗口交互，包括窗口的移动、调整大小以及阴影效果的处理。它是一个帮助类，通过捕捉和处理鼠标事件，实现窗口的自定义行为。
8. MuWinDWMAPI.cpp：这个类 MuWinDwmapi 是一个单例模式类，用于与 Windows DWM (Desktop Window Manager) API 交互，主要功能是管理和控制 Windows 窗口的 Aero Glass 效果和其他 DWM 特性。该类封装了多个与 DWM 相关的操作，可以检查和控制 Aero Glass 效果、启用模糊效果、扩展玻璃效果到客户区域、启

用/禁用窗口动画等。

9. MuWinTitlebar.cpp:自定义的标题栏类，用于创建具有最小化、最大化/还原、关闭功能的标题栏。该类继承自 QWidget，并且可以在 Qt 应用程序中替代默认的窗口标题栏，实现自定义外观和行为。
10. MuWinWindow.cpp:MuWinWindow 类是一个自定义的窗口类，继承自 QWidget，用于创建一个带有自定义功能和布局的窗口。该类特别为 Windows 操作系统设计，提供了集成用户信息、用户分数显示、任务管理和用户列表等功能。
11. Singleholder.cpp: SingleHolder 是一个模板类，用于实现单例模式（Singleton Pattern）。单例模式确保一个类只有一个实例，并提供一个全局访问点。
    SingleHolder 使用模板技术，可以使任何类型的类成为单例。
    单例模式:通过模板类 SingleHolder，可以使任何类型的类成为单例。
    线程安全:使用 std::mutex 和 std::unique_lock 实现线程安全，确保在多线程环境下单例实例的唯一性。
    懒汉式单例:单例实例在第一次调用 getInstance 时创建，使用懒汉式加载避免程序启动时就占用资源。
    内存管理:使用内嵌类 Deletor 确保在程序结束时正确释放单例实例的内存，避免内存泄漏。
12. SqlObject.cpp: 用于管理数据库操作的 Qt 对象。它提供了一系列方法和属性来处理数据库的连接、查询和操作。这个类封装了常见的数据库操作，使得在 Qt 应用程序中使用数据库变得更加方便和直观。
13. task_table.cpp: 继承自 kcdz::SqlObject。它使用 Qt 的 Q_PROPERTY 宏来声明属性，并通过信号和槽机制实现数据绑定和变化通知。这使得它适合在使用 Qt 的项目中管理任务数据，并与 Qt 的其他组件进行数据交互。
14. task_tableRepository.cpp: 是一个模型管理器，用于管理 task_table 实例和相应的模型 TaskTableModel。它继承自 QObject 和 AbstractModelMgr，通过提供插入和过滤功能来管理任务表数据。
15. tasktablemodel.cpp: 是一个自定义的表格模型类，继承自 AbstractTableModel，用于在 Qt 的模型-视图框架中管理和显示 task_table 数据。该类实现了数据的获取和设置、行选择状态的管理，并提供了删除选中行的功能。
16. taskwnd.cpp: 是一个自定义的窗口类，继承自 QWidget，用于展示和管理任务数据。它与 task_tableModelMgr 紧密结合，通过四个不同的 task_tableModelMgr 实例来分别管理不同优先级和重要性的任务，并提供相关的交互功能，如检查紧急重要任务和任务完成处理。
17. techloginwnd.cpp: 是一个带有阴影效果的登录窗口类，继承自 MuShadowWindow<QWidget> 并实现单例模式，通过继承 PH::SingleHolder<TechLoginWnd> 实现。该类还集成了用户信息更新和显示功能。
18. user_table.cpp: 定义了一个名为 user_table 的 C++ 类，它继承自 kcdz::SqlObject，并且使用了 Qt 框架的一些特性，比如 Q_OBJECT 和 Q_PROPERTY。下面我将详细解释这个类的功能和设计思路。
19. user_tableRepository.cpp: 于管理 user_table 数据模型的操作，确保数据操作的一致性和可靠性
20. userinfownd.cpp: 用于展示用户信息窗口。它继承自 QWidget，并且使用了 Qt 的信号和槽机制。
21. userlistwnd.cpp: 定义了一个名为 UserListWnd 的类，用于展示用户列表窗口。它继

承自 QWidget，并且使用了 Qt 的信号和槽机制。

22. userregwnd.cpp: 定义了一个名为 UserRegWnd 的类，用于用户注册窗口。它继承自 QWidget，并且使用了 Qt 的信号和槽机制。
23. userscorelabel.cpp: userScoreLabel 的类，用于显示和更新用户得分标签。它继承自 QWidget，并且使用了 Qt 的信号和槽机制。封装性：将用户得分的显示和更新封装在一个独立的窗口类中，便于管理和扩展。
    UI 集成：通过包含生成的 UI 文件，方便定义和操作用户界面元素。
    简单易用：提供了一个简单的接口 updateScore(int)，可以方便地更新用户得分显示。
24. usertablemodel.cpp: 用于表示用户表的数据模型。它继承自 AbstractTableModel，并重写了 data 方法。
    封装性：将用户表的数据表示封装在一个独立的模型类中，便于管理和扩展。
    模型/视图框架：通过继承和重写方法，实现了与 Qt 模型/视图框架的集成，提供了对数据的灵活访问和显示。
    简洁易用：提供了一个简单的接口来访问和显示用户表中的数据。

三、小组成员分工：

董钰：课表，数据库导入，登录界面创建，完善

周虹好：四象限，专注房间，pet，完善

于越：商店，好友，树种图片，说明文档

四、项目总结与反思：在该项目中，初次尝试体验使用 qt 完成项目创建，与实际生活学习相结合，用 c++尝试做一些有用的东西出来。下面是几点心得和反思：

1.QT 框架的强大与灵活：QT 框架提供了丰富的控件和强大的功能，使得开发过程变得更加高效。从基本的 UI 设计到复杂的网络通信，QT 都能提供强有力的支持。同时，QT 的跨平台特性也让我在开发过程中无需担心兼容性问题。

2.模块化编程的重要性：在开发过程中，我深刻体会到了模块化编程的优势。通过将功能拆分成不同的模块，不仅可以提高代码的可读性和可维护性，还能方便地进行单元测试和功能扩展。

3.团队协作的力量：在这个项目中，我与团队成员紧密合作，共同解决了许多技术难题。通过团队协作，我不仅提高了自己的技术能力，还学会了如何与他人有效沟通和协作。

反思：

1.代码质量的重要性：在开发过程中，我发现一些代码质量问题会严重影响程序的性能和稳定性。因此，我需要更加重视代码质量，遵循良好的编程规范和最佳实践，确保程序的健壮性和可维护性。

2.持续学习的必要性：随着技术的不断发展，我需要不断地学习和更新自己的知识库。通过参加技术交流会、阅读技术文档和博客等方式，我可以了解最新的技术动态和最佳实践，提高自己的技术水平和竞争力。