**“C++程序设计与训练”课程大作业**

**项目报告**

**项目名称：众包协作翻译平台**

**姓名：** 郑旸

**学号：** 2018011564

**班级：** 自86

**日期：** 2019/8/2

目 录

[1平台功能设计 3](file:///E:\C++大作业项目报告模板(2019).docx#_Toc12829525)

[1.1总体功能描述 3](file:///E:\C++大作业项目报告模板(2019).docx#_Toc12829526)

[1.2功能流程描述 3](file:///E:\C++大作业项目报告模板(2019).docx#_Toc12829527)

[2平台结构设计 4](file:///E:\C++大作业项目报告模板(2019).docx#_Toc12829528)

[3平台详细设计 4](file:///E:\C++大作业项目报告模板(2019).docx#_Toc12829529)

[3.1 类结构设计 4](file:///E:\C++大作业项目报告模板(2019).docx#_Toc12829530)

[3.2 数据库/文件结构设计 4](file:///E:\C++大作业项目报告模板(2019).docx#_Toc12829531)

[3.3 界面结构设计 4](file:///E:\C++大作业项目报告模板(2019).docx#_Toc12829532)

[3.4 关键设计思路 4](file:///E:\C++大作业项目报告模板(2019).docx#_Toc12829533)

[3.5 附加功能实现 5](file:///E:\C++大作业项目报告模板(2019).docx#_Toc12829534)

[4项目总结 5](file:///E:\C++大作业项目报告模板(2019).docx#_Toc12829535)

[5相关问题的说明 5](file:///E:\C++大作业项目报告模板(2019).docx#_Toc12829536)

# 1平台功能设计

该部分将具体描述软件的功能及目的

## 1.1总体功能描述

软件功能具体描述如下

|  |  |  |
| --- | --- | --- |
| **操作类型** | **功能名称** | **功能描述** |
| 用户基本信息操作 | 登录 | 通过输入用户名和密码进行登录 |
| 注册 | 输入合法用户名以及密码进行注册 |
| 注销  （注销菜单） | 用户登出，自动更新数据 |
| 基本信息维护  （用户基本信息维护菜单） | 维护用户邮箱、用户名、密码等信息 |
| 资质信息更新、实名认证  （资质认证菜单） | 更新用户资质信息，上传资质证明图片等 |
| 充值  （充值菜单） | 进行充值 |
| 用户任务操作 | 查看所有任务 | 点击查询平台上所有任务的信息 |
| 查找任务 | 标题栏输入任务名进行任务检索 |
| 分别查看招募中、进行中任务并报名 | 用户可查看所有任务分类下的正在招募、正在进行的任务，对于正在招募中的任务，用户可以进行报名 |
| 查看用户发布的任务并进行任务操作 | 用户可以在“我发布的任务”中查看任务，并点击进行任务操作，如负责人指定、下载最终译文等 |
| 查看用户负责的任务并进行任务操作 | 用户可以在“我负责的任务”中查看任务，并点击进行任务操作，如译者指定、对译者评价、上传最终译文等 |
| 查看用户翻译的任务并进行任务操作 | 用户可以在“我翻译的任务”中查看任务，并点击进行任务操作，如暂存译稿、查看负责人评价等 |
| 发布任务 | 用户可以点击“发布任务”进行任务发布，上传详细任务信息，成为任务发布者 |
| 管理员 | 登录  （系统特定管理员ID） | 输入平台特定标识符以管理员身份登录 |
|  | 审核用户提交的资质证明并更改用户积分  （用户认证菜单） | 管理员可以查看各个用户上传的资质信息并对其积分进行增减 |

## 1.2功能流程描述

结合上一节的总体功能和翻译任务的进行过程的不同状态描述对应的功能流程。

这一部分在附件《Z众包翻译详细流程介绍》中给出。

# 2平台结构设计

这是进行复杂软件开发的第二步，即概要设计。此部分需要说清楚整个软件系统包含哪些模块（或功能部件），模块之间的关系和是怎样的；包含哪些主要的类，类之间的关系是怎样的（可以用UML类图或对象图表达）。

（此部分的子标题和结构自行拟定。）

## 2.1 主要结构

本程序包括任务类、用户类、数据库类、报名平台类、文本交互类、消息通知类、资质证明类，其中任务类与数据库类是本程序的核心，各个类间关系如下所示;

## 本程序的User类设计为一大亮点，由于用户的任务身份特殊，任务在不同任务中拥有不同身份，因此将不同任务对象作为User的成员变量，进而进行任务操作。

SignUp

AssignmentPublisher

certification

User

AssignmentTranslator

MessageInformBox

AssignmentManager

## 任务类， 由抽象基类派生出三个任务子类，充分实现继承与多态

AssignmentPublisher

AssignmentTranslator

AssignmentManager

Assignment

1. 数据库类

详细的数据库设计是本程序的亮点之一，同时也是最大难点之一，在程序中占据千行代码。数据库类是底层数据库与上层类对象沟通的桥梁。由抽象基类派生出四个面向对象的数据库类。

DataBaseAssignment

（针对任务类）

DataBaseUser

(针对用户基本数据)

DataBase

DataBasePicture

(针对图片类)

DataBaseCertification

（针对用户资质类）

## 2.2 UML类图

用户类：(展示部分重要成员及函数)

|  |
| --- |
| User |
| -m\_sUserName: QString  -m\_sId: QString  -m\_sPassWord: QString  -m\_sEmail: QString  -m\_iScore: int  -m\_fMoney: float  -m\_pixPicture: QPixmap |
| +**User**()  +**User**(username:QString, password: QString, score: int, money:float,id:QString ,email:QString)  ~**User()**  +all\_assignment: QVector<Assignment>  +m\_userAsManager: QVector<AssignmentManager>  +m\_userAsPublisher:QVector<AssignmentPublisher>  +m\_userAsTrans: QVector<AssignmentTranslator>  +m\_certification: certification  +**signUpForManager(**assignmentName: QString**)** :bool  +**signUpForTrans(**assignmentName: QString**)** :bool  +**signUpForPublisher(**assignmentName: QString**)** :bool  +**getAllAssignment**(): void  +**getUserAssignment**():void |

任务基类(STATUS为任务状态枚举，LANGUAGE为翻译类型枚举)

|  |
| --- |
| Assignment |
| #m\_sAssignmentName:QString  #m\_sUserId:QString  #m\_sPublisherId:QString  #m\_slTranslorID:QStringList  #m\_sAssignmentText:QString  #m\_sDescription  #m\_eStatus:STATUS  #m\_eTranslationForm:LANGUAGE  #m\_fReward:float  #m\_pixPictur:QPixmap  #m\_deadline:QDateTime  #m\_currentDDL:QDateTime  #m\_keyWords:QStringList |
| +Assignment()  +Assignment(assignmentName: QString, publisher\_id:QString,managerId: QString,  translatorID:QStringList, description:QString, status:int, form:int, reward:float, assignmentText:QString, keywords:QStringList, Pix:pixmap)  +Assignment(assignmentName:QString)  +~Assignment()  +setDDL():bool  +setCurrentDDL():void  +uploadText():bool  +downloadText():bool  +getPublisherID():QString  +getManagerID():QString  +getTranslators():QStringList  +getKeywors():QStringList  +getPix():QPixmap |

任务子类(TextInteraction为文本交互类，SignUpInteraction为报名平台类)

|  |
| --- |
| AssignmentPublisher |
| -m\_textManagement:TextInteraction  -m\_finalText:TextInteraction  -m\_managerSignup:QVector<SignUpInteraction> |
| + AssignmentPublisher ()  +AssignmentPublisher (id:QString, assignmentName:QString)  +AssignmentPublisher (assignmentName:QString)  +~AssignmentPublisher ()  +setDDL():bool  +setCurrentDDL():void  +setTranslationForm(form:int):void  +uploadText():bool  +uploadPictureMessage(filename:QString):void  +downloadText():bool  +payReward():float  +getFilnalTranslation():QString  +getSignUp():QVector<SignUpInteraction> |

|  |
| --- |
| AssignmentManager |
| -m\_finalText:QString  -m\_review:QMap<QString, TextInteraction>  -m\_subAssignment QMap<QString, TextInteraction>  -m\_subTranslated: QMap<QString, TextInteraction>  -m\_tranlatorsrSignup:QVector<SignUpInteraction> |
| + AssignmentManager ()  +AssignmentMnager(assignmentName:QString)  +~AssignmentManager()  +setDDL():bool  +setCurrentDDL():void  +setTranslationForm(form:int):void  +uploadText():bool  +downloadText():bool  +payReward():float  +getFilnalTranslation():QString  +chooseTranslator(tranID:QString)  +giveReview(tranID:QString, review:QString):void  +setStatusProcess():void  +serStatusDone():void |
| AssignmentTranslator |
| -m\_subAssignment:TextInteraction  -m\_subTranslatedText:TextInteraction  -m\_managerReview:TextInteraction |
| + AssignmentTranslator ()  +AssignmentTranslator (tranID:QString, assignmentName:QString)  +~AssignmentTranslator ()  +uploadText(filePath:QString):bool  +editTextOnline(text:QString):void |

数据库类

|  |
| --- |
| DataBase |
| << static >>#m\_sDataBase: QSqlDatabase  << static >>#m\_sDatabaseName: QString  << static >>#openDataBase(): QSqlDataBase  #m\_tableName:QString  #m\_table\_parameters:QString |
| +deleteData(parameter:QString, Value:QString):virtual bool  +search(id:QString, parameter:QString):virtual bool  +updateData(QString, value:QString, parametor:QString):virtual bool  +updateInt(QString, value:int, parametor:QString):virtual bool  +updatefloat(QString, value:float, parametor:QString):virtual bool |

任务数据库（由抽象基类派生）

|  |
| --- |
| DataBaseAssignment |
| -updateData(AssignmentName: QString, value:QString, parametor:QString): bool  -updateInt(AssignmentName: QString, value:int, parametor:QString): bool  -updatefloat(AssignmentName: QString, value:float, parametor:QString): bool |
| +search(assignmentName:QString, parameter:QString): bool  +insertAssignment(assignment:Assignment):bool  +updateAssignment(assignment:Assignment):bool  +getAssignment(assignment:Assignment): Assignment  +getAllAssignment(assignment:Assignment): QVector<Assignment>  +insertAssignmentText(text: TextInteraction):bool  +updateAssignmentText(text: TextInteraction):bool  +getAssignmentText(assignmentName: QString):TextInteraction  +insertSignUp(SignUp:SignUpInteraction):bool  +upDateSignUp(SignUp:SignUpInteraction):bool  +getSignUp(assignmentName:QString,  selectorID:QSting, form:SIGNUP):QVector<SignUpinteraction> |

用户数据库、图片数据库等与任务数据库类似。

# 3平台详细设计

这是进行复杂软件开发的第三步，即详细设计。此部分需要说清楚具体的模块是如何设计和实现的，类是如何具体实现的，类中的重点方法和算法是如何设计和实现的，数据库（或文件存储）的结构是如何设计的，界面是如何设计的，容错功能是如何设计的，以及各种设计思路等等，可以用图形或图表的方式加以说明。详细设计是编写代码前的最后一步设计工作，因而需要在需求分析和概要设计的基础上，说清楚**所有需要在编码前明确的设计事项**。

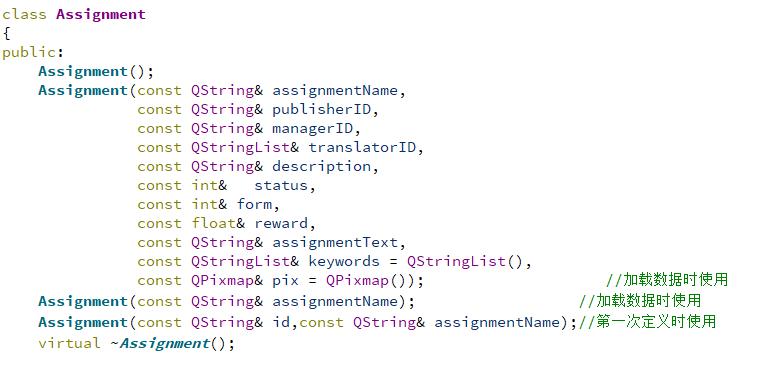
**注意：所有实现的加分功能需要单独详细说明。**

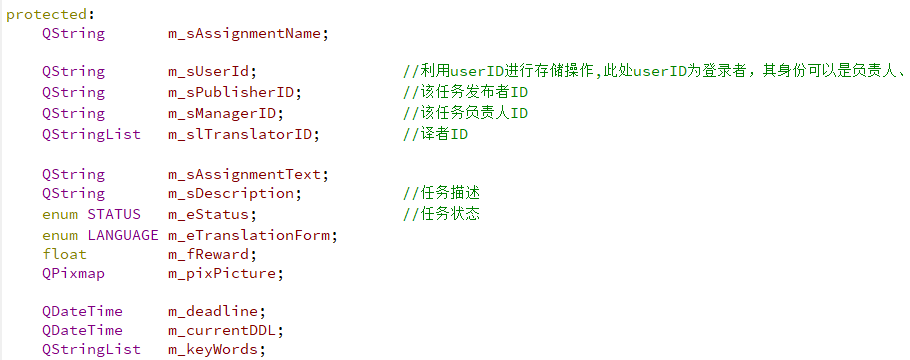
（此部分的子标题和结构可自行拟定，如下供参考。）

## 类结构设计

1. 任务类设计

Assignment基类设计：assignment基类包含了任务的基本数据及属性





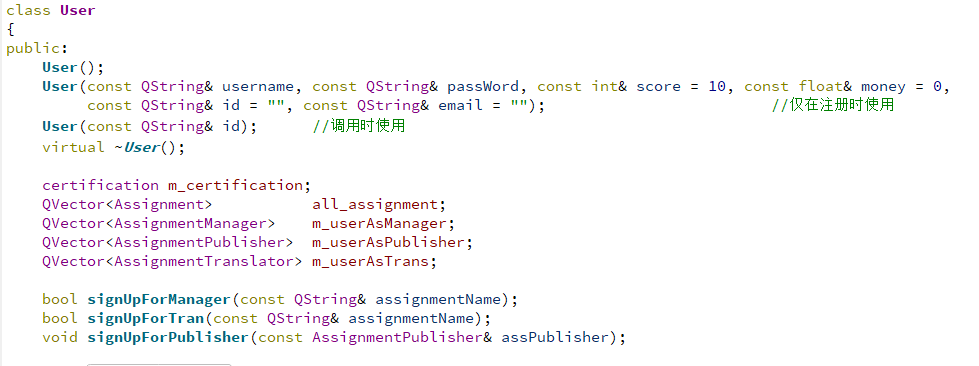
由Assignment派生出AssigmentPublisher、AssignmentManager、AssignmentTranslator三类，对基类虚函数重写并新增子类特有接口，派生类功能主要为设置任务属性（如上传任务信息、招募选拔等），而没有多余的私有数据，在实现多态性的同时简洁明了，利于存储。例如发布者类：



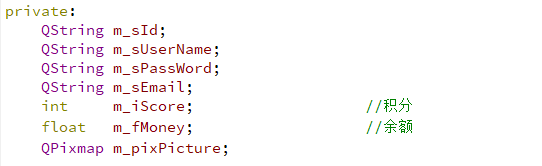
任务发布类增加接口以上传信息，同时以报名类对象为私有成员，便于发布者选择。任务负责类、任务翻译者类与之相似。因此加载子类数据时只需要确定基类信息以及报名信息、文本上传信息即可，从而减少储存开销。

1. 用户类

由于平台上的一个用户可以在不同任务中担任不同角色，因此对用户进行派生并不合适，本程序将任务类对象作为用户的成员，从而可以避免多重继承而实现用户对不同任务的不同行为。如下图所示，通过用户内部成员函数实现用户对不同任务的报名。



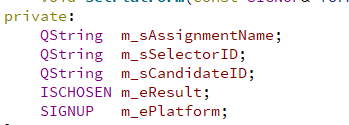
此外，用户拥有其基本数据，此处设置8中基本数据作为示例。

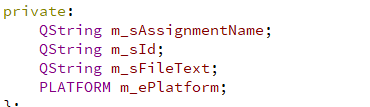


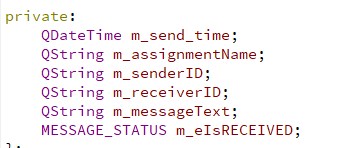


1. 信息交互类

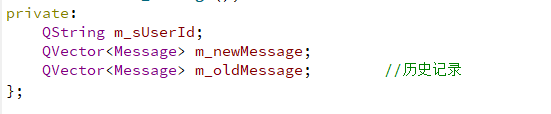
信息交互类是本程序的亮点之一，通过SignUpInteraction、TextInteraction、Message三类实现信息的传递与交互，信息交互类设计的核心思想是利用任务名、发送者ID、接收者ID等进行匹配，从而准确传递信息。

报名类的私有成员结构：

文本类的私有成员结构: 

Message的私有成员结构：

此外设立消息盒子类为特定用户存储信息：



1. 数据库类

面向对象设计的数据库类是本程序的亮点之一，本程序针对各不同数据结构在抽象数据库基类上派生出任务数据库子类、用户数据库子类等，在底层对数据存取进行封装，从而在使用时仅需调用数据库类就能实现对象的更新、删除、存取等功能。

基类数据库设计如下：



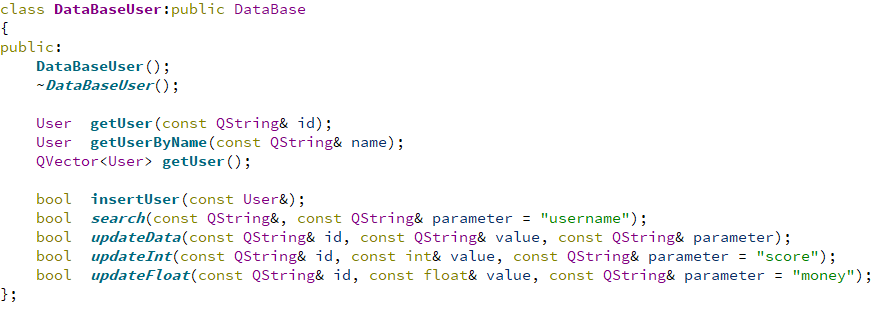
其中更新、查询等函数均作为虚函数在子类中重载，方便实现多态；m\_sDataBase作为静态成员，在程序运行过程中与数据库连接，m\_tableName、m\_table\_parameters为数据库表格名、数据库参数名。

任务类数据库设计：



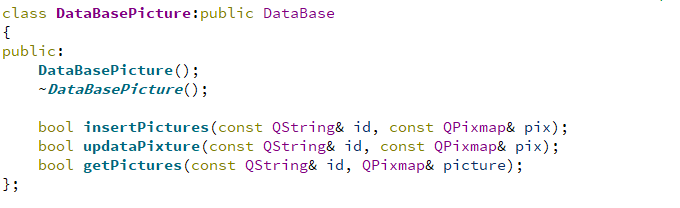
任务类数据库针对assignment类而设计，基类虚函数作为私有变量重写，确保不在外部被调用，仅用于内部数据操作

用户数据库类设计：



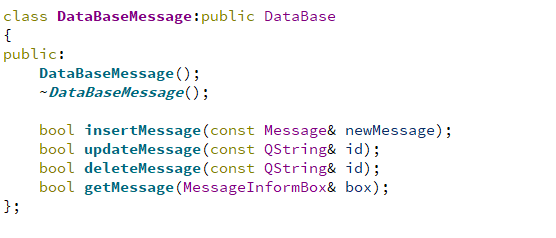
设计多个接口获取用户对象，重写基类函数，仅在User类数据变动时进行更新等操作，从而实现剪枝。

图片数据库类设计：



由于图片需要更大的存储空间，因此将图片单独存储，仅在有需要时调用，加载图片时传QPixmap引用对象，防止存取时增加不必要的开销。

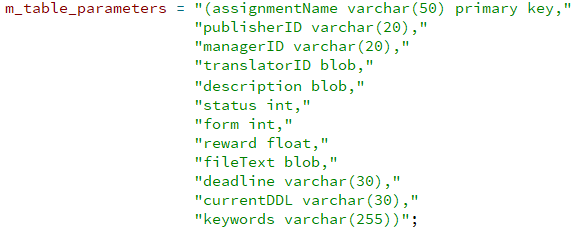
交互平台数据库类设计：



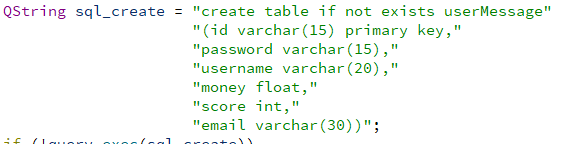
信息交互平台的设计是本程序的创新之处，而数据库是信息交互的基础，如消息通知、报名信息通知等均需要依赖于数据库，信息交互数据库类的设计相当于为不同用户间搭建消息沟通平台，从而实现不同用户间的沟通。

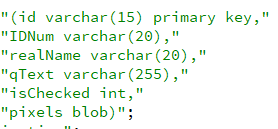
数据库内部存储结构设计：

对于任务类：以任务名为唯一标识符进行存取，储存任务基本属性



对于用户：用户id为唯一标识符，用户基本属性、用户资质认证分别存储在两个表中，方便用户与管理员进行操作，同时避免表格过于繁杂

（用户基本属性）

（用户语言资质信息）

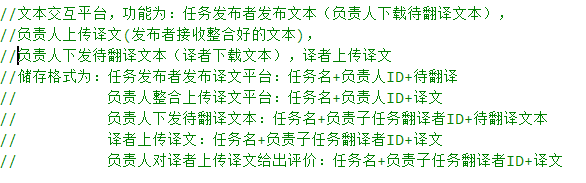
对于文本交互类对象：

对于报名类平台：



对于文本交互平台：

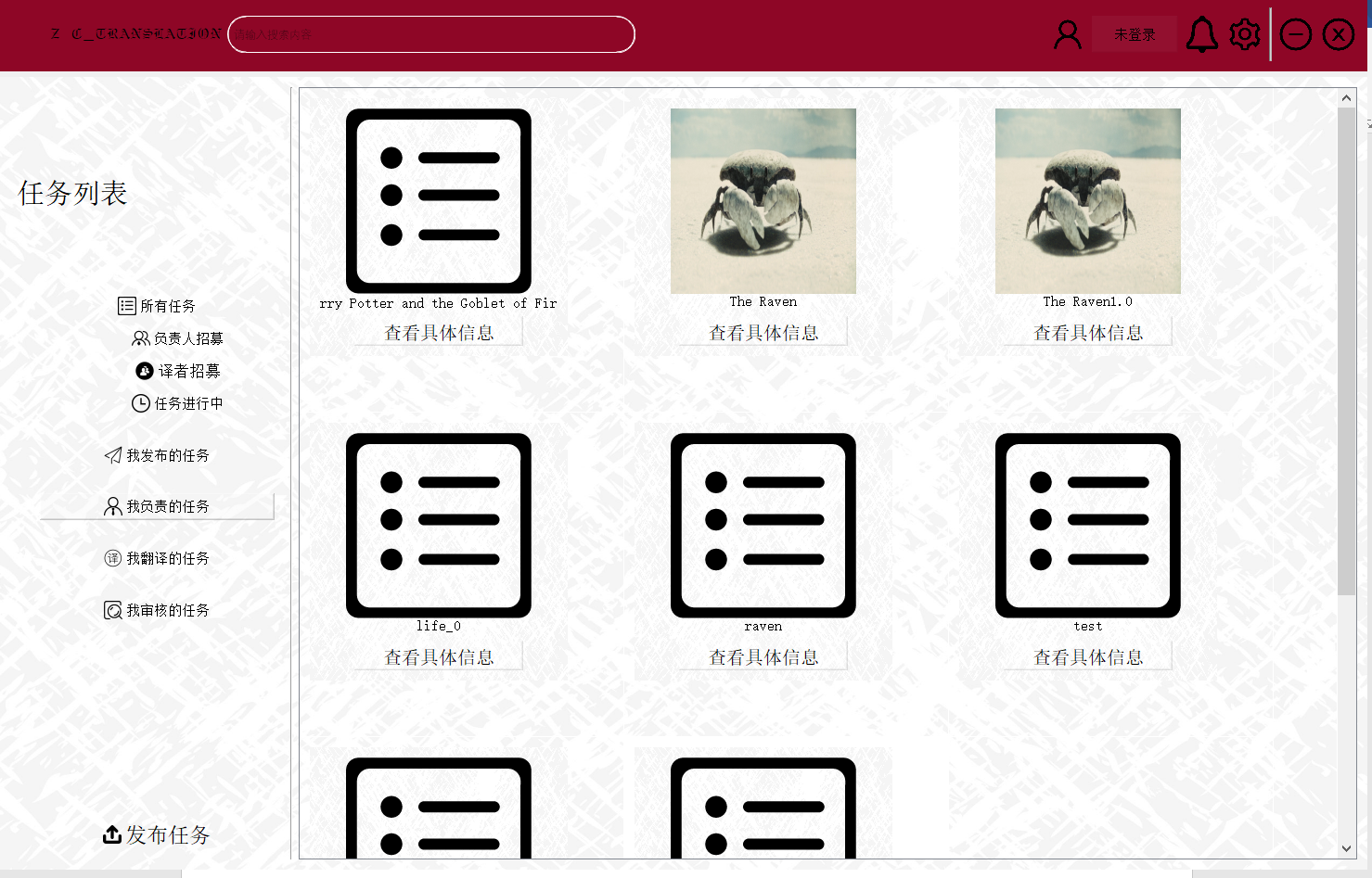
存储格式为任务名+该文本负责人+文本内容，注意到对于同一个任务会有多个翻译子任务的文本，且同一译者可能在不同任务中都有文本上传，因此该数据库不采用唯一标识符储存，而采用任务名+用户ID双重匹配机制



## 3.3 界面结构设计

描述界面的结构以及界面之间的跳转关系。

主界面设计是本程序的闪光之处，主界面模仿网易云音乐界面进行设计，用户的操作均在主界面基础上进行，符合用户使用习惯。



详细流程见附件，以下给出框图梳理：

进入主界面后：

主界面，更新我的任务

登陆界面

点击登录 登录

主界面

未登录

已登录 注册成功

语言资质认证界面

注册界面

消息盒子 菜单栏设置

消息盒子界面

基本信息维护、充值等界面

左上角搜索

任务详细信息界面

主界面

点击 点击发布任务

发布任务界面

所有任务/…招募/任务进行中

左上角搜索任务

点击查看具体信息

任务具体信息界面

任务具体信息界面

文本下载 报名

成功报名

下载文本界面

主界面

我负责的任务

我翻译的任务

我发布的任务

查看具体信息 查看具体信息 查看具体信息

译者界面

负责人界面

任务具体信息界面

查看评价、暂存译稿界面

分配任务

上传终稿

评价译稿

选择译者

选择负责人界面

下载界面

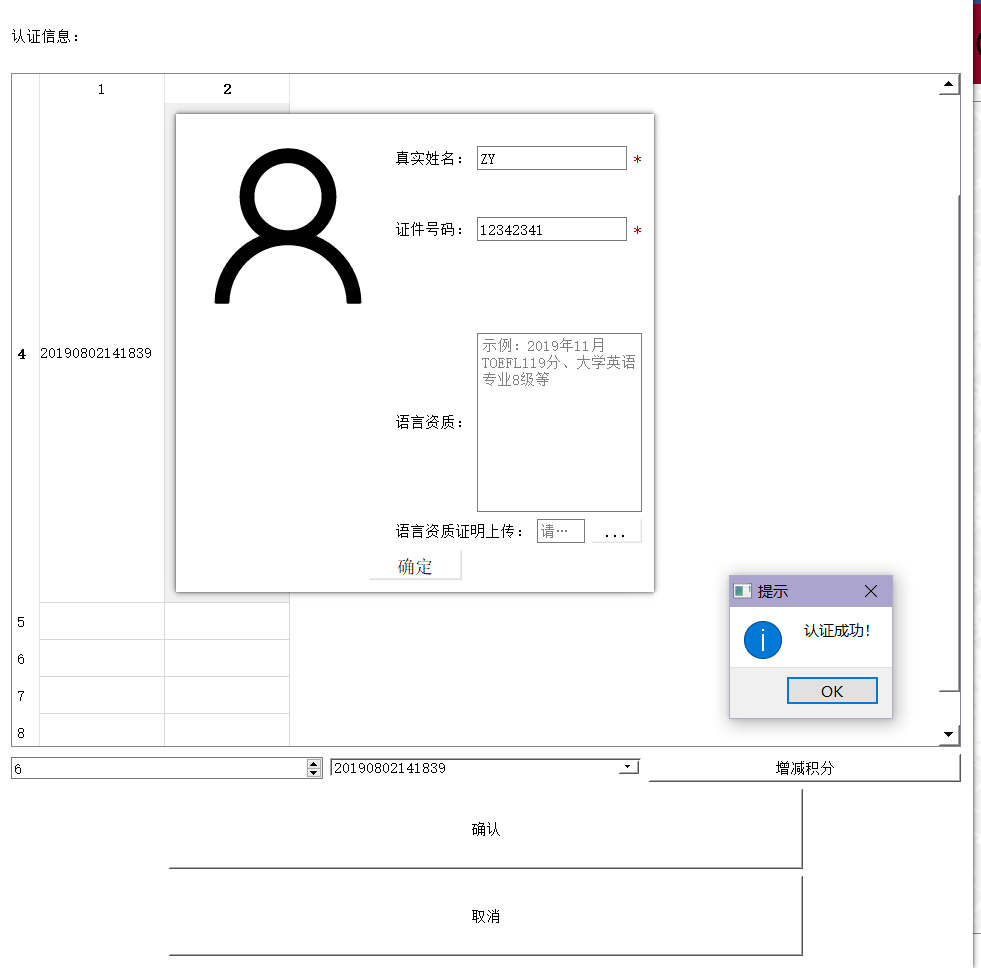
## 3.4 关键设计思路

众包翻译平台设计的难点很大程度在于信息的展示、交互以及用户的多重属性。前面已经介绍过，本程序利用任务名+多方用户ID进行匹配可以有效实现信息的传达与沟通，并且在用户类中将任务类对象作为成员，很好地解决了用户多重身份的问题，在代码层面可以完全实现众包翻译平台功能，而接下来的难点在于界面的设计，如何将报名信息、状态变更信息展示给用户是一大难点。由于报名人数未知、信息数量未知，故应当采用“动态展示”的思路，即动态创建信息名片以进行展示。本程序利用tableWidget内嵌子窗口的动态创建有效进行信息传递，进而可以实现复杂的信息界面跳转。内嵌子窗口的使用也是本程序的亮点之一。

## 3.5 附加功能实现

实现附加功能的具体描述。

1. 实现平台管理员，审核用户语言资质认证并对用户积分进行修改。平台管理员可以查看所有用户上传的有效资质证明，并修改相应积分。

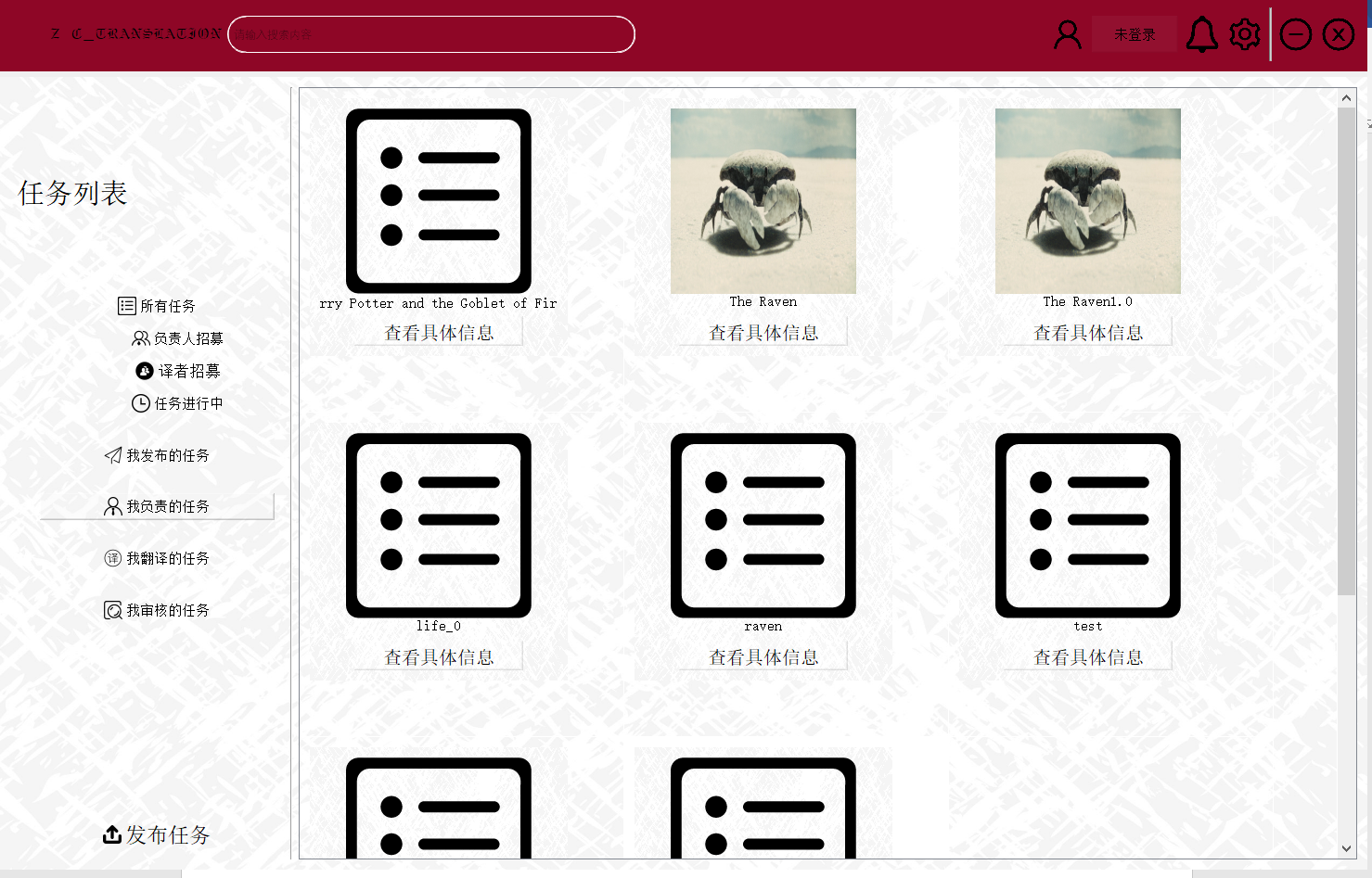


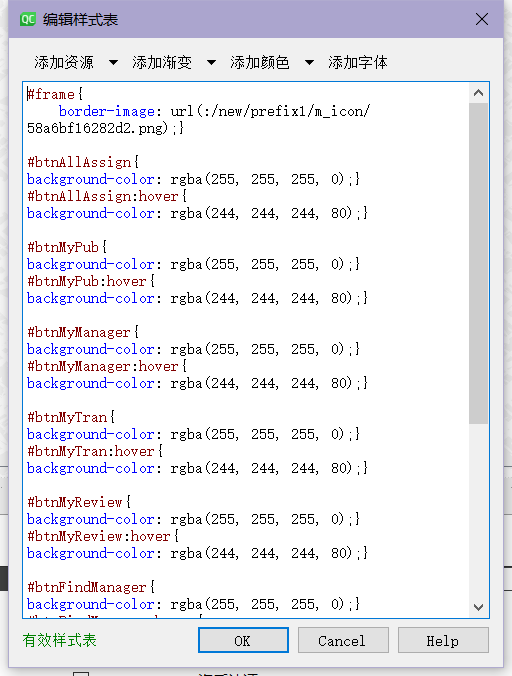
1. 任务搜索功能，通过搜索任务名直接跳转任务详细信息。
2. 个性化设计

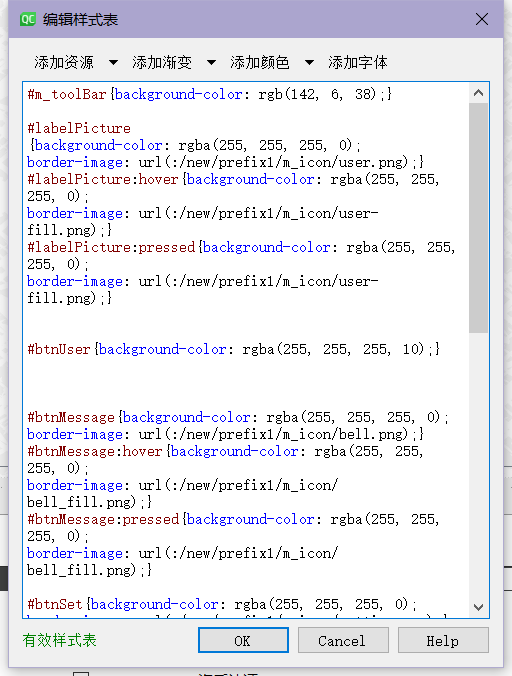
增加用户头像、任务封面等附加效果，符合用户使用习惯。

1. 窗体界面设计

界面利用qss进行设计，达到美化效果；自定义可拖动标题栏，符合用户使用习惯；标题栏与菜单结合。

（主界面效果）

（qss设置界面）

（qss设置标题栏）



（重写Widget与鼠标操作相关虚函数实现可拖动自定义标题栏）

# 4项目总结

总结设计、开发及调试工作中的问题及解决方法、难点、亮点和心得体会等。

1. 遇到的问题及解决方法

程序开发过程中遇到无数bug，开发过程确实万分艰难。首先我选择采用面向对象的数据库设计，而不是在程序开始全盘载入数据库，但是这种数据库的设计确实非常困难，需要对数据库操作十分了解，最终选择一条路走到黑，终于调通了1500行+的数据库类的代码。此外，我选择在类对象析构时将其更新或者插入数据库，这同样造成了很大的麻烦，因为这要求开发者对析构顺序即成员的生命周期十分清楚，要时时刻刻防止一些局部变量干扰到正确数据的插入。然而在庞大的工程里这是十分困难的，这样的操作方式一度带来了无法维护的bug, 最终我认真研究了变量的析构顺序并改动了大量代码，才确保数据的正确插入。

此外，对于界面而言，相信很多人会在一开始就把界面与代码混在一起写，但个人认为这十分面向过程，不符合面向对象的程序开发体验。因此我是在把所有类写好之后才开始做界面，希望通过界面连接所有类的接口，进而完成众包翻译的任务，但是这需要对qt有着极其熟练的掌握，并且程序的容错率要极高。如动态创建内嵌子窗口一度让我不知如何下手，界面构造过程也发现了很多重大bug,但最终还是解决了界面的问题。

1. 程序亮点

程序亮点诸如主界面设计、数据库设计、动态内嵌子窗口设计、个性化功能、搜索功能等前面已经提到，这里也不再赘述。

1. 心得体会

最大的体会就是压力。由于我开发该程序的顺序是从代码到界面，我的界面部分是放在最后做的，因此许多之前没有发现的bug都冒了出来，直到第三周周四晚（或是周五早）还出现了一个几乎无法维护的bug, 改了大量代码才解决。还有最恐怖的事情是周三晚（或是周四早）我的电脑突然黑屏，周四折腾大半天被告知主板损坏，而那个时候我的主界面才刚刚做完，最后借到学长电脑开始紧赶慢赶（所以后面的界面没有时间制作显得相当粗糙），确实是绝地求生，虽然本来向追求完美，最后很多预想的功能没办法完善还是有点遗憾，但能做到现在这样还写了这么长的报告我也知足了…….收获也蛮大的，至少现在觉得网易云音乐的界面也并不难实现，另外还掌握了数十种让exe crashed的操作。

# 5相关问题的说明