**“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：** 孙浩文

**学号：** 2018011702

**班级：** 自84

**日期：**  2019.8.2

目 录

[1平台功能设计 3](#_Toc12829525)

[1.1总体功能描述 3](#_Toc12829526)

[1.2功能流程描述 3](#_Toc12829527)

[2平台结构设计 5](#_Toc12829528)

[3平台详细设计 6](#_Toc12829529)

[3.1 类结构设计 6](#_Toc12829530)

[3.2 数据库/文件结构设计 1](#_Toc12829531)5

[3.3 界面结构设计 1](#_Toc12829532)5

[3.4 关键设计思路 1](#_Toc12829533)7

[3.5 附加功能实现 1](#_Toc12829534)8

[4项目总结 1](#_Toc12829535)9

# 1平台功能设计

## 1.1总体功能描述

本项目实现了一套众包协作翻译平台的体系。利用该系统，用户可以在平台上发布任务（成为发布者）、申请成为负责人、申请成为译者。在这一系统中，发布者可以发布任务并选择负责人，负责人可以选择译者分配并验收任务，译者可以完成翻译并通过审核。本平台还拥有消息盒子功能加以提醒。

## 1.2功能流程描述

**1.2.1 登陆/注册流程**

程序开始运行即进入登陆界面。用户可通过已有用户名密码进行登陆。用户名空白或用户名与密码不匹配即无法登陆。

新用户注册时需要完成包含手机号、身份证号、国家等的信息，并确认密码，两次密码一致、手机号符合格式方可登陆。登陆后进入主界面。

**1.2.2 主界面功能介绍**

主界面拥有“任务类型”、“我参与的”、“用户信息”三个板块。

任务类型板块中是该用户报名参加任务的地方。用户通过点击不同按钮可以选择成为任务发布者、申请成为负责人、申请成为译者。用户还可以点击文字按钮查看当前和历史出现的所有任务。

我参与的板块中显示的是与我相关的所有任务，三个表中分别会列出“我”作为发布者、负责人以及翻译者参与任务的编号和状态。双击该行即可具体进入任务操作（对应阶段的对应操作详见之后的叙述）。下方拥有刷新按钮与消息盒子按钮，分别用于更新当前页面的信息与查看消息盒子。

用户信息板块中呈现出当前登陆用户的所有信息，包括用户注册时的基本信息，用户的当前积分与账户余额。通过点击修改信息按钮可进入修改信息界面，通过点击充值按钮可进入充值界面。

**1.2.3 发布者流程**

**发布任务** 在点击主界面“任务类型”板块中的发布者按钮后即弹出发起任务窗口，用户需完整并规范填写表格中的内容信息，点击发布后即成为该任务的发布者，同时任务进入“招募负责人”状态。

**查看并选择负责人** 在任务处于“招募负责人”状态时，发布者可在“我参与的”板块中找到自己作为发布者的该任务，双击该行即可查看已经报名的候选负责人的用户名与积分，双击该行即可选择负责人，双击后任务立即进入“招募译者”阶段。在任务进入招募译者与翻译进行中阶段时，发布者不可操作。

**结束任务** 在任务处于“任务待结束”状态时，发布者可以双击该任务行，查看译文并结束任务。

**1.2.4 负责人流程**

**报名成为负责人** 在点击主界面“任务类型”板块中的负责人按钮后即弹出窗口并显示出当前未过期的招募负责人状态的任务ID、酬金与积分要求，若已报名或积分不足则双击后会有对应提示，否则进入任务详情界面，在该界面中点击报名即视为报名。

**查看并招募译者** 在任务进入“招募译者”阶段后，负责人可在“我参与的”板块中找到自己作为负责人的该任务，双击该行即可查看已经报名的候选译者的用户名与积分，双击该行进入译者任务分配界面，需完整填写内容后确认，此时对应译者已被招募并收到信息。完成所有译者招募后需在查看候选译者界面中点击确认按钮确认整个译者招募过程的完成。完成后任务立即进入“翻译进行中”状态。

**查看并审核译者译文** 在任务进入“翻译进行中”状态后，负责人可在“我参与的”板块中找到自己作为负责人的该任务，双击该行即可查看译者名字以及其译文是否待审核，若为待审核状态，双击后即可查看其译文，并可在审核意见中加入意见，点击“不通过”按钮则将退回译文与审核意见，使得该译者重新进入翻译进行中状态。点击通过按钮则使得该译者进入待结束任务状态。

**提交最终译文** 在任务进入“翻译进行中”状态后，负责人可在“我参与的”板块中找到自己作为负责人的该任务，双击该行即可查看译者名字以及其译文是否待审核，若所有译者审核均通过后，负责人可以在该页面下方文本框内输入最终译文并提交，提交后该任务进入“待结束任务”状态。

**1.2.5 翻译者流程**

**报名成为译者** 在点击主界面“任务类型”板块中的翻译者按钮后即弹出窗口并显示出当前未过期的招募译者状态的任务ID、原文语言与目标语言，若已报名则双击后会有对应提示，否则进入任务详情界面，在该界面中点击报名即视为报名。

**进行翻译** 在被选为译者后，用户收到信息并且可在“我参与的”板块中找到自己作为译者的该任务，双击该行即可进入翻译界面，可看到需翻译文本与来自负责人的要求，点击暂存按钮即可保存已翻译内容，点击提交按钮即可将译文提交给负责人审核，同时对应任务进入待审核状态。若未通过审核则重新返回“翻译进行中”状态并收到更新后的反馈消息。若通过审核则进入“待结束任务”状态。

# 2平台结构设计

平台主要包括了数据模块和界面模块。其中界面模块提供了视觉上的信息效果与操作方式，而数据模块主要负责数据的存储与读入，为界面模块提供支持。

## 2.1数据模块

整个系统的数据主要分为两种：用户基本数据与任务基本数据。用户基本数据包含了用户类、积分类、账户类与消息盒子类，任务类下派生出发布者任务类、负责人任务类与译者任务类。通过SQLite数据库实现对数据的存储与读入。

**2.1.1 用户类**

用户类UserClass为注册时存入的基本信息类，包含了用户的唯一用户名、密码、姓名、国籍、手机号、身份证号、个人兴趣点，均为QString类型的变量。

用户类的设计目的是对用户注册、用户登陆以及用户信息维护提供便利。

**2.1.2 积分类**

积分类UserPoints为单独存储用户积分的类，包含了用户的用户名以及积分。

主要运用于负责人报名时积分判定，选择负责人或译者时的积分查看，以及任务结束后积分的增加。

**2.1.3 账户类**

账户类Account为单独存储用户账户的类，包含了用户名与账户金额。主要用于账户的充值，发起者发布任务的支付、负责人分配任务的支付、任务过期的各项退款以及任务结束后负责人、译者的账户增加。

**2.1.4 消息盒子类**

消息盒子类MyMessageBox为存储消息的类，包含了用户名、任务ID与消息内容。主要用于存储每一个需要提醒位置处的消息。

**2.1.5 任务类**

任务类Mission一共派生出三种类，包括了发布者任务类RMission、负责人任务类CMission与译者任务类TMission。任务类统一保存了任务的基本信息、各个角色的用户名、任务的DDL，而其派生出来的各个类中拥有独自的数据成员。

每一个派生类负责存储该角色视角下的任务内容。

**2.1.6 数据库类**

数据库类MyDatabase包含了与数据库对接需要的所有操作，使得主程序中对数据库的使用简化。

## 2.2界面模块

本项目的界面主要由11个界面类构成，包括了开始界面、注册与修改信息界面、主界面、发布者报名界面、负责人报名界面、翻译者报名界面、报名详细信息界面、选择候选者界面、翻译进行中的信息界面、结束任务界面、充值界面、消息盒子与总信息界面。

# 3平台详细设计

## 3.1 类结构设计

**3.1.1 用户类**

UserClass包含有用户名、密码、手机号、姓名、身份证、国籍、附加说明七个私有成员，均为QString类型，以及对应的构造函数与访问函数（即get开头的函数）。

**3.1.2 用户积分类**

UserPoints包含了私有成员用户名与积分，用户名为QString类型，积分为double类型，初始积分的给定根据注册时填入的六级考试分数确定。

公有成员函数包括：

·QString getUserName()和double getUserPoints()：输出用户名与积分

·void finishMission()：完成任务，使得积分+1

·void showUserPts(int x,QTableWidget \*j)：在表中输出单条积分信息

**3.1.3 账户类**

Account类包含了私有成员用户名、余额，用户名为QString类型，余额为double类型，注册后初始余额为0。

公有成员函数包括：

·QString getUserName()和double getMoney()：输出用户名与余额

·void addMoney(double add)：充值、任务完成支付酬金或是任务过期退款

·void payMoney(double pay)：发起任务时的支付与分配译者任务时的支付

**3.1.4 消息盒子类**

MyMessageBox类包含了私有成员用户名、任务ID、消息，均为QString类型。除了构造函数外包括公有函数：

·QString getId()和QString getMessage()：输出消息对应任务ID与内容

·void showMessageTable(int x,QTableWidget \*j)：在表中输出单条消息

**3.1.5 任务类**

**Mission类**包含了任务编号、任务状态、原始语言、目标语言、待翻译内容、任务说明、总酬金、发布者名字、负责人名字、译者名字列表、最终过期时间这些保护成员，作为其余任务类型的基类。拥有两种构造函数，分别是为其派生类的构造服务的。除了拥有各个信息读取函数以外，还拥有公有成员函数：

·void overDDL()：任务过期（进度赋值为-1）

·void setmissionstate()：任务状态推进

·void addTranslater(QString name)：增加译者

·bool isTranslater(QString name)：判断是否为译者

·bool translaterNum():判断是否选择了译者

·void show(int x,QTableWidget \*j)：将任务在主窗口中显示（显示id与状态）

**RMission类**为发布者任务类，继承于Mission类，有私有成员候选负责人列表、需要的积分、候选人招募截止日期与译者招募截止日期。拥有两种构造函数，分别用于新发布任务与数据库导入任务。除了拥有信息读取函数以外，还拥有公有成员函数：

·void addcan\_cname(QString canname)：增加负责人候选者

·void show\_choosecname(int x,QTableWidget \*j)：在表中打印出负责人候选人名单

·void show\_allmission(int x,QTableWidget \*j)：在表中打出全部任务清单

·void choosecname(QString i)：选择负责人

·void confirmfinish()：确认任务完成

·bool iscan\_name()；判断用户是否报名为负责人

**CMission类**为负责人任务类，继承于Mission类，有私有成员候选译者列表与译者招募截止日期。拥有两种构造函数，分别用于新创建任务与数据库导入任务。除了拥有信息读取函数以外，还拥有公有成员函数：

·void addcan\_tname(QString canname)：增加译者候选者

·void show\_choosetname(int x,QTableWidget \*j)：在表中打印出译者候选人名单

·bool iscan\_tname()；判断用户是否报名为译者

**TMission类**为译者任务类，继承于Mission类，有私有成员当前译者姓名、分配给该译者的酬金、译者需要翻译的文本、是否通过负责人审核(bool)，公有成员变量负责人建议、译者暂存文本与译者提交文本。拥有两种构造函数，分别用于新创建任务与数据库导入任务。除了拥有信息读取函数以外，还拥有公有成员函数：

·void accept()：审核通过

·void disPass()：审核不通过

·void show\_realTname(int x,QTableWidget \*j)：在表中打印出译者名单

**3.1.6 数据库类**

**MyDataBase类**是程序与数据库对接的关键接口，包含了一系列数据存储与读入的操作。私有成员中包含了私有成员数据库变量以及私有函数：

Account \*searchAccount(QSting username,QVector<Account> \*t)：与主窗口中账户Vector对接函数，主要是为了在判断任务过期时提供退款服务的搜索函数，

int query(QString username,QString password)：对数据库中的用户名与密码进行匹配的函数,匹配成功则返回2，无用户返回1，失败返回0

void clear(QString tbname,QString parameter)：清空名为tbname的表格

QSqlDatabase openDatabase()：检验、打开数据库，返回数据库变量

包含的公有函数：

公有函数主要包括了初始化信息函数、保存信息函数、注册登录函数与消息盒子函数

初始化信息函数：包括UserClass createUserClass(QString username)以及返回值为QVector<T>(T为 MyMessageBox, Account, UserPoints, RMission, CMission, TMission)的函数，用于在主界面中构建出用户类对象以及各种类型的Vector，便于数据的读取。其中所有任务均在读取时判定是否过期，若过期则标定为过期任务并调用searchAccount函数进行退款。

保存信息函数：包括所有void save开头的函数，作用是将主界面中的QVector与当前用户对象信息保存至数据库中。

注册登陆函数：包括了：

·Bool CreateTable()：登陆时判定用户相关表格是否创建

·Int addUser():尝试增加用户信息，1为重名，2为成功，-1为失败

·Int loginCheck()：登陆检验

消息盒子函数：包括了：

·Void createMessageTable(QString username,QTableWidget \*j)：在表格中打印出所有发送给username的消息

·Void addNewMessage(QString username,QString id,QString message)：新增新消息

·void deleteMessage(QString username)：删除与username有关的消息

**3.1.7 主界面类**

**MainWindow类**提供了用户登陆后的主窗口，是存放数据库导入信息、提供各种按钮与操作的基础平台。

除了主窗口ui外，包含有私有成员变量：充值界面类对象指针、发布者/负责人/翻译者报名界面对象指针、数据库对象指针与消息盒子界面对象指针。除此之外，还有私有成员函数：

·void setUserInfo()与void setPtsInfo()与void setAccountInfo()：分别为在“我的信息”板块中配置用户信息、积分信息、账户信息的函数

拥有私有信号函数：

void showAllMission()：弹出展示所有任务的窗口

void showReleaserWindow()与void showChargerSignupWindow()与void showTranslaterSignupWindow()：分别为弹出发布者、负责人、翻译者报名的窗口

void showMessageBox()：弹出消息盒子窗口

void showChooseCnameWindow(int row,int col)：弹出选择负责人窗口

void showChooseTnameWindow(int row, int col)：弹出选择译者窗口

void showTranslaterWindow(int,int)：弹出翻译者工作界面

void showChangeInfoWindow()：弹出修改信息窗口

void refreshMyInfo()：点击刷新按钮后的刷新操作

void showDepositWindow()：弹出充值窗口

公有成员中有公有变量：包括当前用户类对象u、当前账户对象指针thisaccount、当前用户积分类对象指针p，以及消息盒子类、账户类、用户积分类、三个任务类对象的Vector

公有函数分为四类，主要包括了在vector中的搜索函数、增加新对象函数、信息更新函数与消息盒子操作函数

在vector中的搜索函数：返回值为各类型对象指针的函数，提供在Vector中根据关键词查找对象指针的功能，方便于一切的查找

增加新对象函数：包括了void save开头的所有函数，功能是在相应对象的Vector中加入新成员

信息更新函数：包括了void saveAllInfo()与void autoRefresh()函数，分别是将所有信息存入数据库与自动刷新主界面内容

消息盒子操作函数：包括了3个与数据库类中的消息盒子函数对接的函数

**3.1.8 登陆界面类**

**MyApp类**中包括了登陆所需要的操作，除了自身ui外包括私有成员：数据库类对象的指针与主界面对象指针，私有信号成员有

·void login()实现了登陆检验与主界面的跳转

·void fregister()实现了显示注册界面

**3.1.9 注册界面/信息修改界面类**

**registerwindow类**中包括了注册需要进行的操作与修改个人信息需要的操作。由于两者在流程上几乎一致，因而选择共用一个类。除了自身ui外包括了私有成员：数据库类对象的指针、用户类对象指针与各个QString成员。私有成员函数有：

·bool checkphonenum()：检验手机号码格式

·bool checkall()：检验信息总格式

·bool IsEnglishOrNumber(QString &qstrSrc)：判定是否为数字或字母

公有成员函数void setChangeWindow(UserClass \*u)用于在注册界面的基础上构造出信息修改界面。

信号函数：

·void confirm\_slots()：检验注册格式并进行注册流程

·void changeUserInfo()：检验格式并进行信息修改流程并替换Mainwindow中的当前用户信息

·void back()：返回

**3.1.10 发布者报名界面类**

**ReleaserWindow类**提供给用户发布任务的界面，也是MainWindow的一个成员变量子窗口。核心信号函数为void addnewRMission()，即将表中所填写的数据通过构建新的RMission类对象并添加到父窗口的相应Vector中来达到效果。添加完成后自动保存并在父窗口中存入数据库。

**3.1.11 负责人报名界面类**

**ChargerWindow类**提供给用户报名负责人的查看界面，也是MainWindow的一个成员变量子窗口。由于其下还有子窗口，因而将主窗口的存储与增加信息函数利用指针写入此类中成为void saveAll()与void addMessage(QString, QString,QString)函数，功能与主窗口中对应函数完全一致。此外还有：

·void showRMinfo()在表中展示出当前可供报名的信息

·int cmsignup(RMission \*temp)用于检验并返回报名信息，通过在temp的候选人列表中查询来保证报名成功

·void showdetail(int row,int column)用于打开对应该行的报名详细信息窗口

**3.1.12 翻译者报名界面类**

**TranslaterWindow类**提供给用户报名译者的查看界面，也是MainWindow的一个成员变量子窗口。有功能函数：

·int tmsignup(CMission \*temp)用于检验并返回报名信息，通过在temp的候选人列表中查询来保证报名成功

·void showdetail(int row,int column)用于打开对应该行的报名详细信息窗口

**3.1.13 报名详细信息界面类**

**CSignuup类**给用户展现出了查看任务详细信息的界面。由于翻译者与负责人的报名所需看的任务信息类似，因而两者共用一个窗口界面，只是构造函数需要传输的父对象不同。两个核心信号函数void signup()与void tsignup()分别是从其父窗口的对应报名函数中获得检验值并确认完成报名过程。

**3.1.14 选择候选者界面类**

**ChooseCname类**共有3个构造函数，分别对应的是提供选择负责人界面、选择译者界面与管理译者工作界面（三者界面相似且功能相关）。由于其下还有子窗口，因而将主窗口的存储与增加信息函数利用指针写入此类中成为void saveAll()与void addMessage(QString, QString,QString)函数，功能与主窗口中对应函数完全一致。

在选择负责人界面中，对应有函数：

·void showCname()：在表中展示出负责人候选者名单

·void chooseCharger(int,int)：双击选择该行展示的用户为负责人

在选择译者界面中，对应有函数：

·void showTname()：在表中展示出译者候选者名单

·void chooseTrans(int,int)：双击打开为该译者分配任务的界面

·void finishChooseTrans()：结束译者的选择工作

·void addNewTranslater(QString , QString ,double ,QString ,QString)：新建出译者任务并放入主界面对应Vector中

在管理译者的工作界面中，对应有函数：

·void showRealTname()：在表中展示出译者名单及对应任务状态

·void showTransWork(int,int)：双击打开译者工作详情的界面

·void finalConfirm():负责人最终将输入在最下方的译文提交

**3.1.15 翻译进行中的信息界面类**

**RmissionInfo类**共有3个构造函数，分别对应的是提供分配译者任务界面、译者工作界面与审核译者内容界面（三者界面相似且功能相关）。

分配译者任务界面是给负责人具体分配译者任务设置的，核心函数为void chooseTranslater()，即完成当前任务的布置，创建出新的TMission对象并存储，而后发送相关信息。

译者工作界面是给译者翻译过程中使用的，会提供需要翻译的文本内容和要求，主要函数有：

·void tempSave()：将翻译的内容暂存，即任务进度不变

·void tConfirm()：将翻译的内容提交，即使得任务进入待审核状态

审核译者内容界面是给负责人审核并给出修改意见使用的，会保留上次的意见，主要函数有：

·void tmisssionPass()：任务通过，使得译者任务进入待完成状态

·void tmissionDisPass(): 任务不通过，退回译文与审核意见，使得译者继续进入翻译进行中状态

**3.1.16 结束窗口类**

**FinishWindow类**提供给发起者最后的结束任务操作，即可以在此查看最终译文并确认结束任务。其核心函数为void finish()，即将所有相关任务设置为已结束状态并结算酬金。

**3.1.17 充值界面类**

**Deposit类**提供给用户充值界面，核心函数为void depositMoney()，即将用户在框内输入的数值加入自己账户余额中，实现充值功能。

**3.1.18 消息盒子与总信息类**

**MessageInfo类**共有两个构造函数，分别对应于消息盒子界面和全部任务信息界面。在消息盒子界面中，核心函数为void readAllMessage()，即在按下全部标为已读时清空消息盒子，而在全部任务信息界面中，通过私有函数void showTable()来在表中打出所有的任务信息。

## 3.2 数据库/文件结构设计

本项目使用了SQLite数据库，在数据库中存在7张表。

USER表中存入的是每个用户的基本信息，共有7列，分别为用户名、密码、姓名、手机号、国籍、身份证号、附加信息。

Userpoints表中存入的是每个用户的积分信息，共有2列，分别为用户名、积分。

Account表中存入的是每个用户的余额信息，共有2列，分别为用户名、余额。

MyMessageBox表中存入的是消息盒子新信息，共有3列，分别为用户名、任务ID、消息具体内容。

RMission表中存入的是发起者任务的全部信息，共有16列，分别为任务ID、任务状态、原始语言、目标语言、翻译文本、任务描述、总酬金、发起者用户名、负责人用户名、翻译者用户名序列、候选负责人用户名序列、积分需求、最终译文、最终DDL、负责人招募DDL、译者招募DDL。

CMission表中存入的是发起者任务的全部信息，共有13列，分别为任务ID、任务状态、原始语言、目标语言、翻译文本、任务描述、总酬金、发起者用户名、负责人用户名、翻译者用户名序列、候选译者用户名序列、最终DDL、译者招募DDL。

TMission表中存入的是发起者任务的全部信息，共有13列，分别为任务ID、任务状态、原始语言、目标语言、负责人用户名、该译者用户名、该译者可获得的酬金、该译者需要翻译的文本、负责人反馈意见、暂存文本、提交文本、是否被接受、最终DDL。

其中大部分信息是以字符串存入，对于原先为QStringList的变量，先在每个字符之间加入“-”后存为字符串，读取时取消“-”生成QStringList。

## 3.3 界面结构设计

**3.3.1注册登陆界面设计**

考虑到整个项目是需要提供给用户的，因而登陆界面尽可能设计得简洁美观，仅有用户名密码的输入框以及登陆、注册按钮，点击注册按钮后弹出注册界面，输入正确的用户名和密码后进入主界面。

在注册界面中，用户需要完整合法地输入信息后点击注册按钮才能完成注册，除了注册与返回按钮外，还设置了重置按钮，即可以重置输入内容。

**3.3.2主界面设计**

用户登陆后即进入主界面。主界面提供了四个板块窗口。

在“任务类型”板块中可以选择以不同的角色身份报名：点击发起者按钮进入发布者报名界面，点击负责人按钮查看可供报名的正在招募负责人的界面，点击翻译者按钮查看可供报名的正在招募负责人的界面，点击查看全部任务可以查看到所有的任务。

在“我参与的”板块中可以查看用户分别作为发布者、负责人、翻译者的参与任务，双击该行则可以进入具体的操作界面。在左下方则有两个按钮，点击刷新按钮可以刷新该页面的信息，点击消息盒子按钮可以进入消息盒子界面。

在“用户信息”板块中可以查看用户基本信息、积分与账户信息，点击修改信息按钮可以进入修改信息界面，点击“充值”按钮可以进入充值界面。

在“关于我们”界面中可看到开发者信息。

**3.3.3 发布者报名界面设计**

从主界面进入后需要完成所有信息后点击发布按钮才能发布，发布成功弹出消息窗口点确认或者点击取消按钮后自动关闭界面。

**3.3.4 负责人/翻译者报名界面设计**

从主界面进入后首先显示出目前可报名任务的表格，在表格显示后双击某一行即可进入任务详情介绍界面，若积分不足或者已经报名则在双击时弹出提示框并返回，在任务详情介绍界面中所有任务信息为只读信息，点击报名按钮即可报名，点击取消按钮后返回，在报名完成弹出提示窗口确认后或取消按钮后自动关闭界面。

**3.3.5 负责人/翻译者选择界面设计**

在主界面中“我参与的”中分别双击对应状态的任务即可进入负责人/翻译者选择界面。

在负责人选择界面中，表中打印出报名者的信息，双击某行即可选择其成为负责人，选择完成后自动关闭页面。

在翻译者选择界面中，表中打印出报名者的信息，双击某行即可进入对该报名者的任务分配界面，在任务分配界面中可输入分配任务与任务要求，点击确认即分配任务并自动关闭分配界面，在所有译者确定后点击确认即可完成译者招募并自动关闭界面。

**3.3.6 翻译者工作界面设计**

在主界面中“我参与的”中双击翻译者的正处于翻译进行中状态的任务即可进入翻译者工作界面，译者可以查看任务内容、审核意见并进行翻译，点击暂存按钮可以暂存，点击提交按钮可以提交给负责人并自动关闭界面。

**3.3.7 负责人审核界面设计**

在主界面中“我参与的”中双击负责人的正处于翻译进行中状态的任务即可进入审核界面。在表中显示出对应译者姓名与状态，若为待审核则可以双击进入，提供审核意见并决定通过与否。在所有人都通过后，在审核界面底部可以输入最终译文并点击按钮提交，提交后自动关闭界面。

**3.3.8 结束任务界面设计**

在主界面中“我参与的”中双击发起者处于待完成任务状态的任务即可进入结束任务界面。在文本框中显示出最终译文，点击确认即可结束任务并自动关闭界面。

**3.3.9 消息盒子界面设计**

在主界面“我参与的”中点击消息盒子即可进入消息盒子界面，点击全部标为已读可以清空消息盒子并自动关闭。

**3.3.10 充值界面设计**

在主界面“用户信息”中点击充值按钮即可进入充值界面，输入金额并点击充值后即可充值并自动关闭界面。

## 3.4 关键设计思路

**3.4.1 容错机制设计**

由于本项目涉及到用户与用户的信息交流，因而良好的容错机制显得尤为必要。本项目主要依靠如下手段实现容错：

函数检验：包括在注册与修改信息时对于手机号、密码相等、用户名重复以及信息是否完整的检验函数，在发布任务时对于任务内容是否完整、原文语言和目标语言是否重复、DDL设置是否正确的检验函数，在提交任务时对于提交内容是否为空的检验函数。

文本只读：包括在负责人、翻译者的报名界面中所有任务信息只读、所有表中显示数据只读、“用户信息”中数据只读、翻译者工作界面中对于任务内容与审核意见的只读等等。

大小限定：包括对于所有窗口的大小限定，使得不会因为最大化而使得结构变形。

**3.4.2 数据隔离机制设置**

由于任务类的设定，数据的隔离得以进行。包括了发布者与翻译者之间几乎截断了一切联系，各个任务彼此关联而数据能够很好的隔离开来。

同时，由于用户类、积分类与账户类的分离，用户个人信息的安全性得到了更好的保证，不会出现过多信息的泄露。

**3.4.3 任务编号机制**

鉴于三类任务的联系与任务编号的唯一性，我选择在任务发布时选取当时时间序列作为任务的编号，且同一系列的任务用同一编号，既保证了任务编号的唯一性，又使得编号有了一定的意义。

**3.4.4 数据传输与保存机制**

在整个项目中，几乎所有的数据均会先保存到对象中，再将对象相同的保存至主界面的Vector中，最后由主界面调用保存函数存入数据库中，读入数据的过程即为上述过程的反转，这样使得数据传输与保存简洁有效。

同时，为了便于数据的传输，每个新窗口均可以通过构造函数的指针变量联系到上一级窗口，实现传输功能。

**3.4.5 窗口复用设计**

由于不少窗口界面外观类似、功能有一定联系，因而很多界面共用一个ui，这就造成了很多类会有不同的构造函数，这大大节约了开发时间。

## 3.5 附加功能实现

**3.5.1 多语言互译**

在发布者发布任务时，发布者可以选择不同语种之间的转换。

**3.5.2 消息盒子新消息提醒**

一旦该用户的消息盒子中出现了新消息，在消息盒子按钮的右上角即可以出现new的标志，做到真正的消息盒子提醒效果。而当用户读完全部消息后，标志自动消失。

# 4项目总结

**4.1 项目亮点**

**4.1.1 界面美观，操作舒适**

无论是登陆界面还是主界面中，许多文本框与按钮均通过stylesheet进行了对参数的调整，登陆界面中的文本框与按钮均有一定的透明度，而在主界面中的按钮在鼠标移动上去时会有立体凹凸效果，刷新按钮与消息盒子按钮均使用了图标，使得整体很美观而简洁。

同时，几乎每一个界面中都有文字的操作提示，这也使得操作上手十分容易。以及登陆界面输入完密码后按下回车即可确认，让用户操作更为便捷。

**4.1.2 功能完整，考虑周全**

除了正常要求以外，对于数据的更新与保存非常及时性，保证了操作完成即使是中途关闭也能够保存数据。同时，对于过期任务的退款，也在每次导入数据时进行检验，保证款项正常。

**4.2 开发过程遇到的主要问题与解决办法**

**4.2.1 任务类的逻辑设定**

在最开始的时候，我的开发效率并不高，主要就是因为类的逻辑关系没有完全想通，初期设置很混乱。不过，当我停下来仔细思考一两天并绘制出关系图后，很多东西逐渐清晰起来，并且对于原来思维上的阻隔也有了新的想法。

**4.2.2 数据库的连接**

数据库的连接是我在整个开发与调试过程中遇到的较大问题之一。一开始几乎次次报错，咨询了很多网上的资料但结果仍然不尽人意。后来我改变了思路，决定好好理解清楚每一行代码的作用，并且开出小项目进行调试，这才使得问题逐渐得到解决。

**4.3 心得体会**

在整个项目开发的过程中，我主要有以下方面的体会。

**4.3.1 自学能力**

很多时候，项目开发所需的知识是超出自身知识范畴的，包括此次Qt、SQLite的使用，但这并不意味着问题无法解决。在网络发达、信息交流便捷的今天，无论是从例如CSDN等的网站上得到知识，还是通过询问老师、助教，都能够使得问题更好地解决。

**4.3.2 DEBUG的耐心**

由于时常会出现程序异常结束的报错却没有具体错误信息，在此次过程中我更加有了DEBUG的耐心，包括一步步设置断点、寻找突破契机等等，因为这无非是一个找错的过程，错误既然出现，那就必须要寻找到底。

**4.3.3 交流中碰撞出火花**

在进度突然受阻的时候，与遇到过类似问题的同学们的交流能够打开自己的思路，进而有可能解决问题。