**Министерство образование и науки Украины**

**Харьковский национальный университет радиоэлектроники**

Кафедра ЭВМ

**Контрольная работа**

**по дисциплине**

**«Параллельные и распределенные вычисления»**

Выполнил:

Студент группы КИз-12-3

Лахно О.Г.

Проверил:

Аксак Н.Г

Харьков 2017 г.

**Цель работы.**

Практическое освоение архитектурных принципов реализации параллельной обработки в вычислительных машинах и методов конструирования параллельных программ.

**Задача.**

Постройте автоматизированную банковскую систему с несколькими пользовательскими "терминалами" и несколькими банками. Каждый терминал связан с одним банком, банки связаны между собой. Каждый счет хранится в банке, но пользователь должен иметь возможность получить доступ к своему счету с любого терминала. В программе следует реализовать несколько типов пользовательских транзакций, таких как помещение денег на счет, снятие денег, запрос баланса.

**Выполнение.**

Программа писалась в QT Creator 4.1.0 с использованием sqlite3.

**Листинг:**

**bank.pro**

#-------------------------------------------------

#

# Project created by QtCreator 2016-08-13T09:26:05

#

#-------------------------------------------------

QT += core gui sql

greaterThan(QT\_MAJOR\_VERSION, 4): QT += widgets

TARGET = bank

TEMPLATE = app

SOURCES += main.cpp\

mainwindow.cpp \

terminal1.cpp \

globals.cpp \

terminal2.cpp

HEADERS += mainwindow.h \

terminal1.h \

globals.h \

terminal2.h

FORMS += mainwindow.ui \

terminal1.ui \

terminal2.ui

QMAKE\_LIBS+=-lgomp

QMAKE\_CXXFLAGS+=-fopenmp

QMAKE\_LFLAGS += -fopenmp

**globals.h**

#ifndef GLOBALS\_H

#define GLOBALS\_H

extern int userid;

#endif // GLOBALS\_H

**mainwindow.h**

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <omp.h>

namespace Ui {

class MainWindow;

}

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

explicit MainWindow(QWidget \*parent = 0);

~*MainWindow*();

private slots:

void on\_btnterminal1\_clicked();

void on\_btnexit\_clicked();

void on\_btnterminal2\_clicked();

private:

Ui::MainWindow \*ui;

};

#endif // MAINWINDOW\_H

**Terminal1.h**

#ifndef TERMINAL1\_H

#define TERMINAL1\_H

#include <QDialog>

#include <QDebug>

#include <QtSql>

#include <QFileInfo>

#include <omp.h>

namespace Ui {

class Terminal1;

}

class Terminal1 : public QDialog

{

Q\_OBJECT

public:

explicit Terminal1(QWidget \*parent = 0);

~*Terminal1*();

private slots:

void on\_btnback\_clicked();

void on\_btnlogin\_clicked();

void on\_brnLogout\_clicked();

void on\_btnadd\_clicked();

void on\_bankBox\_currentIndexChanged(int index);

void on\_btnwithdraw\_clicked();

private:

Ui::Terminal1 \*ui;

QSqlDatabase MyDB;

};

#endif // TERMINAL1\_H

**terminal2.h**

#ifndef TERMINAL2\_H

#define TERMINAL2\_H

#include <QDialog>

#include <QDebug>

#include <QtSql>

#include <QFileInfo>

#include <omp.h>

namespace Ui {

class Terminal2;

}

class Terminal2 : public QDialog

{

Q\_OBJECT

public:

explicit Terminal2(QWidget \*parent = 0);

~*Terminal2*();

private slots:

void on\_brnLogout\_clicked();

void on\_btnlogin\_clicked();

void on\_btnback\_clicked();

void on\_btnadd\_clicked();

void on\_btnwithdraw\_clicked();

void on\_bankBox\_currentIndexChanged(int index);

private:

Ui::Terminal2 \*ui;

QSqlDatabase MyDB;

};

#endif // TERMINAL2\_H

**globals.cpp**

#include "globals.h"

int userid = 0;

**main.cpp**

#include "mainwindow.h"

#include <QApplication>

int main(int argc, char \*argv[])

{

QApplication a(argc, argv);

MainWindow w;

w.show();

return a.exec();

}

**mainwindow.cpp**

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include "terminal1.h"

#include "terminal2.h"

MainWindow::MainWindow(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~*MainWindow*()

{

delete ui;

}

void MainWindow::on\_btnterminal1\_clicked()

{

Terminal1 \*term1 = new Terminal1(this);

term1->show();

}

void MainWindow::on\_btnexit\_clicked()

{

close();

}

void MainWindow::on\_btnterminal2\_clicked()

{

Terminal2 \* term2 = new Terminal2(this);

term2->show();

}

**terminal1.cpp**

#include "terminal1.h"

#include "ui\_terminal1.h"

#include "mainwindow.h"

#include "QMessageBox"

#include "globals.h"

#include "QDir"

Terminal1::Terminal1(QWidget \*parent) :

QDialog(parent),

ui(new Ui::Terminal1)

{

ui->setupUi(this);

ui->lblwelcome->hide();

ui->brnLogout->hide();

MyDB = QSqlDatabase::addDatabase("QSQLITE");

MyDB.setDatabaseName("D:/pro/bank/db1.sqlite");

QFileInfo CheckFile("D:/pro/bank/db1.sqlite");

if(CheckFile.isFile())

{

if (MyDB.open()){

ui->lblconnect->setText("Connected");

}

}else{

ui->lblconnect->setText("Not Connected");

}

}

Terminal1::~*Terminal1*()

{

delete ui;

qDebug() << "Closing connection";

MyDB.close();

}

void Terminal1::on\_btnback\_clicked()

{

hide();

}

void Terminal1::on\_btnlogin\_clicked()

{

QString Login, Pass;

Login = ui->txtLogin->text();

Pass = ui->txtPass->text();

if (!MyDB.isOpen()){

qDebug() << "no connection";

return;

}

QSqlQuery qry;

QSqlDatabase MyDB = QSqlDatabase::database();

if(qry.exec("SELECT \* FROM USER\_AUTH WHERE LOGIN =\'" + Login + "\' AND PASSWORD =\'" + Pass + "\'"))

{if(qry.next())

{

userid = qry.value(0).toInt();

ui->btnlogin->hide();

ui->txtLogin->hide();

ui->txtPass->hide();

ui->lbllogin->hide();

ui->lblpassword->hide();

ui->lblwelcome->setText("Welcome, " + Login + "!");

ui->lblwelcome->show();

ui->brnLogout->show();

QMessageBox msga;

msga.setText("Login successful");

msga.*exec*();

}else{

QMessageBox msgw;

msgw.setText("Wrong Login information");

msgw.*exec*();

}

}

}

void Terminal1::on\_brnLogout\_clicked()

{

ui->btnlogin->show();

ui->txtLogin->show();

ui->txtPass->show();

ui->lbllogin->show();

ui->lblpassword->show();

ui->lblwelcome->hide();

ui->brnLogout->hide();

ui->bankBox->setCurrentIndex(0);

}

void Terminal1::on\_btnadd\_clicked()

{ int balance;

if(ui->txtAct->text().toInt()>0)

{

QString ID = QString::number(userid);

QSqlQuery qryaddmoney;

QSqlQuery qrycheckmoney;

if(ui->bankBox->currentIndex()==1)

{

if(qryaddmoney.exec("SELECT \* FROM Bank1 WHERE ID =\'" + ID + "\'"))

if(qryaddmoney.next())

#pragma omp parallel for

{

balance = qryaddmoney.value(1).toInt();

balance = balance + ui->txtAct->text().toInt();

QString sbalance = QString::number(balance);

qryaddmoney.exec("UPDATE Bank1 SET CURRENT\_BALANCE =\'" + sbalance +"\'WHERE ID =\'" + ID + "\'");

qrycheckmoney.exec("SELECT \* FROM Bank1 WHERE ID =\'" + ID + "\'");

if(qrycheckmoney.next())

ui->txtBalance->setText(QString::number(qrycheckmoney.value(1).toInt()));

}

}else if(ui->bankBox->currentIndex()==2)

{ if(qryaddmoney.exec("SELECT \* FROM Bank2 WHERE ID =\'" + ID + "\'"))

if(qryaddmoney.next())

#pragma omp parallel for

{

balance = qryaddmoney.value(1).toInt();

balance = balance + ui->txtAct->text().toInt();

QString sbalance = QString::number(balance);

qryaddmoney.exec("UPDATE Bank2 SET CURRENT\_BALANCE =\'" + sbalance +"\'WHERE ID =\'" + ID + "\'");

qrycheckmoney.exec("SELECT \* FROM Bank2 WHERE ID =\'" + ID + "\'");

if(qrycheckmoney.next())

ui->txtBalance->setText(QString::number(qrycheckmoney.value(1).toInt()));

}

}else if(ui->bankBox->currentIndex()==0)

ui->txtBalance->clear();

}

}

void Terminal1::on\_bankBox\_currentIndexChanged(int index)

{

QString ID = QString::number(userid);

QSqlQuery qrybankinfo;

if(ui->bankBox->currentIndex()==1)

{

if(qrybankinfo.exec("SELECT \* FROM Bank1 WHERE ID =\'" + ID + "\'"))

if(qrybankinfo.next())

{

ui->txtBalance->setText(QString::number(qrybankinfo.value(1).toInt()));

}

}else if(ui->bankBox->currentIndex()==2)

{ if(qrybankinfo.exec("SELECT \* FROM Bank2 WHERE ID =\'" + ID + "\'"))

if(qrybankinfo.next())

{

ui->txtBalance->setText(QString::number(qrybankinfo.value(1).toInt()));

}

}else if(ui->bankBox->currentIndex()==0)

ui->txtBalance->clear();

}

void Terminal1::on\_btnwithdraw\_clicked()

{

int balance;

if(ui->txtAct->text().toInt()>0)

{

QString ID = QString::number(userid);

QSqlQuery qryaddmoney;

QSqlQuery qrycheckmoney;

QMessageBox msgw;

if(ui->bankBox->currentIndex()==1)

{

if(qryaddmoney.exec("SELECT \* FROM Bank1 WHERE ID =\'" + ID + "\'"))

if(qryaddmoney.next())

{

balance = qryaddmoney.value(1).toInt();

if(balance>=0 && balance>ui->txtAct->text().toInt())

#pragma omp parallel for

{

balance = balance - ui->txtAct->text().toInt();

QString sbalance = QString::number(balance);

qryaddmoney.exec("UPDATE Bank1 SET CURRENT\_BALANCE =\'" + sbalance +"\'WHERE ID =\'" + ID + "\'");

qrycheckmoney.exec("SELECT \* FROM Bank1 WHERE ID =\'" + ID + "\'");

if(qrycheckmoney.next())

ui->txtBalance->setText(QString::number(qrycheckmoney.value(1).toInt()));

msgw.setText("Money withdrawn = " + ui->txtAct->text() + "");

msgw.*exec*();

}else

{

msgw.setText("Unable to perform action");

msgw.*exec*();

}

}

}else if(ui->bankBox->currentIndex()==2)

{ if(qryaddmoney.exec("SELECT \* FROM Bank2 WHERE ID =\'" + ID + "\'"))

if(qryaddmoney.next())

{

balance = qryaddmoney.value(1).toInt();

if(balance>=0 && balance>ui->txtAct->text().toInt())

#pragma omp parallel for

{

balance = balance - ui->txtAct->text().toInt();

QString sbalance = QString::number(balance);

qryaddmoney.exec("UPDATE Bank2 SET CURRENT\_BALANCE =\'" + sbalance +"\'WHERE ID =\'" + ID + "\'");

qrycheckmoney.exec("SELECT \* FROM Bank2 WHERE ID =\'" + ID + "\'");

if(qrycheckmoney.next())

ui->txtBalance->setText(QString::number(qrycheckmoney.value(1).toInt()));

msgw.setText("Money withdrawn = " + ui->txtAct->text() + "");

msgw.*exec*();

}else

{

msgw.setText("Unable to perform action");

msgw.*exec*();

}

}

}else if(ui->bankBox->currentIndex()==0)

ui->txtBalance->clear();

}

}

**terminal2.cpp**

#include "terminal2.h"

#include "ui\_terminal2.h"

#include "mainwindow.h"

#include "QMessageBox"

#include "globals.h"

Terminal2::Terminal2(QWidget \*parent) :

QDialog(parent),

ui(new Ui::Terminal2)

{

ui->setupUi(this);

ui->lblwelcome->hide();

ui->brnLogout->hide();

MyDB = QSqlDatabase::addDatabase("QSQLITE");

MyDB.setDatabaseName("D:/pro/bank/db1.sqlite");

QFileInfo CheckFile("D:/pro/bank/db1.sqlite");

if(CheckFile.isFile())

{

if (MyDB.open()){

ui->lblconnect->setText("Connected");

}

}else{

ui->lblconnect->setText("Not Connected");

}

}

Terminal2::~*Terminal2*()

{

delete ui;

}

void Terminal2::on\_brnLogout\_clicked()

{

ui->btnlogin->show();

ui->txtLogin->show();

ui->txtPass->show();

ui->lbllogin->show();

ui->lblpassword->show();

ui->lblwelcome->hide();

ui->brnLogout->hide();

ui->bankBox->setCurrentIndex(0);

}

void Terminal2::on\_btnlogin\_clicked()

{

QString Login, Pass;

Login = ui->txtLogin->text();

Pass = ui->txtPass->text();

if (!MyDB.isOpen()){

qDebug() << "no connection";

return;

}

QSqlQuery qry;

QSqlDatabase MyDB = QSqlDatabase::database();

if(qry.exec("SELECT \* FROM USER\_AUTH WHERE LOGIN =\'" + Login + "\' AND PASSWORD =\'" + Pass + "\'"))

{if(qry.next())

{

userid = qry.value(0).toInt();

ui->btnlogin->hide();

ui->txtLogin->hide();

ui->txtPass->hide();

ui->lbllogin->hide();

ui->lblpassword->hide();

ui->lblwelcome->setText("Welcome, " + Login + "!");

ui->lblwelcome->show();

ui->brnLogout->show();

QMessageBox msga;

msga.setText("Login successful");

msga.*exec*();

}else{

QMessageBox msgw;

msgw.setText("Wrong Login information");

msgw.*exec*();

}

}

}

void Terminal2::on\_btnback\_clicked()

{

hide();

}

void Terminal2::on\_btnadd\_clicked()

{

int balance;

if(ui->txtAct->text().toInt()>0)

{

QString ID = QString::number(userid);

QSqlQuery qryaddmoney;

QSqlQuery qrycheckmoney;

if(ui->bankBox->currentIndex()==1)

{

if(qryaddmoney.exec("SELECT \* FROM Bank1 WHERE ID =\'" + ID + "\'"))

if(qryaddmoney.next())

#pragma omp parallel for

{

balance = qryaddmoney.value(1).toInt();

balance = balance + ui->txtAct->text().toInt();

QString sbalance = QString::number(balance);

qryaddmoney.exec("UPDATE Bank1 SET CURRENT\_BALANCE =\'" + sbalance +"\'WHERE ID =\'" + ID + "\'");

qrycheckmoney.exec("SELECT \* FROM Bank1 WHERE ID =\'" + ID + "\'");

if(qrycheckmoney.next())

ui->txtBalance->setText(QString::number(qrycheckmoney.value(1).toInt()));

}

}else if(ui->bankBox->currentIndex()==2)

{ if(qryaddmoney.exec("SELECT \* FROM Bank2 WHERE ID =\'" + ID + "\'"))

if(qryaddmoney.next())

#pragma omp parallel for

{

balance = qryaddmoney.value(1).toInt();

balance = balance + ui->txtAct->text().toInt();

QString sbalance = QString::number(balance);

qryaddmoney.exec("UPDATE Bank2 SET CURRENT\_BALANCE =\'" + sbalance +"\'WHERE ID =\'" + ID + "\'");

qrycheckmoney.exec("SELECT \* FROM Bank2 WHERE ID =\'" + ID + "\'");

if(qrycheckmoney.next())

ui->txtBalance->setText(QString::number(qrycheckmoney.value(1).toInt()));

}

}else if(ui->bankBox->currentIndex()==0)

ui->txtBalance->clear();

}

}

void Terminal2::on\_btnwithdraw\_clicked()

{

int balance;

if(ui->txtAct->text().toInt()>0)

{

QString ID = QString::number(userid);

QSqlQuery qryaddmoney;

QSqlQuery qrycheckmoney;

QMessageBox msgw;

if(ui->bankBox->currentIndex()==1)

{

if(qryaddmoney.exec("SELECT \* FROM Bank1 WHERE ID =\'" + ID + "\'"))

if(qryaddmoney.next())

{

balance = qryaddmoney.value(1).toInt();

if(balance>=0 && balance>ui->txtAct->text().toInt())

#pragma omp parallel for

{

balance = balance - ui->txtAct->text().toInt();

QString sbalance = QString::number(balance);

qryaddmoney.exec("UPDATE Bank1 SET CURRENT\_BALANCE =\'" + sbalance +"\'WHERE ID =\'" + ID + "\'");

qrycheckmoney.exec("SELECT \* FROM Bank1 WHERE ID =\'" + ID + "\'");

if(qrycheckmoney.next())

ui->txtBalance->setText(QString::number(qrycheckmoney.value(1).toInt()));

msgw.setText("Money withdrawn = " + ui->txtAct->text() + "");

msgw.*exec*();

}else

{

msgw.setText("Unable to perform action");

msgw.*exec*();

}

}

}else if(ui->bankBox->currentIndex()==2)

{ if(qryaddmoney.exec("SELECT \* FROM Bank2 WHERE ID =\'" + ID + "\'"))

if(qryaddmoney.next())

#pragma omp parallel for

{

balance = qryaddmoney.value(1).toInt();

if(balance>=0 && balance>ui->txtAct->text().toInt())

{

balance = balance - ui->txtAct->text().toInt();

QString sbalance = QString::number(balance);

qryaddmoney.exec("UPDATE Bank2 SET CURRENT\_BALANCE =\'" + sbalance +"\'WHERE ID =\'" + ID + "\'");

qrycheckmoney.exec("SELECT \* FROM Bank2 WHERE ID =\'" + ID + "\'");

if(qrycheckmoney.next())

ui->txtBalance->setText(QString::number(qrycheckmoney.value(1).toInt()));

msgw.setText("Money withdrawn = " + ui->txtAct->text() + "");

msgw.*exec*();

}else

{

msgw.setText("Unable to perform action");

msgw.*exec*();

}

}

}else if(ui->bankBox->currentIndex()==0)

ui->txtBalance->clear();

}

}

void Terminal2::on\_bankBox\_currentIndexChanged(int index)

{

QString ID = QString::number(userid);

QSqlQuery qrybankinfo;

if(ui->bankBox->currentIndex()==1)

{

if(qrybankinfo.exec("SELECT \* FROM Bank1 WHERE ID =\'" + ID + "\'"))

if(qrybankinfo.next())

{

ui->txtBalance->setText(QString::number(qrybankinfo.value(1).toInt()));

}

}else if(ui->bankBox->currentIndex()==2)

{ if(qrybankinfo.exec("SELECT \* FROM Bank2 WHERE ID =\'" + ID + "\'"))

if(qrybankinfo.next())

{

ui->txtBalance->setText(QString::number(qrybankinfo.value(1).toInt()));

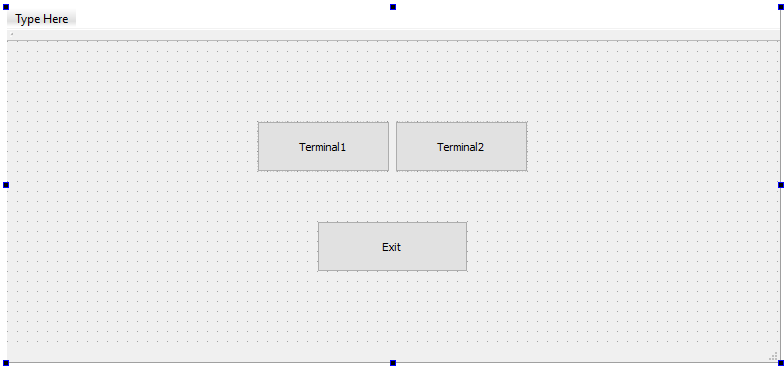
}

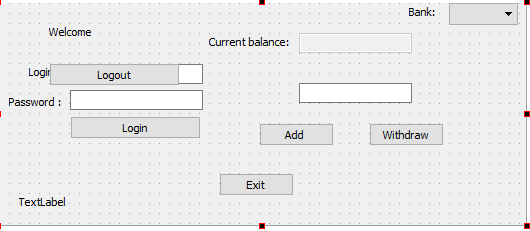
}else if(ui->bankBox->currentIndex()==0)

ui->txtBalance->clear();

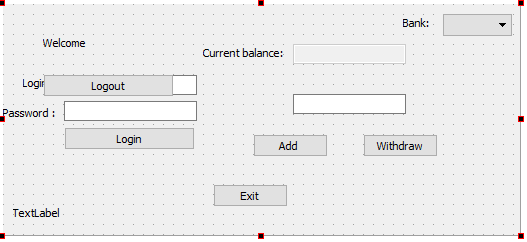
}

**Формы:**

****

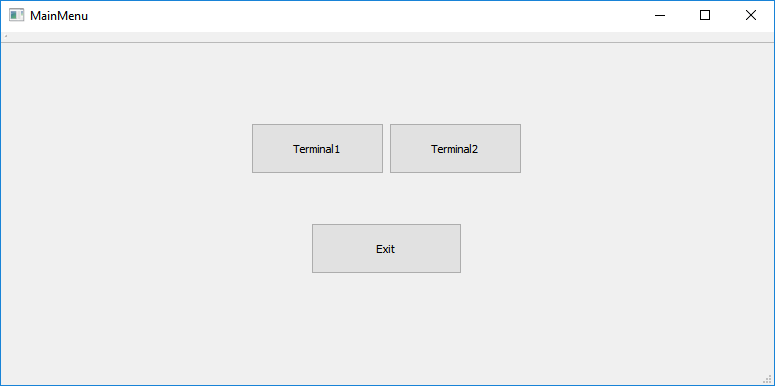
**Рисунок1.1** mainwindow.ui  
  


**Рисунок1.2** terminal1.ui

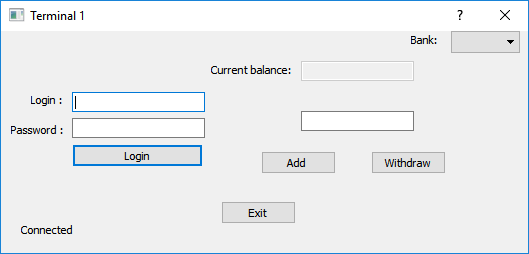


**Рисунок 1.3** terminal2.ui

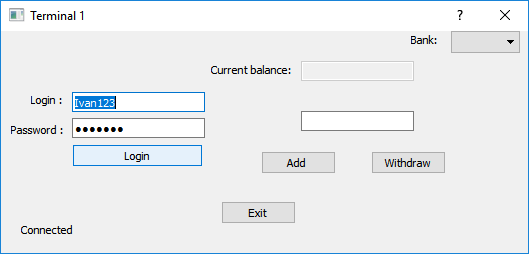
**Скриншоты программы:**



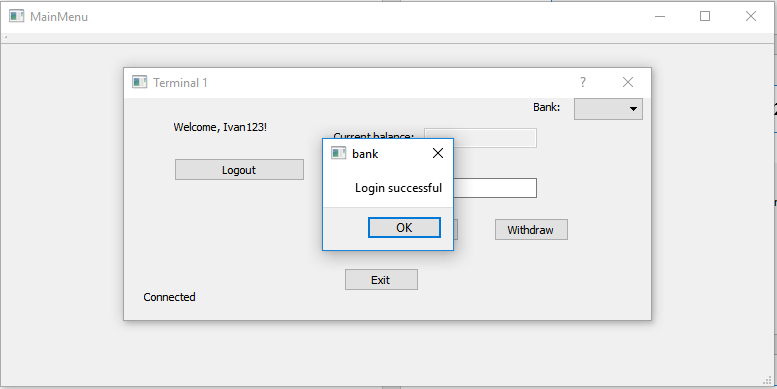
**Рисунок 2.1** MainMenu



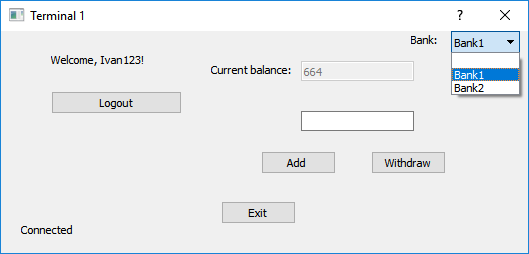
**Рисунок 2.2** Меню Terminal1



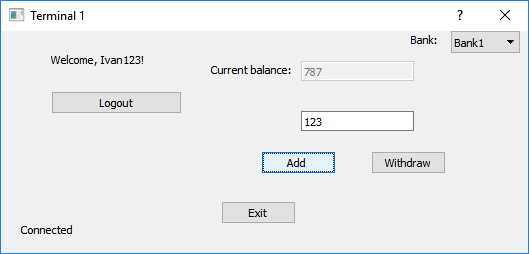
**Рисунок 2.3** Ввод логина/пароля.



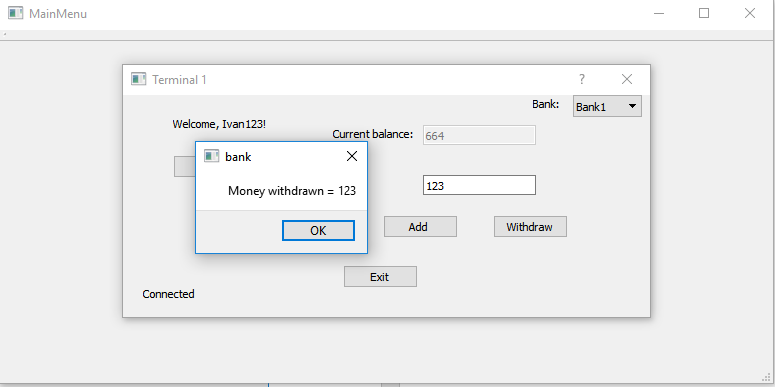
**Рисунок 2.4** Успешная авторизация.



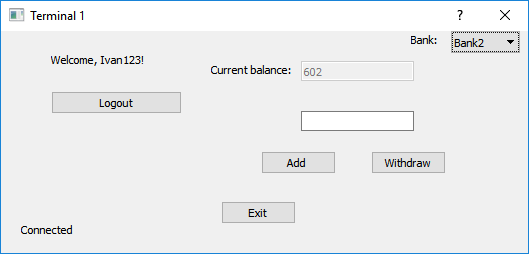
**Рисунок 2.5** Выбор банка, получение текущего баланса.



**Рисунок 2.6** Добавление средств на счет.

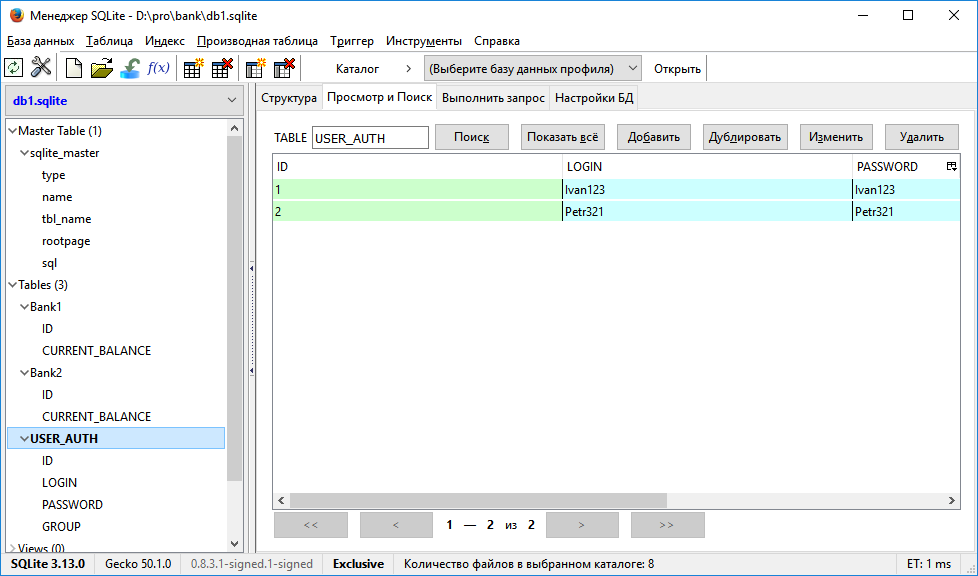


**Рисунок 2.7** Снятие средств.



**Рисунок 2.8** Баланс в Bank2.

**База данных db1.sqlite:**



**Вывод.**

При выполнении данной контрольной работы я закрепил практические навыки использования архитектурных принципов реализации параллельной обработки в вычислительных машинах и методов конструирования параллельных программ.