



Министерство науки и высшего образования Российской Федерации  
Калужский филиал  
федерального государственного бюджетного  
образовательного учреждения высшего образования  
«Московский государственный технический университет имени Н.Э. Баумана  
(национальный исследовательский университет)»  
(КФ МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУК «Информатика и Управление»

КАФЕДРА ИУК4 «Программное обеспечение ЭВМ, информационные технологии»

## ЛАБОРАТОРНАЯ РАБОТА №2

ДИСЦИПЛИНА: «Кроссплатформенная разработка ПО»

Выполнил: студент гр. ИУК4-62Б \_\_\_\_\_ (Калашников А. С.)  
(Подпись) (Ф.И.О.)

Проверил: \_\_\_\_\_ (Пчелинцева Н. Н.)  
(Подпись) (Ф.И.О.)

Дата сдачи (защиты):

Результаты сдачи (защиты):

- Балльная оценка:

- Оценка:

Калуга, 2023

**Цель:** Получить навык разработки приложения с использованием фреймворка Qt.

**Задачи:**

1. Ознакомиться со средой разработки Qt Creator.
2. Получить навыки написания программ с использованием фреймворка Qt.

**Задание:** В качестве предметной области использовать вариант из лабораторной работы №1. Разработать приложение с графическим интерфейсом на основе фреймворка Qt. Приложение должно реализовывать функционал из предыдущей работы. В приложение добавить возможность авторизации пользователей. Приложение должно работать на разных платформах (Linux, Windows).

**Код:**

**Main.py**

```
import datetime
import sys
from PyQt5 import uic
from PyQt5 import QtWidgets
from PyQt5.QtWidgets import QDialog, QApplication
from PyQt5.uic import loadUi
from re import sub
from decimal import Decimal
from PyQt5.QtWidgets import QMessageBox
from datetime import date
from Autho import *
from autho_db import *
from patient_db import *
from contract_db import *
from doctor_db import *
from patient_card_db import *
from Menu import *
from Contract import *
from ContractAdd import *
from Doctor import *
from DoctorAdd import *
from Patient import *
from PatientAdd import *
from PatientCard import *
from PatientCardAdd import *
from dateutil import parser

class Menu(QtWidgets.QMainWindow):
    def __init__(self, parent=None):
        super().__init__(parent)
        self.ui = Ui_MainWindow()
        self.ui.setupUi(self)
        self.text = ''
        self.ui.comboTable.addItem("Пациенты")
        self.ui.comboTable.addItem("Доктора")
        self.ui.comboTable.addItem("Контракты")
        self.ui.comboTable.addItem("Карты пациентов")
        self.ui.comboTable.currentTextChanged.connect(self.set_new_data)
        self.ui.listWidget.itemClicked.connect(self.edit_table)
        self.ui.buttonAdd.clicked.connect(self.add_data)

    def edit_table(self, item):
```

```

if "Пациенты" == self.text:
    patient_str = item.text()
    patient_list = patient_str.split(" ")
    menu.close()
    patient.id = patient_list[1]
    search_patient = get_data_patient_id(patient.id)
    patient.set_new_data(search_patient)
    patient.show()
elif "Доктора" == self.text:
    doctor_str = item.text()
    doctor_list = doctor_str.split(" ")
    menu.close()
    print(doctor_list)
    doctor.id = doctor_list[1]
    search_doctor = get_data_doctor_id(doctor.id)
    doctor.set_new_data(search_doctor)
    doctor.show()
elif "Контракты" == self.text:
    contract_str = item.text()
    contract_list = contract_str.split(" ")
    print(contract_list)
    menu.close()
    contract.id = contract_list[1]
    search_contract = get_data_contract_id(contract.id)
    contract.set_new_data(search_contract)
    contract.add_patient_card()
    contract.show()

else:
    patient_card_str = item.text()
    patient_card_list = patient_card_str.split(" ")
    print(patient_card_list)
    menu.close()
    patientCard.id = patient_card_list[1]
    search_patient_card = get_data_patient_card_id(patientCard.id)
    patientCard.set_new_data(search_patient_card)
    patientCard.show()

def add_data(self):
    menu.close()
    if "Пациенты" == self.text:
        patientAdd.show()
    elif "Доктора" == self.text:
        doctorAdd.show()
    elif "Контракты" == self.text:
        contractAdd.show()
        contractAdd.add_patient()
    else:
        patientCardAdd.show()
        patientCardAdd.add_contract()
        patientCardAdd.add_doctor()

def set_new_data(self, text):
    if "Пациенты" == text:
        self.text = text
        self.ui.listWidget.clear()
        data_list = get_data_patient()
        for patient_data in data_list:
            date = patient_data[2].strftime("%m %d %Y")
            str_patient_data = "Id: "+str(patient_data[0])+" \nИмя: " +

```

```

patient_data[1] + " \nДата: " + date + " \nТелефон: " + patient_data[3]
self.ui.listWidget.addItem(str_patient_data)

elif "Доктора" == text:
    self.text = text
    data_list = get_data_doctor()
    self.ui.listWidget.clear()
    for doctor_data in data_list:
        str_doctor_data = "Id: "+str(doctor_data[0]) + " \nИмя: " +
doctor_data[1] + " \nКоэффициент: " + str(doctor_data[2]) + " \nДолжность:" +
doctor_data[3] \
                                + " \nТелефон: " + doctor_data[4]
        self.ui.listWidget.addItem(str_doctor_data)

elif "Контракты" == text:
    self.text = text
    self.ui.listWidget.clear()
    data_list = get_data_contract()

    for contract_data in data_list:
        list =
get_data_patient_card_in_contract_name_proc(str(contract_data[0]))
        list_price =
get_data_patient_card_in_contract_price(str(contract_data[0]))
        price=0
        for cost in list_price:
            price = price+cost[0]
        if price:
            update_data_price(str(contract_data[0]), price)
        if list:
            str_name_proc = ""
            for element in list:
                str_name_proc = str_name_proc + str(element[0]) + " "
        else:
            str_name_proc = ""
        date = contract_data[3].strftime("%m %d %Y")
        str_contract_data = "Id: "+str(contract_data[0]) + " \nНомер
контракта: " + contract_data[1] + " \nИтор: " + str(contract_data[2]) + "
\nДата:" + date \
                                + " \nСписок процедур: " + str_name_proc
        self.ui.listWidget.addItem(str_contract_data)

else:
    self.text = text
    self.ui.listWidget.clear()
    data_list = get_data_patient_card()
    for patient_card_data in data_list:
        date = patient_card_data[2].strftime("%m %d %Y")
        str_patient_card_data = "Id: "+str(patient_card_data[0])+"
\nНомер карты: " + patient_card_data[1] + " \nДата: " + date + " \nНазвание
процедуры: " + \
                                patient_card_data[3] + " \nЦена за услугу"
+ str(patient_card_data[4]) + " \nДоктор: " + str(patient_card_data[5]) + \
                                " \nКонтракт" + str(patient_card_data[6])
+ " \nИтоговая цена" + str(patient_card_data[7])
        self.ui.listWidget.addItem(str_patient_card_data)

```

```

class Doctor(QtWidgets.QMainWindow):

```

```

def __init__(self, parent=None):
    super().__init__(parent)
    self.ui = Ui_Doctor()
    self.ui.setupUi(self)
    self.id = ""
    self.ui.buttonAdd.clicked.connect(self.add_new_data)
    self.ui.buttonChange.clicked.connect(self.change)
    self.ui.buttonDelete.clicked.connect(self.delete)

def add_new_data(self):
    doctor.close()
    doctorAdd.show()

def change(self):
    doctor.close()
    update_data_doctor(self.id, self.ui.name.text(),
self.ui.ratio.text(),self.ui.post.text(), self.ui.phoneNumber.text())
    menu.set_new_data("Доктора")
    menu.show()

def delete(self):
    doctor.close()
    delete_data_doctor(self.id)
    menu.set_new_data("Доктора")
    menu.show()

def set_new_data(self,search_doctor):
    if search_doctor:
        for new_doctor in search_doctor:
            self.ui.name.setText(new_doctor[1])
            self.ui.ratio.setText(str(new_doctor[2]))
            self.ui.post.setText(new_doctor[3])
            self.ui.phoneNumber.setText(new_doctor[4])
class DoctorAdd(QMainWindow):
    def __init__(self, parent=None):
        super().__init__(parent)
        self.ui = Ui_DoctorAdd()
        self.ui.setupUi(self)
        self.ui.buttonBack.clicked.connect(self.back)
        self.ui.buttonAdd.clicked.connect(self.add_new_data)

    def back(self):
        doctorAdd.close()
        menu.show()

    def add_new_data(self):
        doctorAdd.close()
        set_data_doctor(self.ui.name.text(),
self.ui.ratio.text(),self.ui.post.text(), self.ui.phoneNumber.text())
        menu.set_new_data("Доктора")
        menu.show()
class Patient(QMainWindow):
    def __init__(self, parent=None):
        super().__init__(parent)
        self.ui = Ui_Patient()
        self.ui.setupUi(self)
        self.id = ""
        self.ui.buttonAdd.clicked.connect(self.add_new_data)
        self.ui.buttonChange.clicked.connect(self.change)
        self.ui.buttonDelete.clicked.connect(self.delete)

    def add_new_data(self):

```

```

        patient.close()
        patientAdd.show()

    def change(self):
        patient.close()
        day = parser.parse(self.ui.birthDate.text())
        update_data_patient(self.id, self.ui.name.text(), day,
self.ui.phoneNumber.text())
        menu.set_new_data("Пациенты")
        menu.show()

    def delete(self):
        patient.close()
        delete_data_patient(self.id)
        menu.set_new_data("Пациенты")
        menu.show()

    def set_new_data(self, search_patient):
        if search_patient:
            for new_patient in search_patient:
                date = new_patient[2].strftime("%m %d %Y")
                self.ui.name.setText(new_patient[1])
                self.ui.phoneNumber.setText(new_patient[3])
                self.ui.birthDate.setText(date)

class PatientAdd(QtWidgets.QMainWindow):
    def __init__(self, parent=None):
        super().__init__(parent)
        self.ui = Ui_PatientAdd()
        self.ui.setupUi(self)
        self.ui.buttonBack.clicked.connect(self.back)
        self.ui.buttonAdd.clicked.connect(self.add_new_data)

    def back(self):
        patientAdd.close()
        menu.show()

    def add_new_data(self):
        patientAdd.close()
        day = parser.parse(self.ui.birthDate.text())
        set_data_patient(self.ui.name.text(), day,
self.ui.phoneNumber.text())
        menu.set_new_data("Пациенты")
        menu.show()

class Contract(QtWidgets.QMainWindow):
    def __init__(self, parent=None):
        super().__init__(parent)
        self.ui = Ui_Contract()
        self.ui.setupUi(self)
        self.id = ""
        self.patient_str = ""
        self.total_cost = 0
        self.ui.totalCost.setText(str(self.total_cost))
        self.ui.buttonAdd.clicked.connect(self.add_new_data)
        self.ui.buttonChange.clicked.connect(self.change)
        self.ui.buttonDelete.clicked.connect(self.delete)

    def set_new_data(self, search_contract):
        if search_contract:
            for new_contract in search_contract:
                date = new_contract[3].strftime("%m %d %Y")

```

```

        self.ui.numberContract.setText(new_contract[1])
        self.ui.totalCost.setText(str(new_contract[2]))
        self.ui.createDate.setText(date)
        self.add_patient()
    def add_patient_card(self):
        self.ui.listWidget.clear()
        data_list = get_data_patient_card_in_contract(self.id)
        print(data_list)
        if data_list:
            for patient_card_data in data_list:
                date = patient_card_data[2].strftime("%m %d %Y")
                str_patient_card_data = "Id: "+str(patient_card_data[0])+"
\nНомер карты: " + patient_card_data[
                    1] + " \nДата: " + date + " \nНазвание процедуры: " + \
                    patient_card_data[3] + " \nЦена за
услрy" + str(
                        patient_card_data[4]) + " \nДоктор: " +
str(patient_card_data[5]) + \
                        " \nКонтракт" +
str(patient_card_data[6]) + " \nИтоговая цена" + str(patient_card_data[7])
                self.ui.listWidget.addItem(str_patient_card_data)
    def add_patient(self):
        data_patient = get_name_patient()
        if data_patient:
            for new_patient in data_patient:
                self.ui.comboPatient.addItem(str(new_patient[0])+"
"+new_patient[1])
    def add_new_data(self):
        contract.close()
        contractAdd.add_patient()
        contractAdd.show()

    def change(self):
        contract.close()
        day = parser.parse(self.ui.createDate.text())
        patient_str = self.ui.comboPatient.currentText()
        update_patient = patient_str.split(" ")
        update_data_contract(self.id, self.ui.numberContract.text(),
                                self.ui.totalCost.text(), day,
update_patient[0])
        menu.set_new_data("Контракты")
        menu.show()

    def delete(self):
        contract.close()
        delete_data_contract_patient_card(self.id)
        delete_data_contract(self.id)

        menu.set_new_data("Контракты")
        menu.show()

class ContractAdd(QtWidgets.QMainWindow):
    def __init__(self, parent=None):
        super().__init__(parent)
        self.ui = Ui_ContractAdd()
        self.ui.setupUi(self)
        self.id = ""
        self.patient_str = ""
        day = datetime.datetime.today()
        date = day.strftime("%m %d %Y")
        self.ui.createDate.setText(str(date))

```

```

        self.ui.totalCost.setText("0")
        self.ui.buttonBack.clicked.connect(self.back)
        self.ui.buttonAdd.clicked.connect(self.add_new_data)

    def add_patient(self):
        data_patient = get_name_patient()
        if data_patient:
            for new_patient in data_patient:
                self.ui.comboPatient.addItem(str(new_patient[0])+"
"+new_patient[1])
    def back(self):
        contractAdd.close()
        menu.show()

    def add_new_data(self):
        contractAdd.close()
        update_patient = self.patient_str.split(" ")
        day = parser.parse(self.ui.createDate.text())
        set_data_contract(self.ui.numberContract.text(),
self.ui.totalCost.text(),day, update_patient[0])
        menu.set_new_data("Контракты")
        menu.show()

class PatientCard(QtWidgets.QMainWindow):
    def __init__(self, parent=None):
        super().__init__(parent)
        self.ui = Ui_PatientCard()
        self.ui.setupUi(self)
        self.id = ""
        self.doctor_str = ""
        self.contract_str = ""
        self.ui.price.textChanged.connect(lambda:
self.set_total_price(self.doctor_str))
        self.ui.buttonAdd.clicked.connect(self.add_new_data)
        self.ui.buttonChange.clicked.connect(self.change)
        self.ui.buttonDelete.clicked.connect(self.delete)
        self.ui.comboDoctor.currentTextChanged.connect(self.set_total_price)

        self.ui.comboContract.currentTextChanged.connect(self.set_contract_id)

    def add_doctor(self):
        data_doctor = get_name_doctor()
        if data_doctor:
            for new_doctor in data_doctor:
                self.ui.comboDoctor.addItem(str(new_doctor[0])+"
"+new_doctor[1])

    def add_contract(self):
        data_contract = get_name_contract()
        if data_contract:
            for new_contract in data_contract:
                self.ui.comboContract.addItem(str(new_contract[0])+"
"+new_contract[1])
    def set_total_price(self, text):
        self.doctor_str = text
        str_id = text.split(" ")
        ratio = get_ratio_doctor_id(str_id[0])
        if ratio and self.ui.price.text():
            self.ui.totalPrice.setText(str(int(self.ui.price.text()) *
ratio[0]))
        else:

```



```

        self.ui.totalPrice.setText("0")
def set_contract_id(self, text):
    self.contract_str = text

def add_new_data(self):
    patientCard.close()
    patientCardAdd.show()

def change(self):
    patientCard.close()
    day = parser.parse(self.ui.createDate.text())
    update_doctor = self.doctor_str.split(" ")
    update_contract = self.contract_str.split(" ")
    update_data_patient_card(self.id, self.ui.numberCard.text(), day,
self.ui.nameProc.text(),

self.ui.price.text(), update_doctor[0], update_contract[0], self.ui.totalPrice.t
ext())
    menu.set_new_data("Карты пациентов")
    menu.show()

def delete(self):
    patientCard.close()
    delete_data_patient_card(self.id)
    menu.set_new_data("Карты пациентов")
    menu.show()

def set_new_data(self, search_patient_card):
    self.add_doctor()
    self.add_contract()
    if search_patient_card:
        for new_patient_card in search_patient_card:
            date = new_patient_card[2].strftime("%m %d %Y")
            self.ui.numberCard.setText(new_patient_card[1])
            self.ui.createDate.setText(date)
            self.ui.nameProc.setText(new_patient_card[3])
            self.ui.price.setText(str(new_patient_card[4]))
            self.ui.comboDoctor.addItem(str(new_patient_card[5]))
            self.ui.comboContract.addItem(str(new_patient_card[6]))
            self.ui.totalPrice.setText(str(new_patient_card[7]))

class PatientCardAdd(QMainWindow):
    def __init__(self, parent=None):
        super().__init__(parent)
        self.ui = Ui_PatientCardAdd()
        self.ui.setupUi(self)
        self.id = ""
        self.doctor_str = ""
        self.contract_str = ""
        self.ui.price.setText("0")
        self.ui.buttonAdd.clicked.connect(self.add_new_data)
        self.ui.price.textChanged.connect(lambda:
self.set_total_price(self.doctor_str))
        self.ui.buttonBack.clicked.connect(self.back)
        self.ui.comboDoctor.currentTextChanged.connect(self.set_total_price)

self.ui.comboContract.currentTextChanged.connect(self.set_contract_id)

def back(self):
    patientCardAdd.close()
    menu.show()

```

```

def add_new_data(self):
    patientAdd.close()
    update_doctor = self.doctor_str.split(" ")
    update_contract = self.contract_str.split(" ")
    day = parser.parse(self.ui.createDate.text())
    set_data_patient_card(self.ui.numberCard.text(), day,
self.ui.nameProc.text(),
                                self.ui.price.text(), update_doctor[0],
update_contract[0], self.ui.totalPrice.text())
    menu.set_new_data("Карты пациентов")
    menu.show()
def add_doctor(self):
    data_doctor = get_name_doctor()
    if data_doctor:
        for new_doctor in data_doctor:
            self.ui.comboDoctor.addItem(str(new_doctor[0])+"
"+new_doctor[1])

def add_contract(self):
    data_contract = get_name_contract()
    if data_contract:
        for new_contract in data_contract:
            self.ui.comboContract.addItem(str(new_contract[0])+"
"+new_contract[1])
def set_total_price(self, text):
    self.doctor_str = text
    str_id = text.split(" ")
    ratio = get_ratio_doctor_id(str_id[0])
    if ratio and self.ui.price.text():

self.ui.totalPrice.setText(str(int(self.ui.price.text()*ratio[0])))
    else:
        self.ui.totalPrice.setText("0")
def set_contract_id(self, text):
    self.contract_str = text

class Autho(QtWidgets.QMainWindow):
    def __init__(self, parent=None):
        super().__init__(parent)
        self.ui = Ui_Autho()
        self.ui.setupUi(self)
        self.ui.pushButton.clicked.connect(self.authoriz)

    def authoriz(self):

        check = get_data_patient_for_autho(self.ui.login.text(),
self.ui.password.text())
        if check:
            authori.close()
            menu.show()
        else:
            msg = QMessageBox()
            msg.setIcon(QMessageBox.Critical)
            msg.setText("Авторизация:")
            msg.setInformativeText('Пользователя с таким номером или паролем
не существует')
            msg.setWindowTitle("Error")
            msg.exec_()

if __name__ == "__main__":
    name = ""

```

```

app = QtWidgets.QApplication(sys.argv)

authori = Autho()
menu = Menu()
contract = Contract()
contractAdd = ContractAdd()
doctor = Doctor()
doctorAdd = DoctorAdd()
patient = Patient()
patientAdd = PatientAdd()
patientCard = PatientCard()
patientCardAdd = PatientCardAdd()
authori.show()
sys.exit(app.exec_())

```

## Autho.py

```

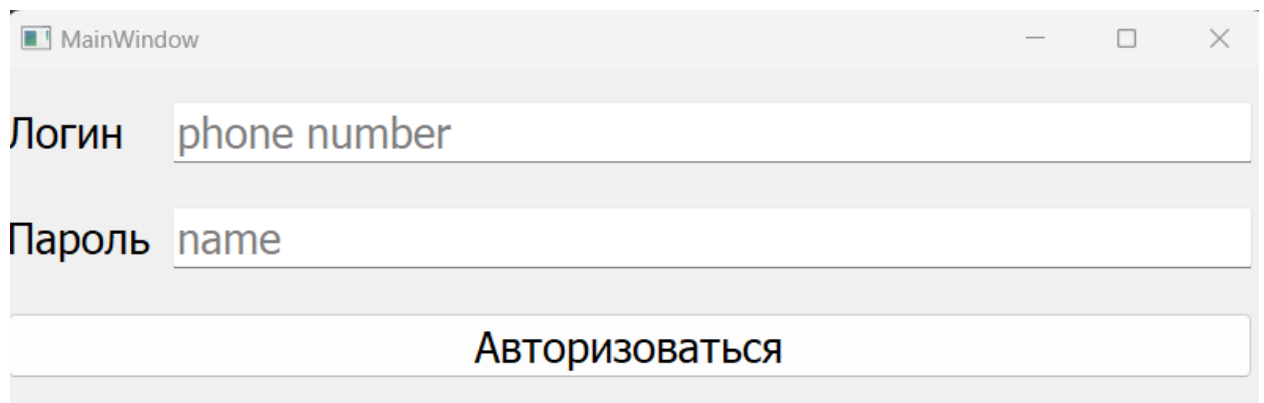
import psycopg2
from config import host, user, password, db_name

def get_data_patient_for_autho(phone_number,name):
    try:
        connection = psycopg2.connect(
            host=host,
            user=user,
            password=password,
            database=db_name
        )
        connection.autocommit = True
        # get data from a table
        with connection.cursor() as cursor:
            cursor.execute(
                """SELECT
                    id,
                    name
                FROM
                    patients
                Where
                    phone_number = '""'+ phone_number +""'
                    and
                    name = '""'+name+""';""
            )
            patient = cursor.fetchall()
            return patient

    except Exception as _ex:
        connection = False
        print("[INFO] Error while working with PostgreSQL", _ex)
    finally:
        if connection:
            connection.close()
            print("[INFO] PostgreSQL connection closed")

```

## Решение:



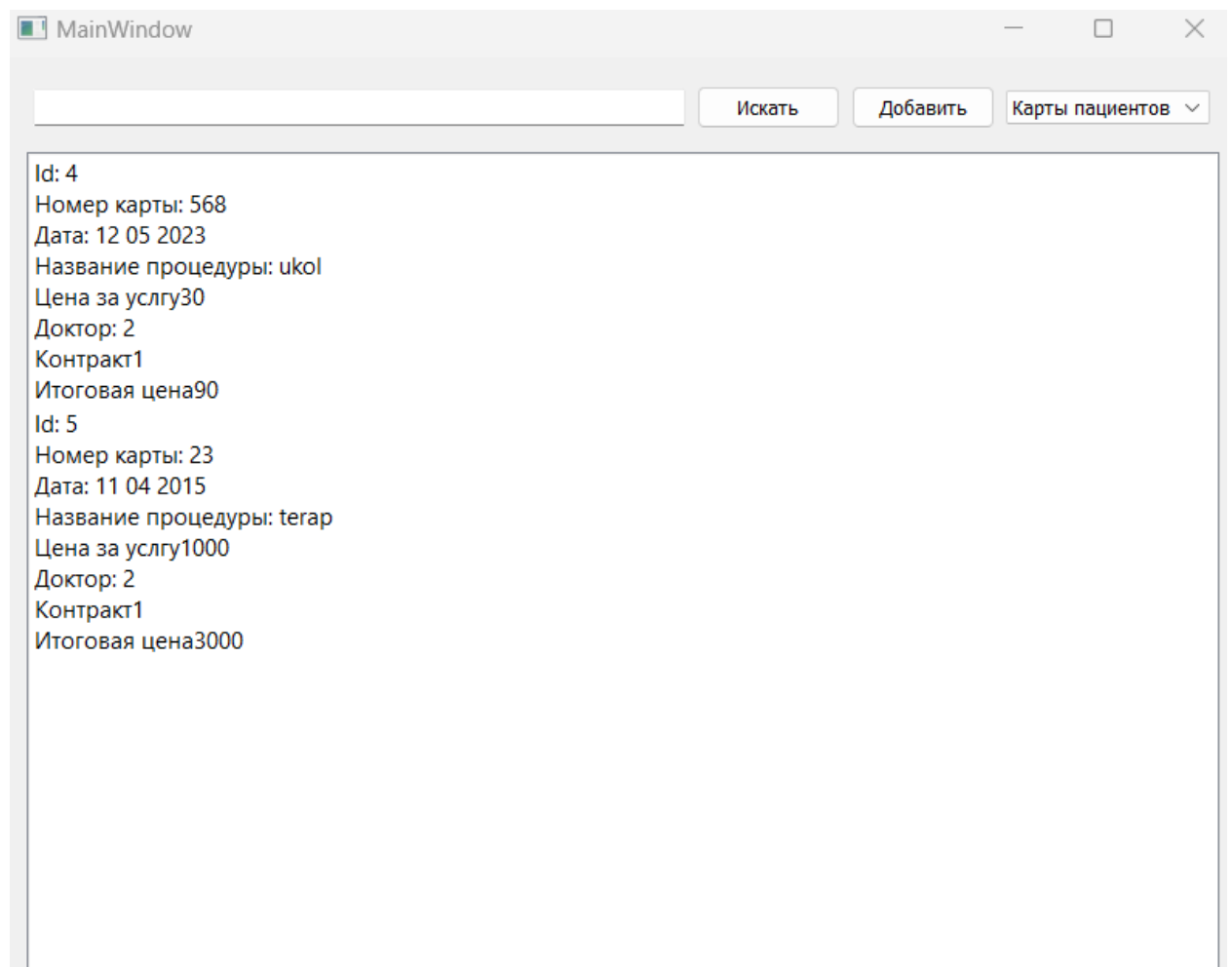
MainWindow

Логин phone number

Пароль name

Авторизоваться

**Рис. 1 Авторизация**



MainWindow

Искать Добавить Карты пациентов ▾

Id: 4  
Номер карты: 568  
Дата: 12 05 2023  
Название процедуры: ukoI  
Цена за услугу30  
Доктор: 2  
Контракт1  
Итоговая цена90

Id: 5  
Номер карты: 23  
Дата: 11 04 2015  
Название процедуры: terar  
Цена за услугу1000  
Доктор: 2  
Контракт1  
Итоговая цена3000

**Рис. 2 Вывод**

**Вывод:** в ходе выполнения лабораторной работы были получены практические навыки разработки приложения с использованием фреймворка Qt.