Федеральное государственное бюджетное образовательное учреждение высшего образования

Уфимский университет науки и технологий

Направление подготовки 09.03.04 —

«Программная инженерия»

Факультет информатики и робототехники Кафедра вычислительной математики и кибернетики

Курс II

Семестр IV

1 2 3 4 5 6 7 8 9 10

10

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

90

80

70

60

50

40

30

20

10

0

# Пояснительная записка к курсовому проекту

Тема: Проектирование и конструирование программной системы «Программная система библиотеки»

|  |  |  |  |
| --- | --- | --- | --- |
| (обозначение документа) | | | |
| Группа ПРО-234Б |  |  |  |
| Студент | Козлов И. А. |  |  |
|  | (Фамилия И.О.) | (дата) | (подпись) |
| Консультант |  |  |  |
|  | (Фамилия И.О.) | (дата) | (подпись) |
| Руководитель |  |  |  |
|  | (Фамилия И.О.) | (дата) | (подпись) |
| Принял |  |  |  |
|  | (Фамилия И.О.) | (дата) | (подпись) |

Уфа - 2023

Федеральное государственное бюджетное образовательное учреждение высшего образования

Уфимский университет науки и технологий

«Кафедра вычислительной математики и кибернетики»

ЗАДАНИЕ

на курсовой проект

|  |  |  |
| --- | --- | --- |
| Студент  Козлов И.А. | Группа  ПРО-234Б | Руководитель  Ризванов Д.А. |

1. Тема курсовой работы

Проектирование и конструирование программной системы «Программная система библиотеки»

1. Основное содержание

Введение, анализ решаемой проблемы, описание назначения системы и ограничений, определение архитектуры системы и описание ее основных элементов, их взаимосвязей, размещения элементов

Обоснование и выбор подхода к проектированию программной системы Разработка моделей, содержащих статическое и динамическое описание программной системы.

Программная реализация системы

1. Требования к оформлению
   1. Пояснительная записка должна быть оформлена в соответствии с требованиями

ЕСКД, ЕСПД, ГОСТ, СТП, др.

* 1. Графическая часть должна содержать

Модели, диаграммы, соответствующие выбранному подходу к созданию программных систем

Дата выдачи 10.03.2023 Дата окончания 31.05.2023

Руководитель

(подпись)

Курсовой проект

стр. , ил. , табл. .

Аnnotation

The paper considers the main automated library systems and designed a specific system using modeling and working with databases.

Introduction

A software system is a system of intercommunicating components based on software forming part of a computer system (a combination of hardware and software). It "consists of a number of separate programs, configuration files, which are used to set up these programs, system documentation, which describes the structure of the system, and user documentation, which explains how to use the system".

The development of library activities is inseparable from the introduction of advanced technologies and processes in order to provide the highest quality service to readers. The introduction of automated information systems is not only a trend of the times, but also a good basis for increasing the productivity and quality of work of library staff, an effective way to provide readers with the data and information they need in a timely manner.

Analysis

Let's analyze the existing library systems.

Analysis of the program "Library" (author A.V. Sosnin) [1]. The main features of the program:

• The "Library" program automates the library of a higher educational institution, includes a card index (catalog cards), a registration and inventory book, a summary book (arrival and departure from the fund, automatic calculation of the fund movement), distribution of books by faculties, archive, composite search ( includes autofilters, lists, booleans, etc.), recycle bin, duplication, etc.

• The program "Library" is designed for its use in the network, ie. makes it possible to install the program on any machine included in the local area network.

• The program is intended both for the library staff themselves (filling in the catalog card, etc.) and for users (students, teachers, etc.). With an existing network within the university, the client part of the program is installed on users' computers, which allows them to view, filter and print a list of books they need without leaving their office.

• "Library" maintains an electronic library system, ie. an electronic version of the book can be attached to the catalog card stored in the database. To install the program, you need to install the Interbase Server 1.1 database.

Analysis of the program "Librarianship". The main features of the program: [1].

1. Program interface in Kazakh, Russian and Ukrainian.

2. Network option with an unlimited number of connections.

3. Full support for Kazakh, Russian, Ukrainian languages (spelling, search, reporting, classifiers).

4. Quick search for information in the database.

5. The system maintains the referential integrity of the infobase.

6. Installation on a working computer with minimum specifications. The main modules of library processes:

7. Compilation of catalogs (alphabetical, systematic). Printing of basic and additional bibliographic cards, book form.

8. Formation and accounting of the book fund, fund of magazines and newspapers, AED, temporary storage fund, etc. Inventory and non-inventory accounting of copies.

9. Maintaining books of total accounting, a diary of the library (their automatic filling), automatic formation of the LBC index, UDC.

10. Maintaining a reader database module, library attendance.

11. Posting book publications, leaving the fund.

12. Maximum search for information in the database (group, context, sliding, heuristic).

13. Export reports to Excel tables, Word documents.

14. Checking the spelling of the text of the Russian, Kazakh, Ukrainian languages.

15. Database administration. Export-import of the fund in UNIMARK format

16. Creation of a full-text version of the table of contents (content) of books, magazines and newspapers.

17. Book availability. Analysis of the library fund with the help of diagrams.

18. Support for barcoding technology.

19. Creation of an electronic library of books, articles and other documents.

20. Order of periodicals (PS-1 f., PS-7 f.).

Advantages:

• allows you to perform all the main tasks of accounting and acquisition of library collections;

• the ability to generate reports on the activities of the library;

• allows keeping records of library readers and borrowed literature;

• does not require installation of additional software;

• low cost.

Disadvantages:

• difficult to understand user interface;

• redundancy of data associated with the introduction of several copies of one edition into the database.

System Purpose and Limitations

In the conditions of automation of information and library services, the following tasks are solved:

1) There is a systematization of resources on different media, as well as the creation of special electronic systems for their processing, accounting, search.

2) Modern search systems are being introduced into the practice of servicing, ensuring efficiency and convenience when searching for information.

3) Information and library processes are automated, which leads to resource savings and increased service efficiency.

4) The range of services provided by the library is expanding in accordance with the needs of users.

5) The library gets the opportunity to improve the information culture of users.

6) The use of computer technology equips the library with new methods and forms of promoting books and reading.

Summing up, it should be noted that today, as a process, library automation has already moved to a fundamentally different level: the most important task of library automation is to build a new unified method of work in all areas - starting with the acquisition of funds and ending with the registration of lending and readers. So, at the moment, an automated library is an information resource that plays an important role not only for a single educational institution, but also for the region and even for the state as a whole. [2]

System architecture. The main elements of the system:

|  |  |
| --- | --- |
| Elements | Description |
| Library fund | A systematized set of documents corresponding to the tasks, type, profile of the library, as well as the information needs of its users and intended for the use and storage of documents for the entire time that they are of real or potential benefit to subscribers. The backbone element of the BF is the document. |
| Users (readers) | An association of individuals and legal entities with related information needs. The backbone element of the CP subsystem is the user |
| Library staff | The totality of all library employees (regardless of their qualifications, type of work performed and other parameters) that ensure the functioning of individual subsystems and the library as a whole. The backbone element of the BP subsystem is the librarian |
| Material and technical base | A set of material and technical means that ensure the conduct of library work related to user service, the acquisition, processing and storage of documents, the production and service work of the library as a whole, including the activities of its subsidiary laboratories and workshops. The backbone element of the MTB subsystem is the material and technical device |

The main links between the elements:

Library fund and Library staff - maintenance of the library fund.

Library Fund and Material and Technical Base - storage of data on the quantitative composition of the library fund.

Users (readers) and Library staff - receiving services for obtaining books from the library fund.

Users (readers) and Material and technical base - obtaining information about the library fund.

Library staff and Material and technical base - database administration.

Choosing an approach to designing a software system:

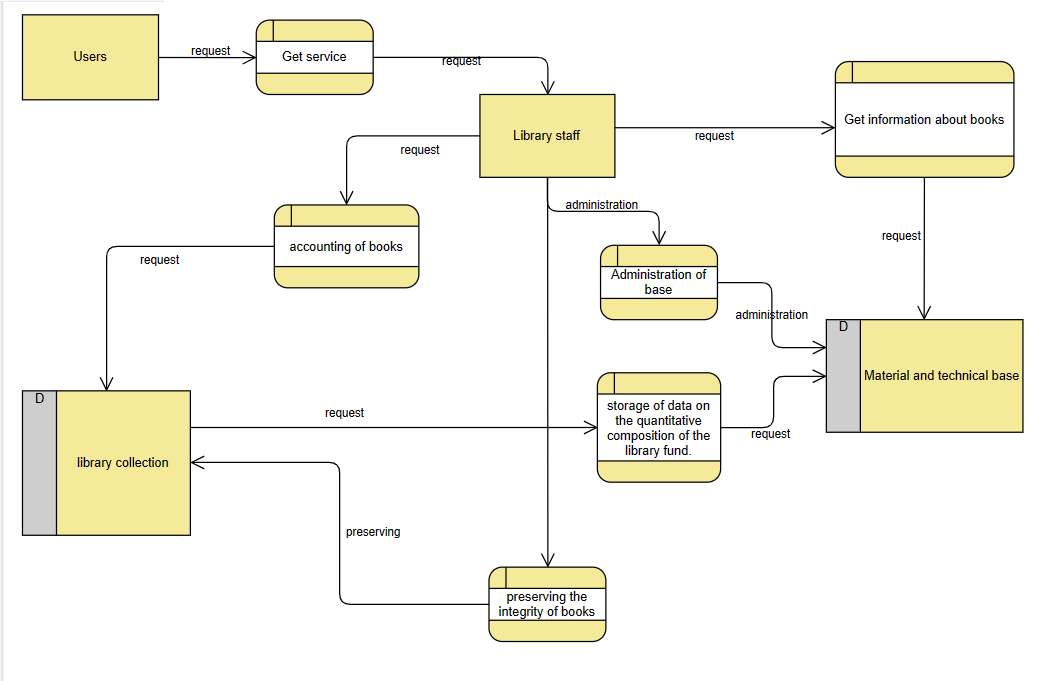
Top-down design (also known as top-down decomposition or incremental improvement) moves from a general statement about a process defined by a problem to be solved to progressively detailed statements about each specific task in that process.

Top-down design works well because it allows you to focus on fewer details at the same time. It is a logical technique that encourages organized development of the system and reduces the level of complexity (degrees of integration) at each of the subsequent stages of the project. For obvious reasons, top-down design is best when applied to problems that are clearly hierarchical in nature. Unfortunately, many of the real problems are not hierarchical. A top-down design also has other limitations that will become apparent when developing and maintaining large software systems.

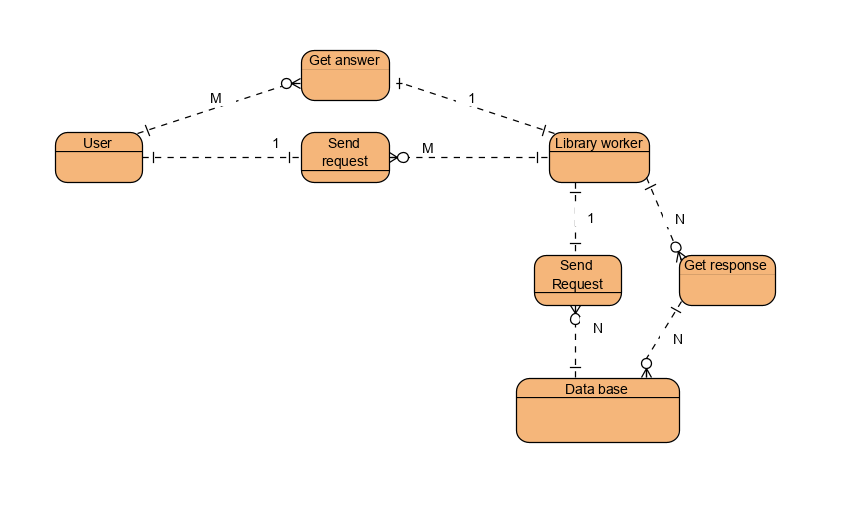
Top-down functional design does not adapt well to the development of large (large) software systems. Top-down design remains a useful paradigm for small programs and custom algorithms... but it doesn't scale very well to large systems. The point is not that you can't develop a system from the top down: you can. But by trading short-term convenience for long-term inflexibility, you are incorrectly piling up one function over another and (quite often) a functional interface over more important system parameters. You lose sight of the data aspect and you sacrifice reusability.

It is because of the usefulness of top-down functional programming in relation to small programs and individual algorithms that we will choose this option.

System Models



*Figure 1 Data-flow diagram*



*Figure 2 Entity-relationship diagram*

Implementation of the software system

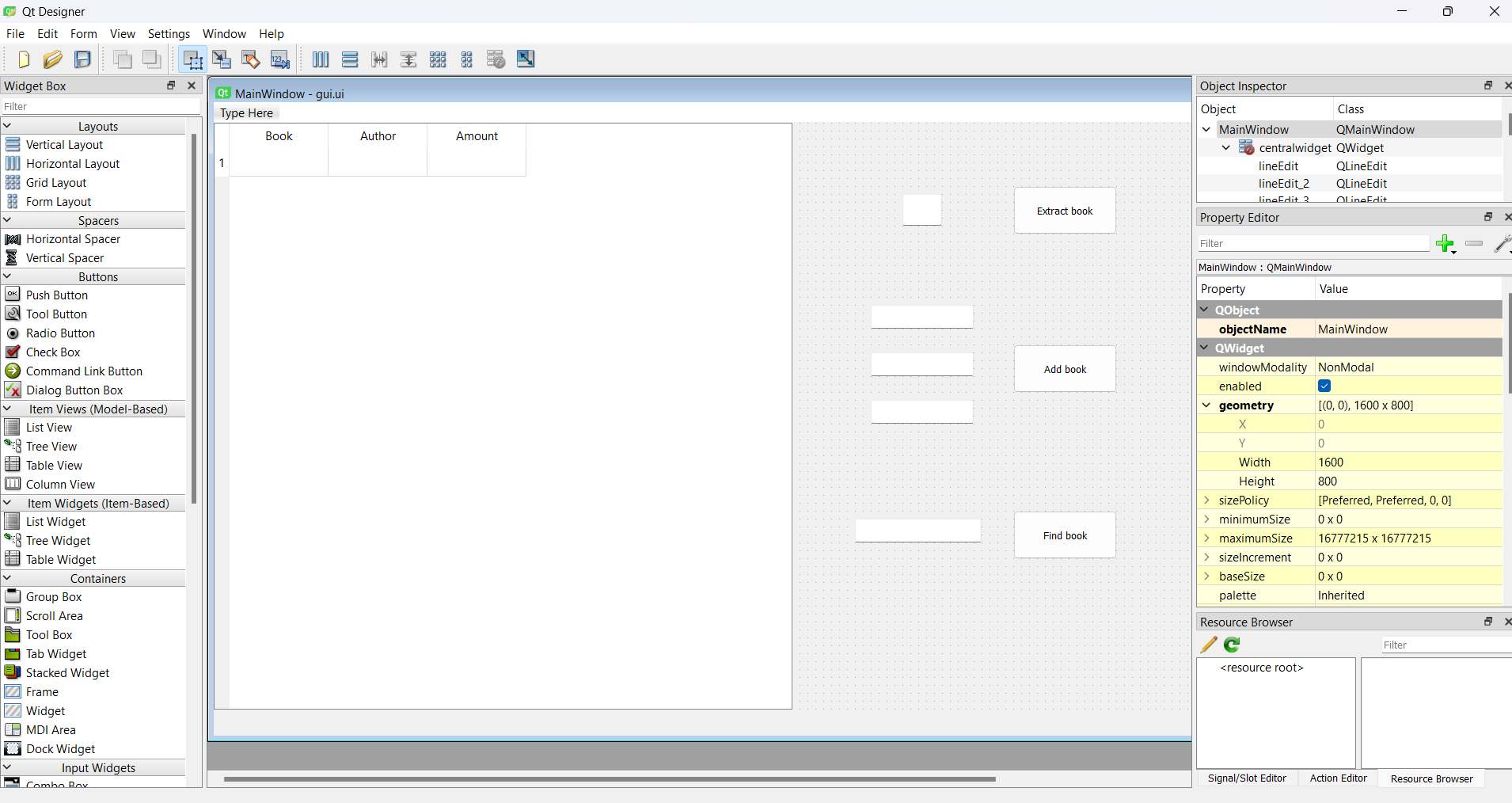
PyQt is a Python library for creating GUI applications using the Qt toolkit. Created by Riverbank Computing, PyQt is free software (GPL licensed) and has been in development since 1999. PyQt5 was released in 2016 and last updated in October 2021.

This complete PyQt5 tutorial takes you from first concepts to building fully-functional GUI applications in Python. It requires some basic Python knowledge, but no previous familiarity with GUI concepts. Everything will be introduced step by by step, using hands-on examples.

PyQt5 is the Qt5-based edition of the Python GUI library PyQt from Riverbank Computing.

We will use PyQt5 for making our desktop application.

Firstly we made the GUI part of our application:



Methods setupUi() and return\_lines\_edit() of Ui\_MainWindow class is used for getting access to buttons and lineedits of the GUI.

After that we have 3 functions which are connected to 3 buttons:

1) *def add\_book(self):*

*book = self.lines[1].text()*

*author = self.lines[2].text()*

*amount = self.lines[3].text()*

*self.put\_data\_into\_db(book, author, amount)*

2) *def extract\_from\_library(self):*

*if len(self.lines[0].text()):*

*self.delete\_row(self.lines[0].text())*

3) *def find\_book(self):*

*request = self.lines[4].text()*

*temp = []*

*count = 0*

*for i in range(len(self.records)):*

*if request == self.records[i][1] or request == self.records[i][2]:*

*print(self.records[i])*

*temp.append(self.records[i])*

*count = count + 1*

*print(temp)*

*self.all\_elements[0].setRowCount(len(temp))*

*for i in range(len(temp)):*

*self.all\_elements[0].setItem(i, 0, QTableWidgetItem(str(temp[i][0])))*

*self.all\_elements[0].setItem(i, 1, QTableWidgetItem(temp[i][1]))*

*self.all\_elements[0].setItem(i, 2, QTableWidgetItem(temp[i][2]))*

*self.all\_elements[0].setItem(i, 3, QTableWidgetItem(str(temp[i][3])))*

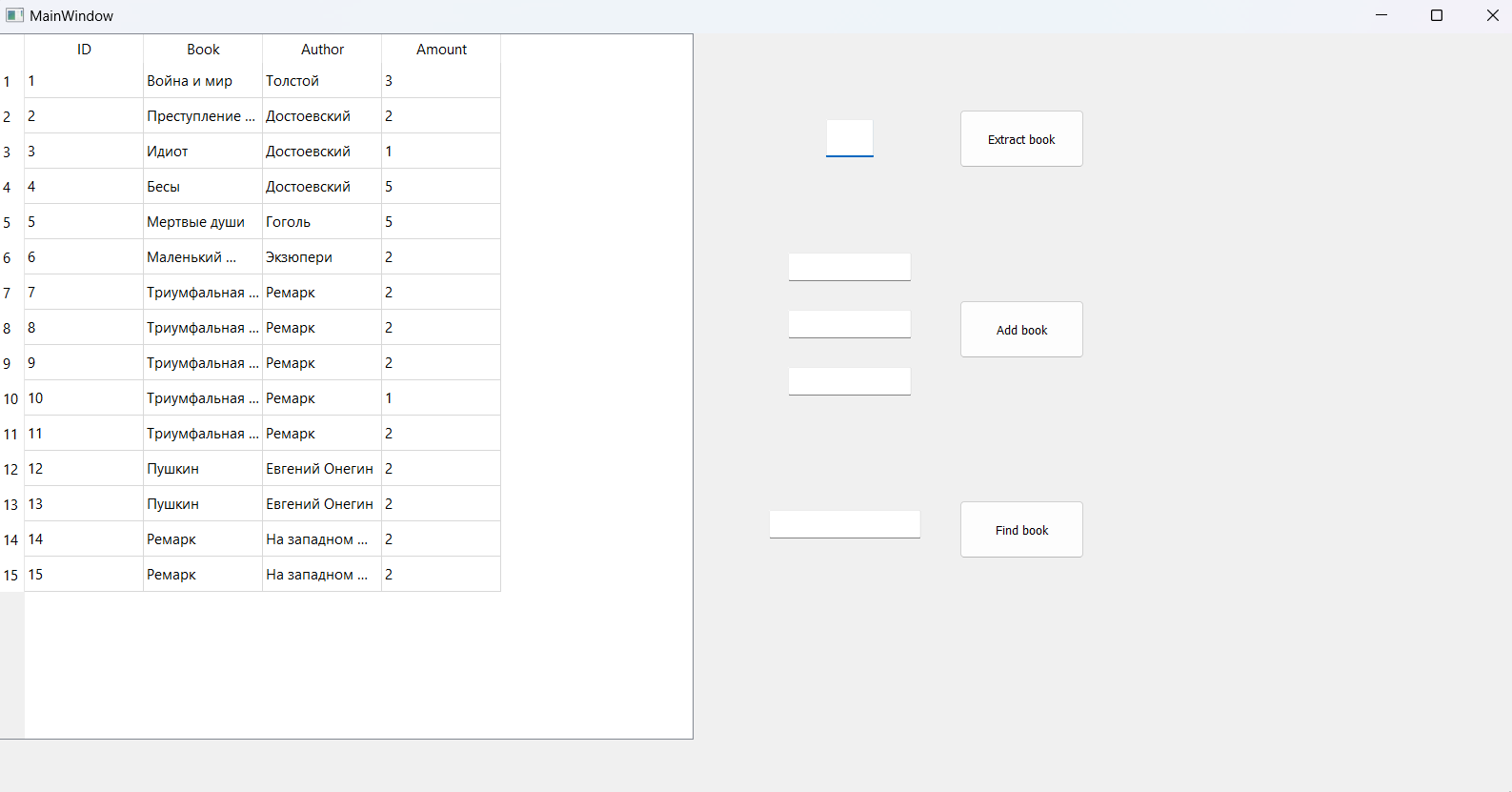
And finally we have 4 functions upload\_data(), update(), put\_data\_into\_db(), delete\_row() for working with database.

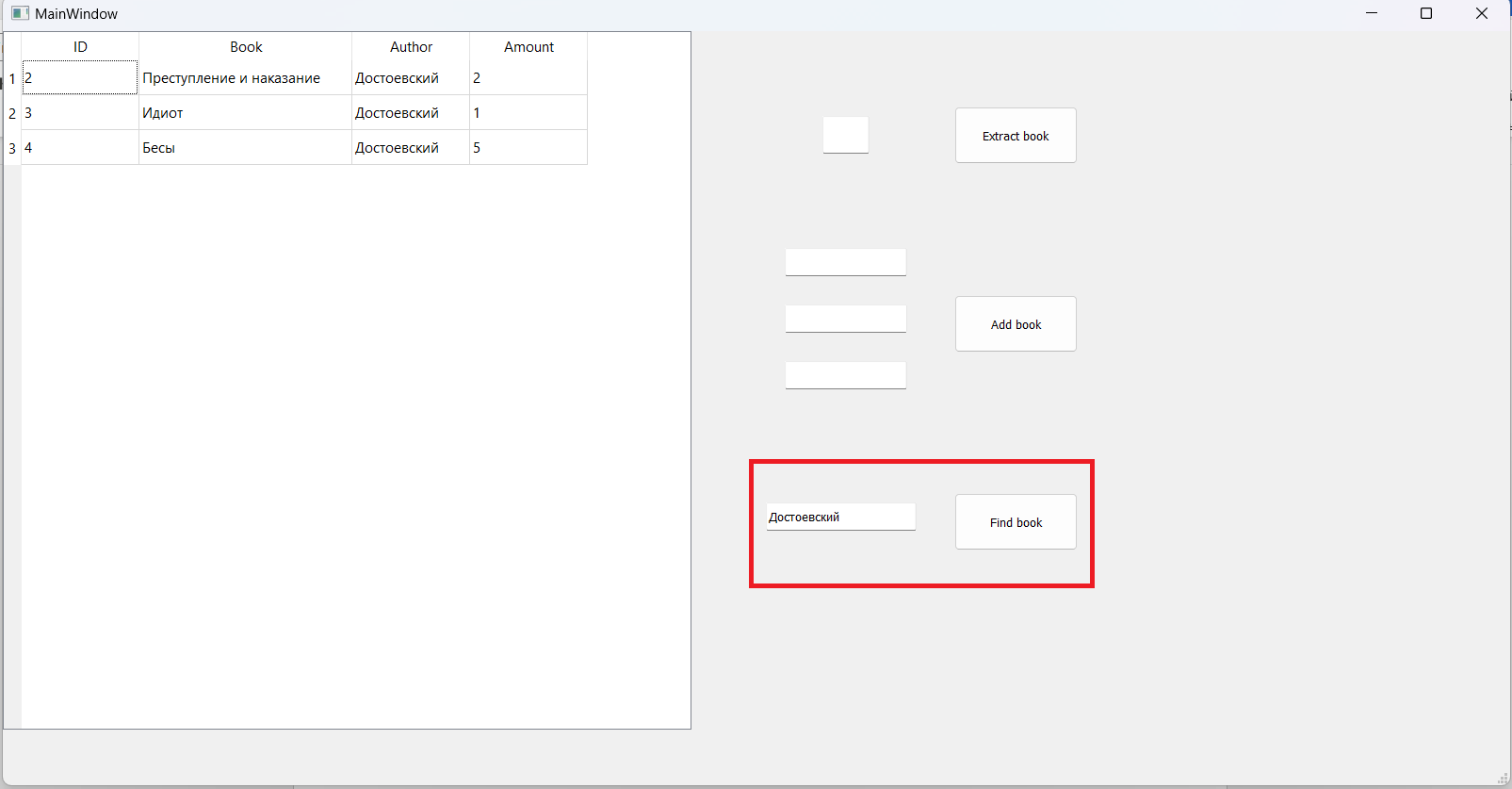
By the way, our database works on SQLite technology:

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Working of my program:





Conclusion: An automated information system is a set of hardware and software tools that implement the ideas and methods of automation. Modern AIS systems allow integrating various software around them, forming a single information system. Thanks to the introduction of automated information systems in various fields of activity, they contribute to the efficiency of labor activity, increase the throughput of systems, and reduce erroneous actions. The automated system supports all operations related to accounting, processing and provision of the necessary data. This system will help in the formation of final reports on data from this database for a user-specified period of time and provides input, deletion, storage and editing of information contained in data tables.

References:

1. В. В. Наумов, А. С. Классен. Анализ распространенных систем автоматизации библиотечного учёта [Электронный ресурс]. Link: <https://articlekz.com/article/13517>.
2. Севрюкова, А. А. Автоматизированные библиотечные системы / А. А. Севрюкова. — Текст : непосредственный // Молодой ученый. — 2015. — № 13 (93). — С. 857-859. — URL: https://moluch.ru/archive/93/20599/ (дата обращения: 07.06.2023).

Application 1

**main.py**

from PyQt5 import QtCore, QtGui, QtWidgets

from PyQt5.QtWidgets import QApplication, QMainWindow, QMessageBox, QTableWidgetItem

from PyQt5.QtGui import QPainter, QPen, QColor

from PyQt5.QtCore import Qt

import sqlite3

import sys

import gui

class window(QMainWindow):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.path = 'C://Users//vanyk//OneDrive//Рабочий стол//Учёба//ПИКПО//course.db'

self.records = []

self.initUI()

def initUI(self):

wind = gui.Ui\_MainWindow()

self.all\_elements = wind.setupUi(self)

self.all\_elements[1].clicked.connect(self.extract\_from\_library)

self.all\_elements[2].clicked.connect(self.add\_book)

self.all\_elements[3].clicked.connect(self.find\_book)

self.lines = wind.return\_lines\_edit()

self.show()

self.upload\_data()

def add\_book(self):

book = self.lines[1].text()

author = self.lines[2].text()

amount = self.lines[3].text()

self.put\_data\_into\_db(book, author, amount)

def extract\_from\_library(self):

if len(self.lines[0].text()):

self.delete\_row(self.lines[0].text())

def find\_book(self):

print ("find")

def upload\_data(self):

sqlite\_connection = sqlite3.connect(self.path)

cursor = sqlite\_connection.cursor()

sqlite\_select\_query = """SELECT \* from books"""

cursor.execute(sqlite\_select\_query)

self.records = cursor.fetchall()

self.all\_elements[0].setRowCount(len(self.records))

for i in range(len(self.records)):

self.all\_elements[0].setItem(i, 0, QTableWidgetItem(str(self.records[i][0])))

self.all\_elements[0].setItem(i, 1, QTableWidgetItem(self.records[i][1]))

self.all\_elements[0].setItem(i, 2, QTableWidgetItem(self.records[i][2]))

self.all\_elements[0].setItem(i, 3, QTableWidgetItem(str(self.records[i][3])))

def update(self, id, book, author, amount):

sqlite\_connection = sqlite3.connect(self.path)

cursor = sqlite\_connection.cursor()

sql\_update\_query = """Update books set name = ? where id = ?"""

data = (book, id)

cursor.execute(sql\_update\_query, data)

sql\_update\_query = """Update books set author = ? where id = ?"""

data = (author, id)

cursor.execute(sql\_update\_query, data)

sql\_update\_query = """Update books set amount = ? where id = ?"""

data = (amount, id)

cursor.execute(sql\_update\_query, data)

sqlite\_connection.commit()

self.upload\_data()

def put\_data\_into\_db(self, book, author, amount):

sqlite\_connection = sqlite3.connect(self.path)

cursor = sqlite\_connection.cursor()

id = len(self.records) + 1

for i in range(len(self.records)):

if i + 1 !=self.records[i][0]:

id = i + 1

break

sqlite\_insert\_query = """INSERT INTO books

(id, name, author, amount)

VALUES

(?, ?, ?, ?);"""

data = (id, book, author, amount)

cursor.execute(sqlite\_insert\_query, data)

sqlite\_connection.commit()

self.upload\_data()

def delete\_row(self, id):

sqlite\_connection = sqlite3.connect(self.path)

cursor = sqlite\_connection.cursor()

for i in range(len(self.records)):

if self.records[i][0] == int(id):

if self.records[i][3] > 1:

self.update(self.records[i][0], self.records[i][1], self.records[i][2], self.records[i][3]-1)

return

sql\_delete\_query = """DELETE from books where id = ?"""

data = (id,)

cursor.execute(sql\_delete\_query, data)

sqlite\_connection.commit()

self.upload\_data()

if \_\_name\_\_ == '\_\_main\_\_':

app = QApplication(sys.argv)

app\_1 = window()

sys.exit(app.exec\_())

**gui.py**

from PyQt5 import QtCore, QtGui, QtWidgets

class Ui\_MainWindow(object):

def setupUi(self, MainWindow):

MainWindow.setObjectName("MainWindow")

MainWindow.resize(1600, 800)

self.centralwidget = QtWidgets.QWidget(MainWindow)

self.centralwidget.setObjectName("centralwidget")

self.tableWidget = QtWidgets.QTableWidget(self.centralwidget)

self.tableWidget.setGeometry(QtCore.QRect(0, 0, 731, 741))

self.tableWidget.setObjectName("tableWidget")

self.tableWidget.setColumnCount(4)

self.tableWidget.setRowCount(1)

item = QtWidgets.QTableWidgetItem()

self.tableWidget.setVerticalHeaderItem(0, item)

item = QtWidgets.QTableWidgetItem()

self.tableWidget.setHorizontalHeaderItem(0, item)

item = QtWidgets.QTableWidgetItem()

self.tableWidget.setHorizontalHeaderItem(1, item)

item = QtWidgets.QTableWidgetItem()

self.tableWidget.setHorizontalHeaderItem(2, item)

item = QtWidgets.QTableWidgetItem()

self.tableWidget.setHorizontalHeaderItem(3, item)

self.pushButton = QtWidgets.QPushButton(self.centralwidget)

self.pushButton.setGeometry(QtCore.QRect(1010, 80, 131, 61))

self.pushButton.setObjectName("pushButton")

self.pushButton\_2 = QtWidgets.QPushButton(self.centralwidget)

self.pushButton\_2.setGeometry(QtCore.QRect(1010, 280, 131, 61))

self.pushButton\_2.setObjectName("pushButton\_2")

self.pushButton\_3 = QtWidgets.QPushButton(self.centralwidget)

self.pushButton\_3.setGeometry(QtCore.QRect(1010, 490, 131, 61))

self.pushButton\_3.setObjectName("pushButton\_3")

self.lineEdit = QtWidgets.QLineEdit(self.centralwidget)

self.lineEdit.setGeometry(QtCore.QRect(870, 90, 50, 40))

self.lineEdit.setObjectName("lineEdit")

self.lineEdit\_2 = QtWidgets.QLineEdit(self.centralwidget)

self.lineEdit\_2.setGeometry(QtCore.QRect(830, 230, 130, 30))

self.lineEdit\_2.setObjectName("lineEdit\_2")

self.lineEdit\_3 = QtWidgets.QLineEdit(self.centralwidget)

self.lineEdit\_3.setGeometry(QtCore.QRect(830, 290, 130, 30))

self.lineEdit\_3.setObjectName("lineEdit\_3")

self.lineEdit\_4 = QtWidgets.QLineEdit(self.centralwidget)

self.lineEdit\_4.setGeometry(QtCore.QRect(830, 350, 130, 30))

self.lineEdit\_4.setObjectName("lineEdit\_4")

self.lineEdit\_5 = QtWidgets.QLineEdit(self.centralwidget)

self.lineEdit\_5.setGeometry(QtCore.QRect(810, 500, 160, 30))

self.lineEdit\_5.setObjectName("lineEdit\_5")

MainWindow.setCentralWidget(self.centralwidget)

self.menubar = QtWidgets.QMenuBar(MainWindow)

self.menubar.setGeometry(QtCore.QRect(0, 0, 1600, 26))

self.menubar.setObjectName("menubar")

MainWindow.setMenuBar(self.menubar)

self.statusbar = QtWidgets.QStatusBar(MainWindow)

self.statusbar.setObjectName("statusbar")

MainWindow.setStatusBar(self.statusbar)

self.retranslateUi(MainWindow)

QtCore.QMetaObject.connectSlotsByName(MainWindow)

return self.tableWidget, self.pushButton, self.pushButton\_2, self.pushButton\_3

def return\_lines\_edit(self):

return self.lineEdit, self.lineEdit\_2, self.lineEdit\_3, self.lineEdit\_4, self.lineEdit\_5

def retranslateUi(self, MainWindow):

\_translate = QtCore.QCoreApplication.translate

MainWindow.setWindowTitle(\_translate("MainWindow", "MainWindow"))

item = self.tableWidget.verticalHeaderItem(0)

item.setText(\_translate("MainWindow", "1"))

item = self.tableWidget.horizontalHeaderItem(0)

item.setText(\_translate("MainWindow", "ID"))

item = self.tableWidget.horizontalHeaderItem(1)

item.setText(\_translate("MainWindow", "Book"))

item = self.tableWidget.horizontalHeaderItem(2)

item.setText(\_translate("MainWindow", "Author"))

item = self.tableWidget.horizontalHeaderItem(3)

item.setText(\_translate("MainWindow", "Amount"))

self.pushButton.setText(\_translate("MainWindow", "Extract book"))

self.pushButton\_2.setText(\_translate("MainWindow", "Add book"))

self.pushButton\_3.setText(\_translate("MainWindow", "Find book"))