

**ОТЧЕТ**

по курсовой работе

по специальности 09.02.03 «Программирование в компьютерных системах»

Выполнил студент гр. П1-18

Ларченко М. А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (подпись)

Проверил преподаватель

Попов В. Н.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(оценка)

Королев, 2021

**Оглавление**

[Введение 3](#_Toc97832982)

[1. Разработка базы данных 4](#_Toc97832983)

[1.1 Получение задания на конкретную разработку 4](#_Toc97832984)

[1.2 Документооборот 4](#_Toc97832985)

[1.3 Построение IDEF-модели 4](#_Toc97832986)

[1.5 Структура базы данных 7](#_Toc97832987)

[1.6 Резервное копирование и восстановление 9](#_Toc97832988)

[2. Разработка клиентского приложения 10](#_Toc97832989)

[2.1 Сценарий клиентского приложения 10](#_Toc97832990)

[2.2 Разработка пользовательского интерфейса программного средства 10](#_Toc97832991)

[2.3 Отладка и тестирование программного средства 12](#_Toc97832992)

[3. Методы и технологии защиты информации в базах данных 13](#_Toc97832993)

[3.1 Регламента работы с ВТ 13](#_Toc97832994)

[3.2 Список доступа к работе с компьютерами и базой данных 13](#_Toc97832995)

[3.3 Регламент разграничивающий доступ 13](#_Toc97832996)

[3.4 Аппаратные средства защиты информации 13](#_Toc97832997)

[3.5 Программные средства защиты информации 13](#_Toc97832998)

[3.6 Разграничения доступа средствами базы данных 13](#_Toc97832999)

[Заключение 14](#_Toc97833000)

[Список литературы и интернет-ресурсов 15](#_Toc97833001)

[Дневник практики 16](#_Toc97833002)

[Приложение 1. Техническое задание 17](#_Toc97833003)

[Приложение 2. Руководство оператора 23](#_Toc97833004)

[Приложение 3. Руководство программиста 25](#_Toc97833005)

[Приложение 4. Листинг кода 26](#_Toc97833008)

# Введение

Данный курсовой проект заключается в разработке парсера маркетплейса WildBerries, который должен иметь базу данных. Разработка парсера будет происходить на Python, SQLite.

# Разработка базы данных

## Получение задания на конкретную разработку

Задача состоит в проектировании и создании парсера для получения данных с маркетплейса WildBerries для последущего их отображения на сайте.

Необходимо разработать питон скрипт для скраппинга цен и остальной информации на сайте WildBerries. На вход подаются айдишники на сайте, а на выходе мы получаем базу данных с нужными нам данными.

Инструменты для реализации серверной части: SQLite [1].

Инструменты для реализации клиентской части:

* Python [2] + Qt6 [3].

## Документооборот

Документооборот в отделе продаж и реализации товаров представляет из себя айди маркетплейса (на вход), а по окончанию процесса – база данных цен (на выход).

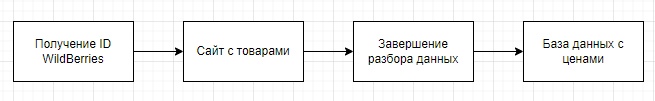


Рисунок 1. Схема документооборота

## Построение IDEF-модели

Ниже приведены IDEF-модели подразделения:

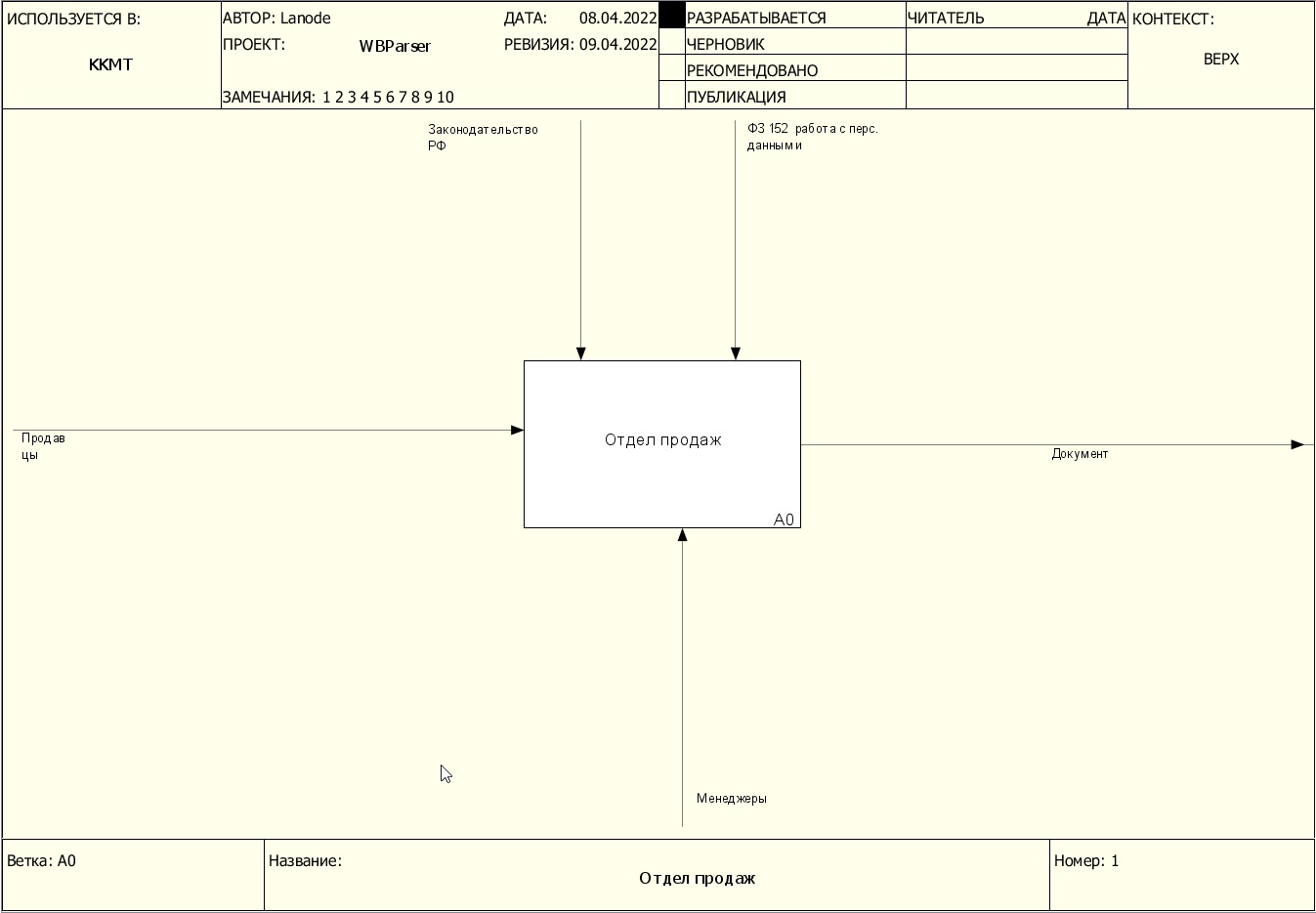


Рисунок 2. IDEF-модель 1-й уровень

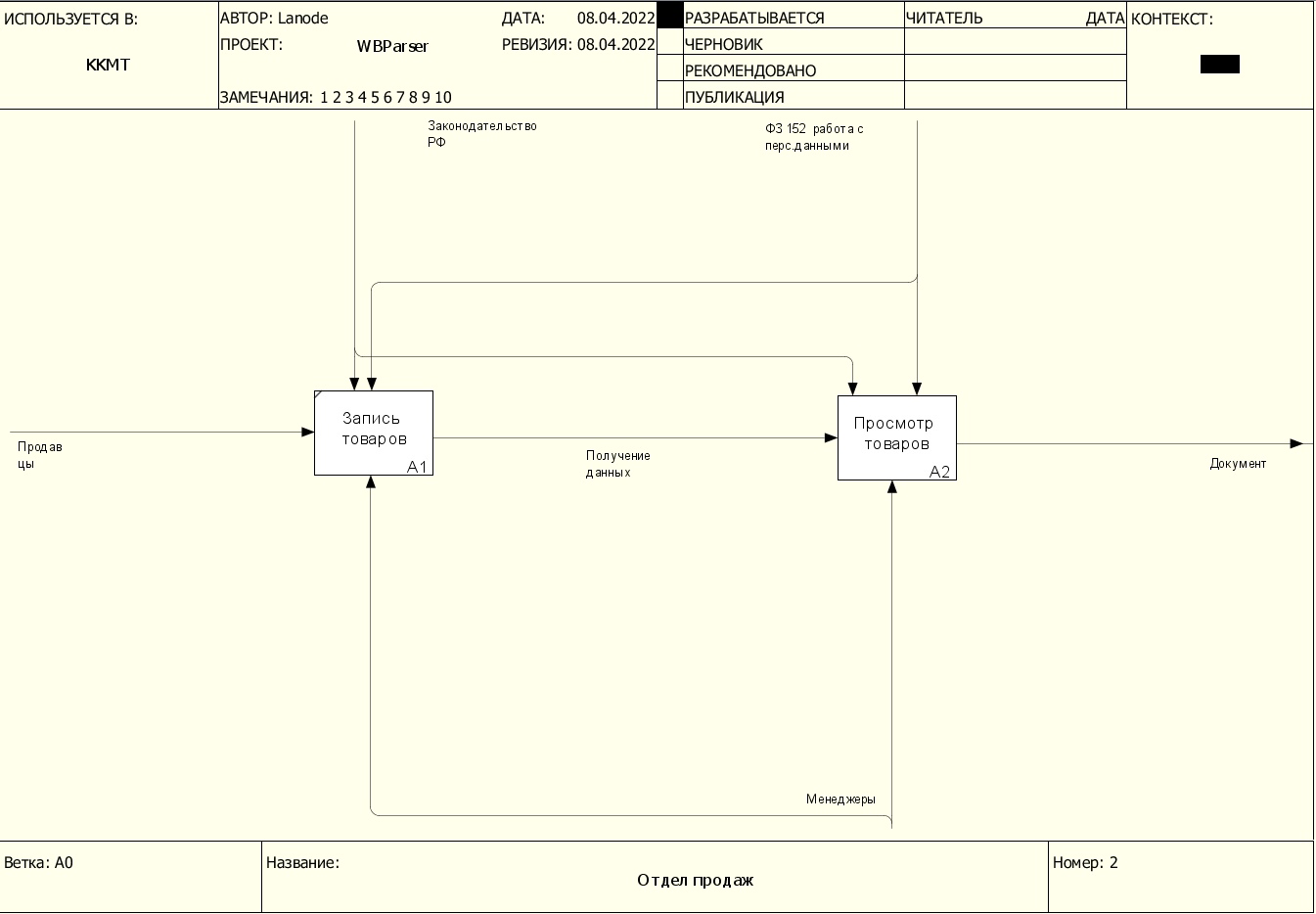


Рисунок 3. IDEF-модель 2-й уровень

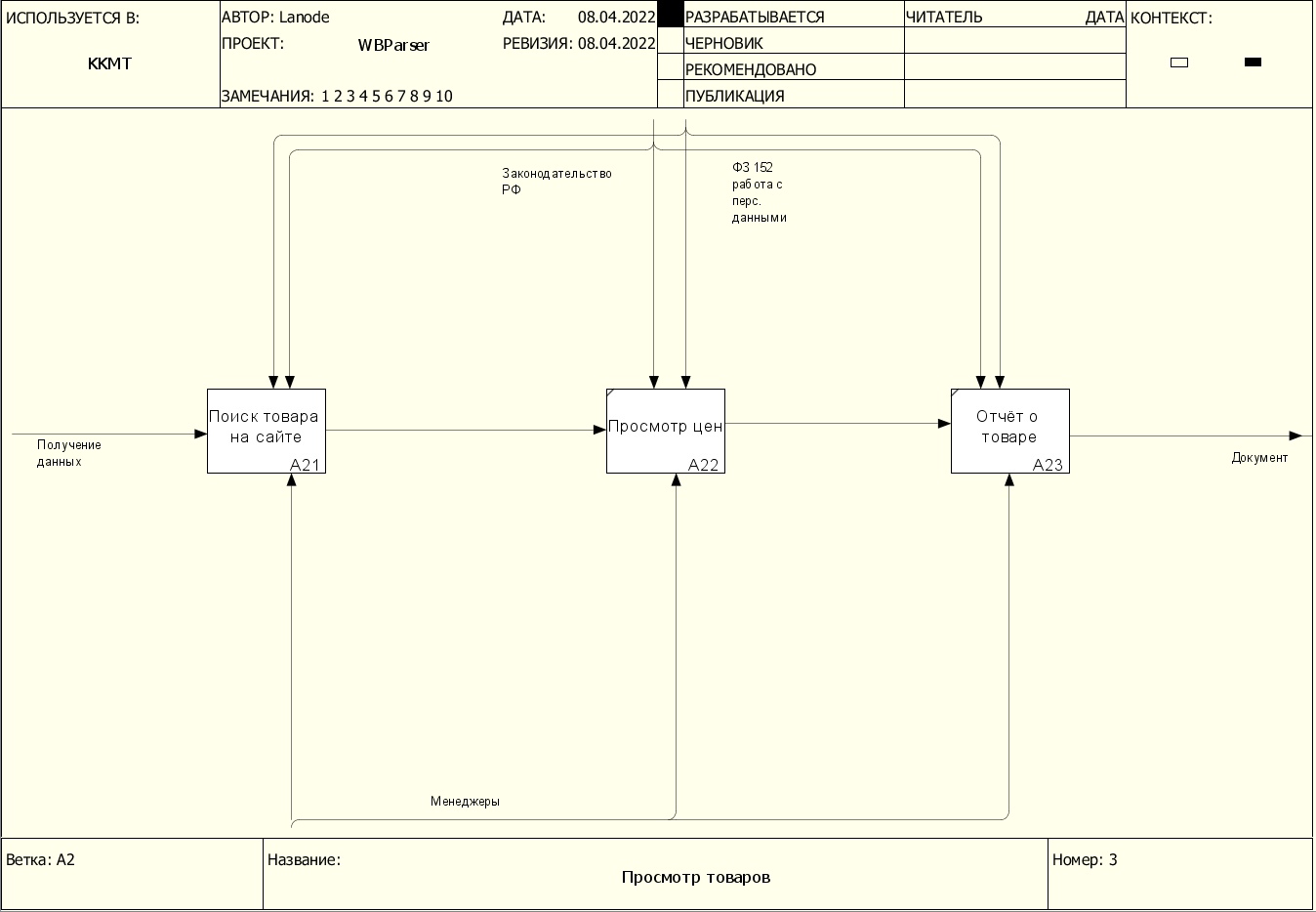


Рисунок 4. IDEF-модель 3-й уровень

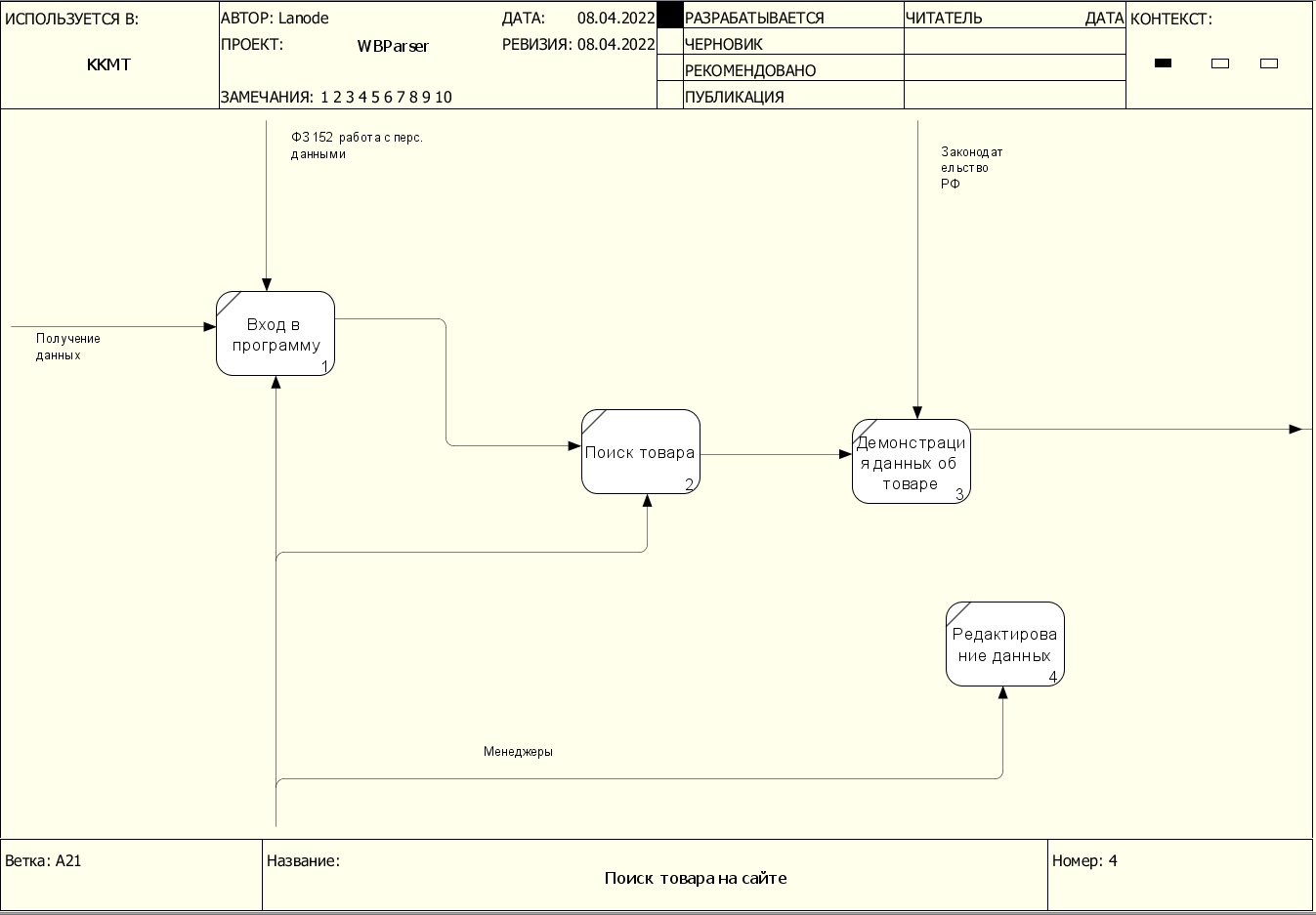


Рисунок 5. IDEF-модель 4-й уровень

* 1. **Входные и выходные данные**

Входные данные записываются администратором сайта:

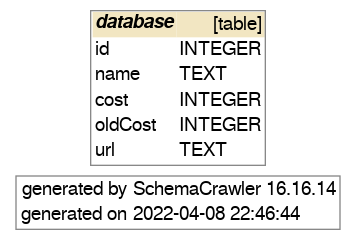
1. Айди товаров на ВайлдБеррис

Выходные данные:

1. Таблица с ценами

## Структура базы данных

Листинг 1. Таблица database



## Резервное копирование и восстановление

Чтобы сделать бэкап (экспорт) текущих моделей базы данных или всей базы данных воспользуемся командой **dumpdata**

python ./manage.py dumpdata > db.json

Если же вы хотите скачать всю информацию о БД и переместить её на другой ПК, то воспользоваться этой командой:

python ./manage.py dumpdata --exclude database --exclude contenttypes > db.json

Чтобы восстановить БД потребуется команда **loaddata** позволяет загрузить фикстуры (экспортированные с помощью dumpdata данные).

./manage.py loaddata db.json

# Разработка клиентского приложения

## Сценарий клиентского приложения

Ниже показан сценарий клиентского приложения

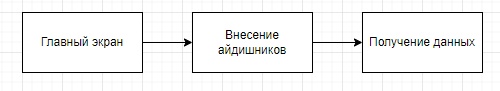


Рисунок 8. Сценарий клиентского приложения

## Разработка пользовательского интерфейса программного средства

При запуске программы открывается приветственное экран (рис. 8):

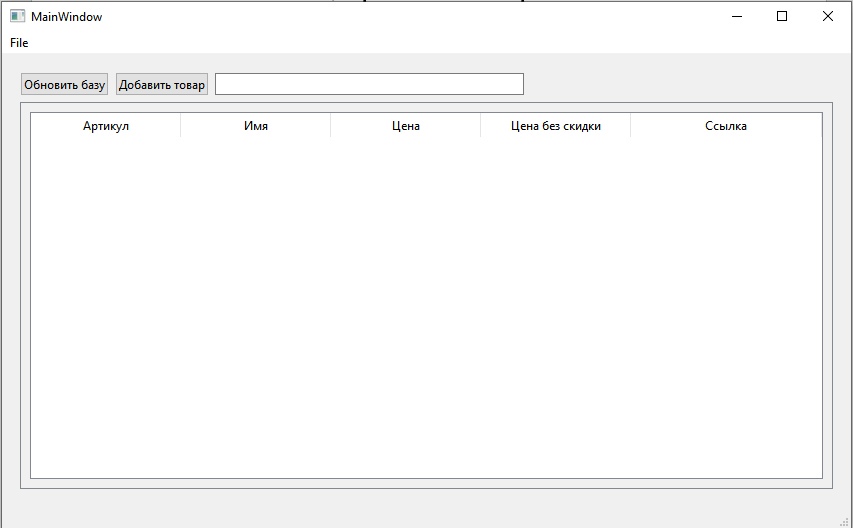


Рисунок . Приветственный экран

Отсюда, при нажатии на кнопку «Курсы», вам откроется экран с курсами (рис. 9)

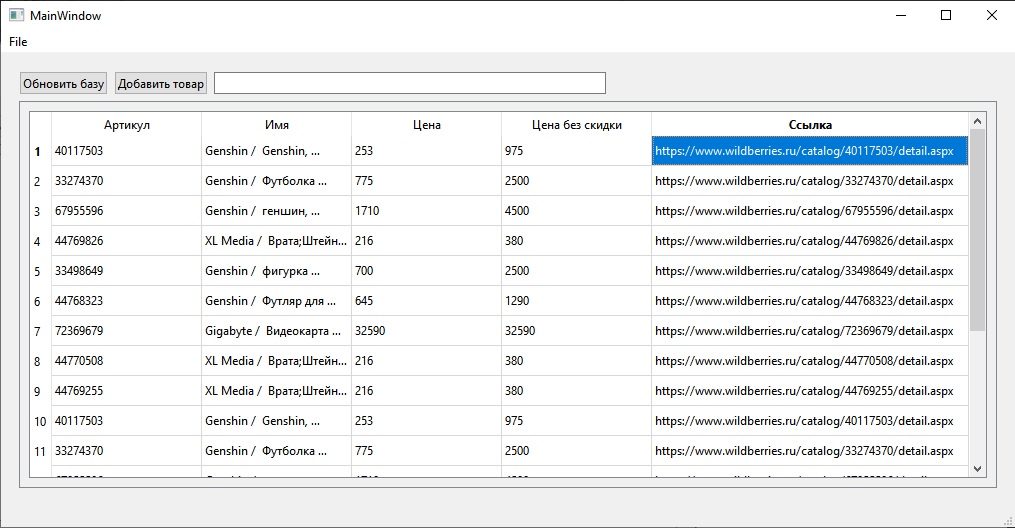


Рисунок 9. Окно с курсами

## Отладка и тестирование программного средства

Отладка и тестирование программного средства велась в PyCharm. В данной IDE можно воспользоваться консолью для вывода значений переменных в нужных точках программы (рис. 12).

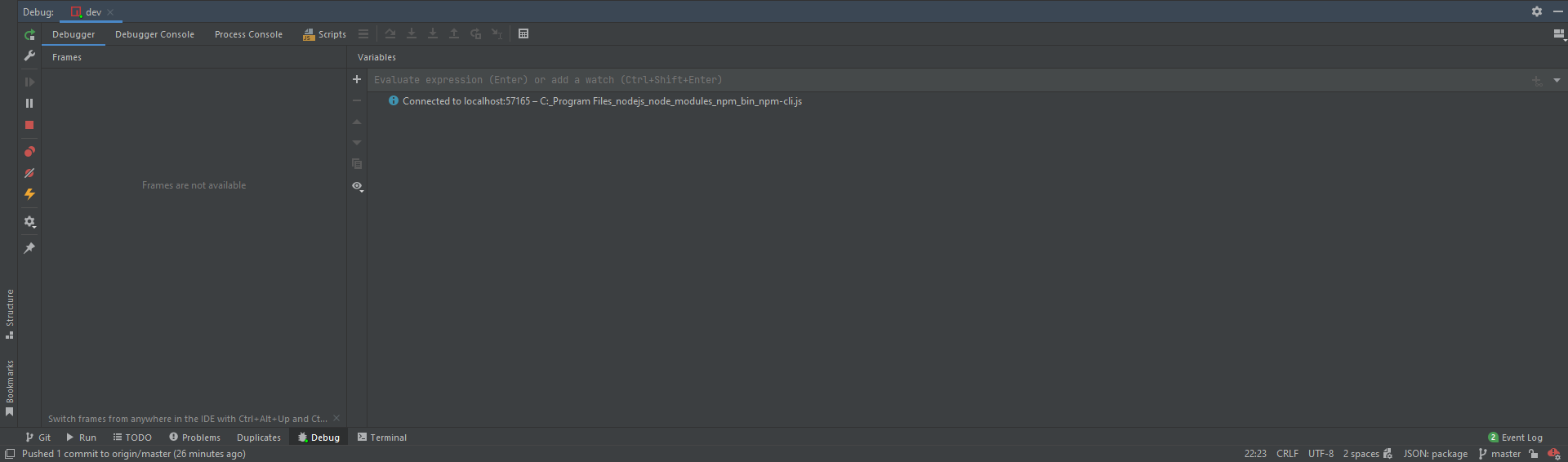


Рисунок . Консоль для отладки программы

Также присутствует система автоматизированного тестирования (рис 13).

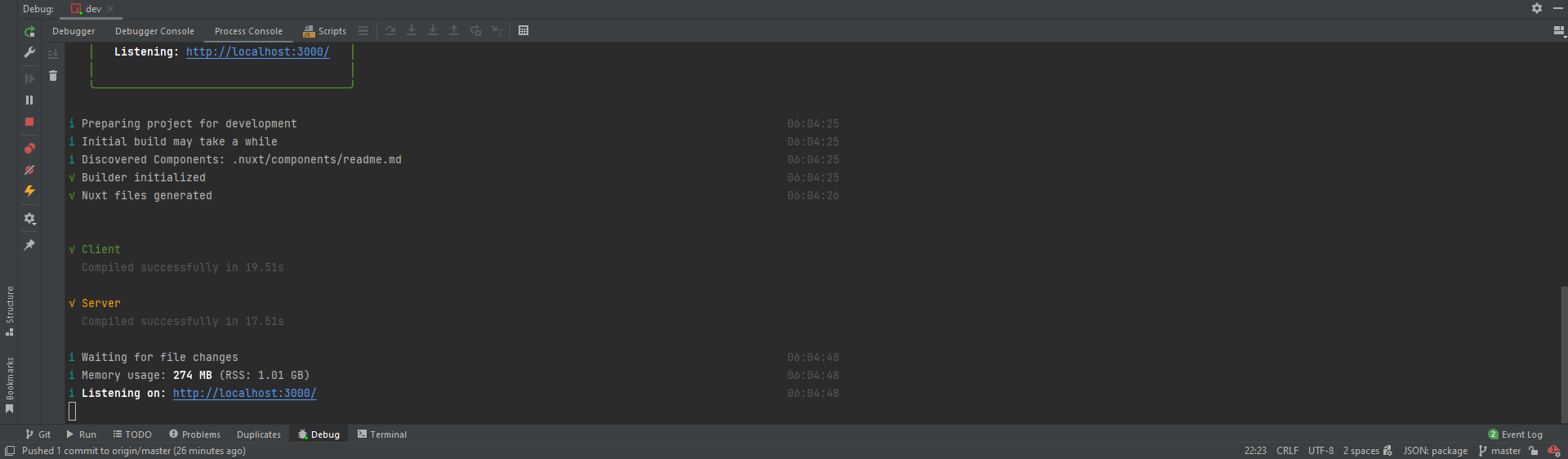


Рисунок . Окно автоматизированного тестирования

# Методы и технологии защиты информации в базах данных

## Регламента работы с ВТ

Не требуется

## Список доступа к работе с компьютерами и базой данных

Доступ к работе с компьютерами и базой данных имеется лишь у администраторов.

## Регламент разграничивающий доступ

Администратор занимается:

1. Размещением айди
2. Управлением БД

## Аппаратные средства защиты информации

В виде аппаратных средств защиты информации выступает **роутер с Firewall**

## Программные средства защиты информации

Используются следующие методы и технологии защиты информации в базе данных:

* Шифрование данных
* Организация восстановления БД
* Защита от SQL-инъекций
* Развёртывание сайта по протоколу HTTPS
* Защита от кликджекинга

## Разграничения доступа средствами базы данных

Присутствуют следующие роли:

1. Админ – непосредственно занимается администрированием БД
2. Пользователь (user)

# Заключение

В ходе производственной практики были получены практические навыки по разработке работе с БД, парсинге сайтов.

# Список литературы и интернет-ресурсов

1. SQLite

<https://www.postgresql.org/>

1. Qt6

<https://www.djangoproject.com/>

# Дневник практики

Таблица 1. Дневник практики.

|  |  |  |  |
| --- | --- | --- | --- |
| Дата | Содержание работы | Отметка о выполнении  работы | Подпись руководителя практики |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Приложение 1. Техническое задание

# Введение

## Наименование программы

Наименование программного продукта – «WBParser» (сайт).

# Краткая характеристика области

Сайт «WBParser» предназначена для парсинга страниц товаров на маркетплейсе WildBerries.

# Основания для разработки

Основанием для разработки является задание на практику.

Наименование темы разработки – «Разработка парсера для маркетплейсов».

# Назначение разработки

Мобильное приложение «WBParser» будет использоваться для парсинга и получения цен с маркетплейсов.

## Функциональное назначение

Данное ПО позволяет просмотреть цены товаров в удобном формате.

## Эксплуатационное назначение

ПО «WBParser» планируется использовать в свободном доступе любым пользователем.

# Требования к программе или программному изделию

## Требования к функциональным характеристикам

* + 1. Требования к составу выполняемых функций

После запуска программы пользователю отображается главный экран, который является ознакомительной страницей. Далее можно выбрать файлы и начать парсинг

* + 1. Требование к организации входных и выходных данных

Цены хранятся на сервере.

* + 1. Требования к временным характеристикам

Требования к временным характеристикам отсутствуют

* + 1. Требования к надёжности

Вероятность безотказной работы системы должна составлять не менее 99.99% при условии исправности версии программы.

* + 1. Требование к обеспечению надёжного функционирования программы

База данных не нуждается в резервном замещении.

Надежное функционирование программы должно быть обеспечено выполнением заказчиком совокупности организационно-технических мероприятий, перечень которых приведен ниже:

* организацией бесперебойного питания технических средств;
* Защита информации. Испытания программных средств на наличие компьютерных вирусов.
  + 1. **Время восстановления после отказа**

Время восстановления после отказа, вызванного неисправностью технических средств, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

Отказы из-за некорректных действий оператора

Отказы программы вследствие некорректных действий оператора невозможны.

# Условия эксплуатации

Программа запускается на любой системе, позволяющей открыть питон и Qt6. Окно программы должно быть открыто на весь экран, должна быть возможность свернуть приложение или запустить любое стороннее программное обеспечение.

## Климатические условия эксплуатации

Специальные условия не требуются.

# Требования к численности и квалификации

Программа не требует проведения каких-либо видов обслуживания.

## Требования к численности и квалификации персонала

Специальные условия не требуются.

# Требования к составу и параметрам технических средств

## Состав технических средств:

* + IBM PC совместимый с процессором Pentium IV и выше, для комфортной работы тактовая частота д. б. не ниже 2 гГц;
  + ОЗУ более 512 Мбайт, для комфортной работы 1 гБ;
  + 64 МБ видеопамяти и выше, для комфортной работы 256 мБ;
  + Требования к информационной и программной совместимости
  + Наличие любого браузера
  + Подключение к интернету

База данных должна составляется по примеру, который представлен в папке с программой.

## Требования к маркировке и упаковке

Требований нет

## Требования к транспортированию и хранению

Для транспортировки программы нужно интернет-соединение.

## Специальные требования

Программа должна обеспечивать взаимодействие с пользователем посредством графического пользовательского интерфейса, разработанного согласно рекомендациям компании-производителя операционной системы.

## Требования к программной документации

Предварительный состав программной документации:

* техническое задание (включает описание применения);
* программа и методика испытаний;
* руководство системного программиста;
* руководство оператора;

# Технико-экономические показатели

Программа «WBParser» пригодна для предпринимателей. Так как программа обеспечивает Быстрое получение данных с маркетплейса.

# Стадии и этапы разработки

Всего этапов разработки:

* Разработка требований
* Проектирование
* Реализация
* Тестирование
* Внедрение

На стадии «Разработка требований» должен быть выполнен этап разработки, согласования и утверждения настоящего технического задания.

На стадии «Проектирование» утверждается интерфейс программы, а также внутренние устройство программы.

На стадии «Реализации» должны быть выполнены перечисленные ниже этапы работ:

* разработка программы;
* разработка программной документации;

На стадии «Тестирование» проверяется работа функций программы и в целом корректность её работы.

На стадии «Внедрение» должен быть выполнено внедрение заказчику на компьютер.

Содержание работ по этапам:

На этапе разработки технического задания должны быть выполнены перечисленные ниже работы:

* постановка задачи;
* определение и уточнение требований к техническим средствам;
* определение требований к программе;
* определение стадий, этапов и сроков разработки программы и документации на нее;
* согласование и утверждение технического задания.

На этапе разработки программы должна быть выполнена работа по программированию (кодированию) и отладке программы.

На этапе разработки программной документации должна быть выполнена разработка программных документов в соответствии с требованиями ГОСТ 19.101-77.

На этапе испытаний программы должны быть выполнены перечисленные ниже виды работ:

* разработка, согласование и утверждение порядка и методики испытаний;
* проведение приемо-сдаточных испытаний;
* корректировка программы и программной документации по результатам испытаний.

На этапе подготовки и передачи программы должна быть выполнена работа по подготовке и передаче программы и программной документации в эксплуатацию на объектах заказчика.

# Порядок контроля и приемки

Приемосдаточные испытания программы должны проводиться согласно разработанной исполнителем и согласованной заказчиком «Программы и методики испытаний».

Ход проведения приемо-сдаточных испытаний заказчик и исполнитель документируют в протоколе испытаний. На основании протокола испытаний исполнитель совместно с заказчиком подписывают акт приемки-сдачи программы в эксплуатацию.

# Приложение 2. Руководство оператора

1. **Назначение программы**
   1. **Функциональное назначение программы**

Данный сайт даёт возможность распарсить сайт маркетплейса WildBerries и получить с него данные

* 1. **Эксплуатационное назначение программы**

ПО “WBParser” планируется использовать в общем доступе любым человеком.

* + 1. **Функция вывода курсов**

Функция парсинга выдает нам цены на товары.

1. **Условия выполнения программы**
   1. **Минимальный состав аппаратных средств**

Минимальный состав используемых технических (аппаратных) средств:

* IBM PC совместимый с процессором Pentium IV и выше, для комфортной работы тактовая частота д. б. не ниже 2 гГц;
* ОЗУ более 512 Мбайт, для комфортной работы 1 гБ;
* 64 МБ видеопамяти и выше, для комфортной работы 256 мБ;
* Наличие любого браузера
* Подключение к интернету
  1. **Требования к пользователю**

Конечный пользователь программы (оператор) должен обладать практическими навыками работы с графическим пользовательским интерфейсом. Персонал должен быть аттестован на II квалификационную группу по электробезопасности.

1. **Выполнение программы**
   1. **Загрузка и запуск сайта**

Загрузка не требуется.

* 1. **Завершение работы программы**

Для завершения работы ПО достаточно закрыть.

1. **Сообщения оператору**

Сообщения оператору отсутствуют.

Приложение 3. Руководство программиста

1. **Назначение и условия применения программы**
   1. **Функции программы**

Дынная программа призвана для парсинга маркетплейсов.

* 1. **Условия выполнения программы**

Минимальный состав используемых технических (аппаратных) средств:

* IBM PC совместимый с процессором Fentium IV и выше, для комфортной работы тактовая частота д. б. не ниже 2 гГц;
* ОЗУ более 512 Мбайт, для комфортной работы 1 гБ;
* 64 МБ видеопамяти и выше, для комфортной работы 256 мБ;
* Наличие любого браузера
* Подключение к интернету

# 

1. **Характеристики программы**

**2.1. Временные характеристики**

Временные характеристики отсутствуют.



**2.2. Режим работы**

У данного ПО режим работы по запросу.

1. **Обращение к программе**

Обращение к программе производится вводом.

Входные и выходные данные:

Входные данные: отсутствуют.

Выходные данные: цены.

1. **Сообщения**

Сообщения отсутствуют.

# Приложение 4. Листинг кода

Модуль 1. main.py

from PyQt6.QtWidgets import QApplication, QWidget, QMainWindow, QFileDialog, QMessageBox

from PyQt6.QtGui import QIcon, QAction

from PyQt6 import QtWidgets

from PyQt6.QtCore import QTimer, pyqtSignal, QThread, QObject

from bs4 import BeautifulSoup as bs

import mainwindow

import requests, re, sqlite3, sys, os, traceback

class MainWindow(QMainWindow, mainwindow.Ui\_MainWindow):

def \_\_init\_\_(self, \*args, obj=None, \*\*kwargs):

super(MainWindow, self).\_\_init\_\_(\*args, \*\*kwargs)

self.setupUi(self)

self.addIdPushButton.clicked.connect(self.addIdSlot)

self.updateDbPushButton.clicked.connect(self.updateDbSlot)

openBdAction = QAction("&Открыть базу",self)

openBdAction.setShortcut("Ctrl+O")

openBdAction.triggered.connect(self.loadDataBase)

saveBdAction = QAction("&Сохранить базу",self)

saveBdAction.setShortcut("Ctrl+S")

saveBdAction.triggered.connect(self.saveDataBase)

openListAction = QAction("&Открыть список",self)

openListAction.setShortcut("Ctrl+L")

openListAction.triggered.connect(self.openList)

fileMenu = self.menubar.addMenu('&File')

fileMenu.addAction(openBdAction)

fileMenu.addAction(saveBdAction)

fileMenu.addAction(openListAction)

class ProductItem:

id = None

name = None

cost = None

oldCost = None

url = None

def openList(self):

fileName,\_ = QFileDialog.getOpenFileName(

self,

"Выберите файл с идами",

"",

"Text Files (\*.txt);;All Files (\*)")

if (not fileName): return

try:

ids = open(fileName,'r').read().split('\n')

for id in ids:

rowPosition = self.tableWidget.rowCount()

self.tableWidget.insertRow(rowPosition)

self.editItem(self.parseProduct(id), rowPosition)

except Exception as E:

msg = QMessageBox()

msg.setText(traceback.format\_exc())

msg.setWindowTitle("Ошибка во время парсинга листа идов")

msg.exec()

print(traceback.format\_exc())

def saveDataBase(self):

fileName,\_ = QFileDialog.getSaveFileName(

self,

"Сохранить базу данных",

"",

"Sqlite Files (\*.db)")

if (not fileName): return

if os.path.exists(fileName): os.remove(fileName)

db = sqlite3.connect(fileName)

db\_cur = db.cursor()

db\_cur.execute('CREATE TABLE IF NOT EXISTS database (id INTEGER,name TEXT,cost INTEGER,oldCost INTEGER,url TEXT)')

rowCount = self.tableWidget.rowCount()

for row in range(rowCount):

productItem = self.ProductItem()

productItem.id = self.tableWidget.item(row,0).text()

productItem.name = self.tableWidget.item(row,1).text()

productItem.cost = self.tableWidget.item(row,2).text()

productItem.oldCost = self.tableWidget.item(row,3).text()

productItem.url = self.tableWidget.item(row,4).text()

db\_cur.execute(

'INSERT INTO database VALUES (?,?,?,?,?)',

[

productItem.id,

productItem.name,

productItem.cost,

productItem.oldCost,

productItem.url

]

)

db.commit()

db\_cur.close()

db.close()

def loadDataBase(self):

fileName,\_ = QFileDialog.getOpenFileName(

self,

"Выберите базу данных",

"",

"Sqlite Files (\*.db);;All Files (\*)")

if (not fileName): return

db = sqlite3.connect(fileName)

db\_cur = db.cursor()

db\_cur.execute('SELECT \* FROM database')

result = db\_cur.fetchall()

db\_cur.close()

db.close()

while self.tableWidget.rowCount() > 0:

self.tableWidget.removeRow(0)

for row in result:

productItem = self.ProductItem()

productItem.id = str(row[0])

productItem.name = row[1]

productItem.cost = str(row[2])

productItem.oldCost = str(row[3])

productItem.url = row[4]

rowPosition = self.tableWidget.rowCount()

self.tableWidget.insertRow(rowPosition)

self.editItem(productItem, rowPosition)

def addIdSlot(self):

id = self.lineEdit.text()

if (not id): return

rowPosition = self.tableWidget.rowCount()

self.tableWidget.insertRow(rowPosition)

self.editItem(self.parseProduct(id), rowPosition)

self.lineEdit.clear()

def parseProduct(self, id):

productItem = self.ProductItem()

productItem.url = 'https://www.wildberries.ru/catalog/{}/detail.aspx'.format(id)

index = bs(requests.get(productItem.url).text,'html.parser')

productItem.id = id

productItem.name = index.find(class\_="same-part-kt\_\_header").text

productItem.cost = ''.join(re.findall("\d+",index.find(class\_="price-block\_\_final-price").text))

productItem.oldCost = index.find(class\_="price-block\_\_old-price")

if productItem.oldCost != None: productItem.oldCost = ''.join(re.findall("\d+",productItem.oldCost.text))

else: productItem.oldCost = productItem.cost

return productItem

def editItem(self, productItem, rowPosition):

self.tableWidget.setItem(rowPosition, 0, QtWidgets.QTableWidgetItem(productItem.id))

self.tableWidget.setItem(rowPosition, 1, QtWidgets.QTableWidgetItem(productItem.name))

self.tableWidget.setItem(rowPosition, 2, QtWidgets.QTableWidgetItem(productItem.cost))

self.tableWidget.setItem(rowPosition, 3, QtWidgets.QTableWidgetItem(productItem.oldCost))

self.tableWidget.setItem(rowPosition, 4, QtWidgets.QTableWidgetItem('https://www.wildberries.ru/catalog/{}/detail.aspx'.format(productItem.id)))

def updateDbSlot(self):

rowCount = self.tableWidget.rowCount()

for row in range(rowCount):

id = self.tableWidget.item(row,0).text()

self.editItem(self.parseProduct(id), row)

def keyReleaseEvent(self, e):

if e.key() == 16777223:

indexes = self.tableWidget.selectionModel().selectedRows()

for index in sorted(indexes):

self.tableWidget.removeRow(index.row())

break

app = QApplication(sys.argv)

window = MainWindow()

window.show()

app.exec()

Модуль 2. mainwindow.py

# Form implementation generated from reading ui file '.\mainwindow.ui'

#

# Created by: PyQt6 UI code generator 6.2.3

#

# WARNING: Any manual changes made to this file will be lost when pyuic6 is

# run again. Do not edit this file unless you know what you are doing.

from PyQt6 import QtCore, QtGui, QtWidgets

class Ui\_MainWindow(object):

def setupUi(self, MainWindow):

MainWindow.setObjectName("MainWindow")

MainWindow.resize(849, 496)

self.centralwidget = QtWidgets.QWidget(MainWindow)

self.centralwidget.setObjectName("centralwidget")

self.verticalLayout = QtWidgets.QVBoxLayout(self.centralwidget)

self.verticalLayout.setObjectName("verticalLayout")

self.stackedWidget = QtWidgets.QStackedWidget(self.centralwidget)

self.stackedWidget.setObjectName("stackedWidget")

self.page = QtWidgets.QWidget()

self.page.setObjectName("page")

self.verticalLayout\_2 = QtWidgets.QVBoxLayout(self.page)

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

self.horizontalLayout\_2 = QtWidgets.QHBoxLayout()

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

self.updateDbPushButton = QtWidgets.QPushButton(self.page)

self.updateDbPushButton.setObjectName("updateDbPushButton")

self.horizontalLayout\_2.addWidget(self.updateDbPushButton)

self.addIdPushButton = QtWidgets.QPushButton(self.page)

self.addIdPushButton.setObjectName("addIdPushButton")

self.horizontalLayout\_2.addWidget(self.addIdPushButton)

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

self.lineEdit.setObjectName("lineEdit")

self.horizontalLayout\_2.addWidget(self.lineEdit)

spacerItem = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Policy.Expanding, QtWidgets.QSizePolicy.Policy.Minimum)

self.horizontalLayout\_2.addItem(spacerItem)

self.verticalLayout\_2.addLayout(self.horizontalLayout\_2)

self.scrollArea = QtWidgets.QScrollArea(self.page)

self.scrollArea.setWidgetResizable(True)

self.scrollArea.setObjectName("scrollArea")

self.scrollAreaWidgetContents = QtWidgets.QWidget()

self.scrollAreaWidgetContents.setGeometry(QtCore.QRect(0, 0, 803, 362))

self.scrollAreaWidgetContents.setObjectName("scrollAreaWidgetContents")

self.verticalLayout\_3 = QtWidgets.QVBoxLayout(self.scrollAreaWidgetContents)

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

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

self.tableWidget.setObjectName("tableWidget")

self.tableWidget.setColumnCount(5)

self.tableWidget.setRowCount(0)

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)

item = QtWidgets.QTableWidgetItem()

self.tableWidget.setHorizontalHeaderItem(4, item)

self.tableWidget.horizontalHeader().setCascadingSectionResizes(False)

self.tableWidget.horizontalHeader().setDefaultSectionSize(150)

self.tableWidget.horizontalHeader().setSortIndicatorShown(True)

self.tableWidget.horizontalHeader().setStretchLastSection(True)

self.tableWidget.verticalHeader().setStretchLastSection(False)

self.verticalLayout\_3.addWidget(self.tableWidget)

self.scrollArea.setWidget(self.scrollAreaWidgetContents)

self.verticalLayout\_2.addWidget(self.scrollArea)

self.stackedWidget.addWidget(self.page)

self.page\_2 = QtWidgets.QWidget()

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

self.stackedWidget.addWidget(self.page\_2)

self.verticalLayout.addWidget(self.stackedWidget)

MainWindow.setCentralWidget(self.centralwidget)

self.menubar = QtWidgets.QMenuBar(MainWindow)

self.menubar.setGeometry(QtCore.QRect(0, 0, 849, 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)

def retranslateUi(self, MainWindow):

\_translate = QtCore.QCoreApplication.translate

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

self.updateDbPushButton.setText(\_translate("MainWindow", "Обновить базу"))

self.addIdPushButton.setText(\_translate("MainWindow", "Добавить товар"))

item = self.tableWidget.horizontalHeaderItem(0)

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

item = self.tableWidget.horizontalHeaderItem(1)

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

item = self.tableWidget.horizontalHeaderItem(2)

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

item = self.tableWidget.horizontalHeaderItem(3)

item.setText(\_translate("MainWindow", "Цена без скидки"))

item = self.tableWidget.horizontalHeaderItem(4)

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

Модуль 3. mainwindow.ui

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>MainWindow</class>

<widget class="QMainWindow" name="MainWindow">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>849</width>

<height>496</height>

</rect>

</property>

<property name="windowTitle">

<string>MainWindow</string>

</property>

<widget class="QWidget" name="centralwidget">

<layout class="QVBoxLayout" name="verticalLayout">

<item>

<widget class="QStackedWidget" name="stackedWidget">

<widget class="QWidget" name="page">

<layout class="QVBoxLayout" name="verticalLayout\_2">

<item>

<layout class="QHBoxLayout" name="horizontalLayout\_2">

<item>

<widget class="QPushButton" name="updateDbPushButton">

<property name="text">

<string>Обновить базу</string>

</property>

</widget>

</item>

<item>

<widget class="QPushButton" name="addIdPushButton">

<property name="text">

<string>Добавить товар</string>

</property>

</widget>

</item>

<item>

<widget class="QLineEdit" name="lineEdit"/>

</item>

<item>

<spacer name="horizontalSpacer">

<property name="orientation">

<enum>Qt::Horizontal</enum>

</property>

<property name="sizeHint" stdset="0">

<size>

<width>40</width>

<height>20</height>

</size>

</property>

</spacer>

</item>

</layout>

</item>

<item>

<widget class="QScrollArea" name="scrollArea">

<property name="widgetResizable">

<bool>true</bool>

</property>

<widget class="QWidget" name="scrollAreaWidgetContents">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>803</width>

<height>362</height>

</rect>

</property>

<layout class="QVBoxLayout" name="verticalLayout\_3">

<item>

<widget class="QTableWidget" name="tableWidget">

<attribute name="horizontalHeaderCascadingSectionResizes">

<bool>false</bool>

</attribute>

<attribute name="horizontalHeaderDefaultSectionSize">

<number>150</number>

</attribute>

<attribute name="horizontalHeaderShowSortIndicator" stdset="0">

<bool>true</bool>

</attribute>

<attribute name="horizontalHeaderStretchLastSection">

<bool>true</bool>

</attribute>

<attribute name="verticalHeaderStretchLastSection">

<bool>false</bool>

</attribute>

<column>

<property name="text">

<string>Артикул</string>

</property>

</column>

<column>

<property name="text">

<string>Имя</string>

</property>

</column>

<column>

<property name="text">

<string>Цена</string>

</property>

</column>

<column>

<property name="text">

<string>Цена без скидки</string>

</property>

</column>

<column>

<property name="text">

<string>Ссылка</string>

</property>

</column>

</widget>

</item>

</layout>

</widget>

</widget>

</item>

</layout>

</widget>

<widget class="QWidget" name="page\_2"/>

</widget>

</item>

</layout>

</widget>

<widget class="QMenuBar" name="menubar">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>849</width>

<height>26</height>

</rect>

</property>

</widget>

<widget class="QStatusBar" name="statusbar"/>

</widget>

<resources/>

<connections/>

</ui>