Министерство науки и высшего образования Российской федерации

Федерального государственного бюджетного образовательного учреждения высшего образования

Московский государственный технический университет имени Н. Э. Баумана

(Национальный исследовательский университет)

Московский техникум космического приборостроения

**КУРСОВОЙ ПРОЕКТ**

по теме: «Разработка ИС «Фотоцентр»»

Специальность: 09.02.07 Информационные системы и программирование – квалификация «Программист»

Дисциплина: МДК.04.01 Внедрение и поддержка информационных систем

Группа ТИП–51

Проверил Осипов В.Н.

Выполнили Низамутдинова А.Р.

Пугачева А.Е

Сонина А.С.

Москва 2022

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет им. Н.Э. Баумана»

(национальный исследовательский университет)

**Московский техникум космического приборостроения**

УТВЕРЖДАЮ

Председатель ПЦК

Е. А. Митрошенкова

« » 2022г.

ЗАДАНИЕ

на выполнение курсового проекта

По дисциплине МДК.04.01 «Информационные системы и программирование»)

Студент Низамутдинова. А. Р. гр. ТИП-51 09.02.07

Студент Пугачева А.Е. гр. ТИП-51 09.02.07

Студент Сонина А.С. гр. ТИП-51 09.02.07

(фамилия, инициалы, индекс группы)

Руководитель В.Н. Осипов

(фамилия, инициалы)

1. ***Тема курсового проекта***

*Разработка ИС «Фотоцентр»*

1. ***Техническое задание***

*Разработать ИС для ведения учетов заказов фотоцентра*

1. ***Оформление курсового проекта***
   1. Пояснительная записка на листах формата А4.
   2. Перечень графического материала КП (схемы, чертежи и т.п.) – схемы алгоритма программы

График выполнения работы: 05.09.2022 - 10.12.2022.

Дата выдачи задания «05» сентября 2022 г.

Руководитель курсового проекта В. Н. Осипов

Перечень сокращений

В данном разделе представлен перечень сокращений, используемых в настоящем документе (см. Таблица 1).

Таблица 1 – Перечень сокращений

| **Сокращение** | **Расшифровка** |
| --- | --- |
| ИС | Информационная система |
| БД | База данных |
| ПО | Программное обеспечение |
| СУБД | Система управления баз данных |

Термины и определения

В данном разделе представлен перечень терминов и определений, используемых в настоящем документе (см. Талица 2).

Таблица 2 – Перечень терминов

| **Термин** | **Определение** |
| --- | --- |
| DFD | Data flow diagrams - диаграммы потоков данных. Так называется методология графического структурного анализа, описывающая внешние по отношению к системе, источники и адресаты данных, логические функции, потоки данных и хранилища данных, к которым осуществляется доступ. |
| IDEF0 | Методология функционального моделирования и графическая нотация, предназначенная для формализации и описания бизнес-процессов. |

СОДЕРЖАНИЕ

[Введение 6](#постан)

Глава 1. Теоретическая часть 7

Глава 2. Практическая часть 10

Создание БД и анализ системы 10

Отладка программы 13

Тестирование 16

Заключение 20

Список использованных источников 21

Приложение 1. Устав 22

Приложение 2. Техническое задание 23

Приложение 3. Руководство по установке 24

Приложение 4. Руководство пользователя 35

Приложение 5. Руководство администратора 36

Приложение 6. Программный код 42

Введение

Современный научно-технический прогресс немыслим без широкого внедрения и использования вычислительной техники в производстве, управлении и научных исследованиях.

Основы современной информационной технологии составляют базы данных (БД) и системы управления базами данных (СУБД), роль которых как единого средства хранения, обработки и доступа к большим объемам информации постоянно возрастает. При этом существенным является постоянное повышение объемов информации, хранимой в БД, что влечет за собой требование увеличения производительности таких систем.

В данной курсовой работе объектом исследования является фотоцентр. Предметом исследования является структура фотоцентра, его состав.

Фотоцентр занимается печатью, фотосьемкой, а также ретушью фотографий. Имеет небольшой штат сотрудников и хорошие амбиции, поэтому уже на ранних этапах директор понял, что нужно автоматизировать систему учета заказов и клиентов.

Целью курсовой работы является разработка базы данных, обеспечивающую работу с информацией, касающуюся структуры фотоцентра и приложение, благодаря которому будет удобно и понятно осуществлять работу.

Задачами курсовой работы является систематизация, закрепление и расширение теоретических и практических знаний по созданию БД а также развитие навыков по разработке приложений. Также задачей курсовой работы является создание качественной документации с применением новых знаний, однозначными доступными инструкциями и целостными комментариями.

Глава 1. Теоретическая часть.

* 1. Выбор средств программирования

Python стал одним из самых популярных языков, он используется в анализе данных, машинном обучении, DevOps и веб-разработке, а также в других сферах, включая разработку игр. За счёт читабельности, простого синтаксиса и отсутствия необходимости в компиляции язык хорошо подходит для обучения программированию, позволяя концентрироваться на изучении алгоритмов, концептов и парадигм. Одни из его больших плюсов для нас это:

* Интерпретируемость;
* Обширная сфера использования;
* Востребованность;
* Большое количество библиотек;
* Кроссплатформенность.

Поэтому наш выбор пал на него.

Для разработки БД мы выбрали SQLite, это быстрая и легкая встраиваемая однофайловая СУБД на языке C, которая не имеет сервера и позволяет хранить всю базу локально на одном устройстве. Для работы SQLite не нужны сторонние библиотеки или службы. СУБД использует множество программ: Firefox, Chrome, Safari, Skype, XnView, AIMP, Dropbox, Viber и другие. Основные достоинства выбранной СУБД:

* доступность;
* SQL-совместимость;
* отсутствие потребности в администрировании;
* простая процедура подключения;
* высокая производительность, при достаточном объеме ОЗУ;
* возможность шифрования данных;

На данных этапах развития ИС фотоцентра, нам достаточно функционала SQLite и Python, в дальнейшем, при развитии проекта, его масштабировании, эти программы могут не справляться с требованиями и придется улучшить её другими средствами.

* 1. Анализ требований и технического оснащения команды

При разработке проекта выдавались требования к проекту.

Техническое задание и устав проекта можно посмотреть в приложении 1 и приложении 2.

Требования к проекту: современное приложение с понятным интерфейсом, с корректной работой БД.

Цель к проекту: получить максимальное удобство для пользователей.

Мы приступили к проработке технического оснащения команды, представлено в таблице 1.1.

Таблица 1.1 техническое оснащение команда.

|  |  |
| --- | --- |
| Должность\ отдел | Технические характеристики |
| Отдел документации | Отдел документации оснащен компьютерами на базе процессора Intel core i5-7500k, графика на базе MSI GeForce GTX 1050 Ti OC, также оперативной памяти 16 г. И SSD диск на 1 гб. |
| Отдел разработки | Отдел разработки оснащен компьютерами на базе процессора Intel core i5-7500k, графика на базе MSI GeForce GTX 1050 Ti OC, также оперативной памяти 16 г. И SSD диск на 1 гб. |

На разработку мы выделили 200 руб. и назначили сроки выполнения работ. Сроки представлены в таблице 1.2.

Таблица 1.2 Сроки выполнения работ по разработке ИС.

|  |  |  |
| --- | --- | --- |
| 01.09.2022 | | Начало разработки приложения (Низамутдинова Анастасия) |
| 05.09.2022 | Согласование интерфейса приложения (Низамутдинова Анастасия, Пугачева Алиса и Сонина Анастасия) | |
| 15.09.2022 | Написание основного кода (Низамутдинова Анастасия) исправление ошибок (Сонина Анастасия) | |
| 25.09.2022 | Проверка функциональных возможностей приложения (Низамутдинова Анастасия, Пугачева Алиса и Сонина Анастасия) | |
| 20.09.2022-30.11.2022 | Исправление ошибок (улучшенное перемещение между окнами приложения) (Сонина Анастасия) | |
| 01.10.2022-01.12.2022 | Написание документации (Пугачева Алиса) | |
| 13.12.2022 | Сдача проекта с документацией (Низамутдинова Анастасия, Пугачева Алиса и Сонина Анастасия) | |

Глава 2. Практическая часть.

2.1 Создание БД и анализ системы

Для корректной работы приложения нужно, чтобы БД была логически правильно разработана и все сущности имели верные атрибуты. Также нужно создать диаграмму IDEF0. Она предназначенная для формализации и описания бизнес-процессов. Отличительной особенностью IDEF0 является её акцент на соподчинённость объектов. В IDEF0 рассматриваются логические отношения между работами, а не их временная последовательность. Диаграмма и её декомпозиция изображена ниже на рисунках 2.1-2.1.

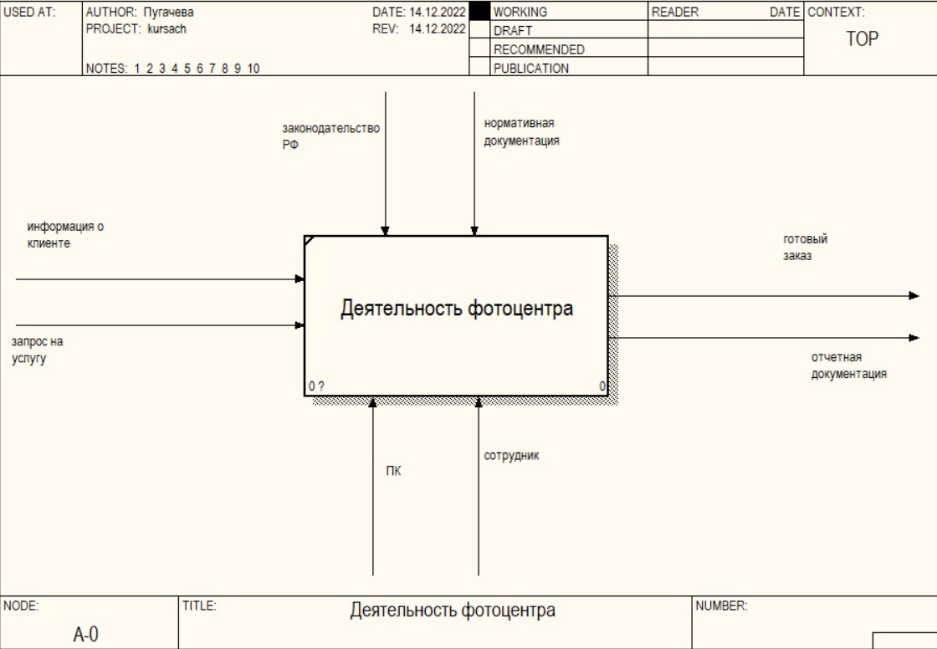


Рисунок 2.1 – Диаграмма IDEF0 первого уровня

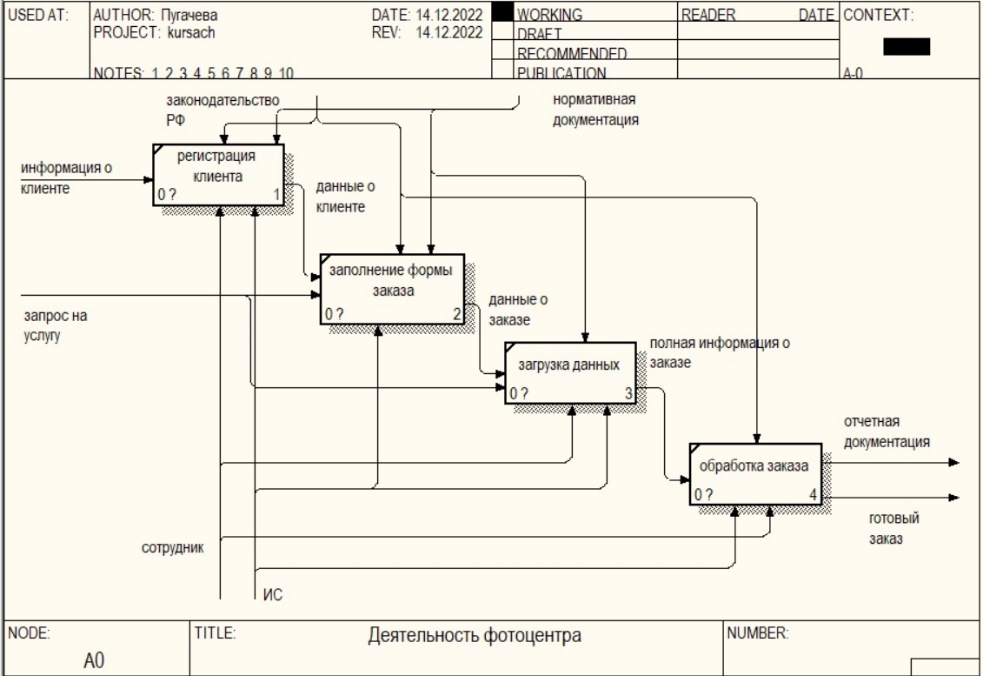


Рисунок 2.1 – Декомпозиция диаграммы IDEF0

DFD (data flow diagram) — диаграмма потоков данных, один из основных инструментов структурного анализа и проектирования информационных систем, существовавших еще до широкого распространения UML. В своей работе мы тоже решили использовать эту диаграмму для более ясного представления работы нашей ИС. Диаграмма изображена на рисунке 2.3.

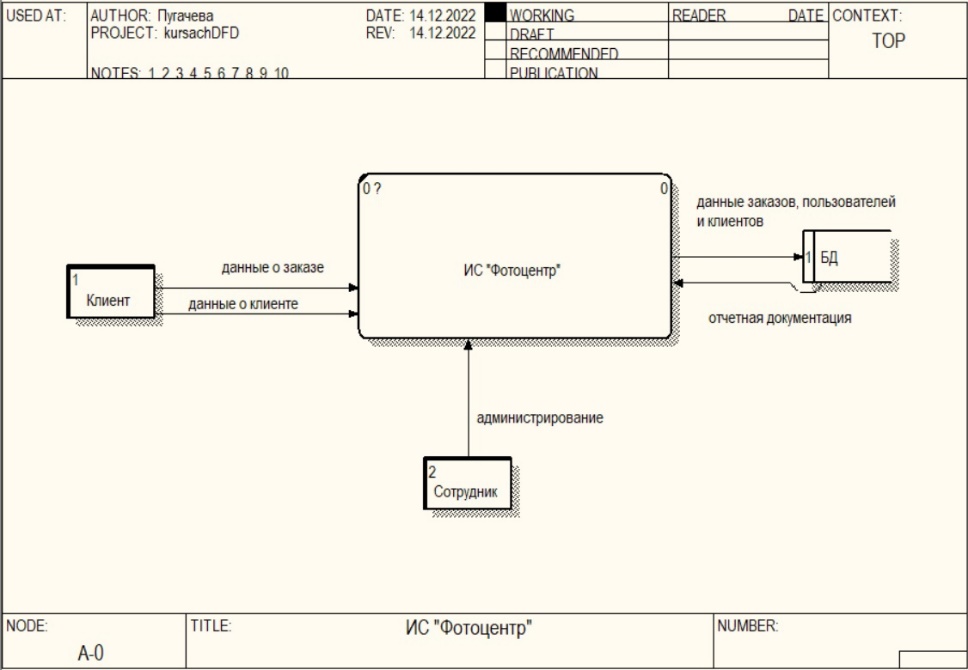


Рисунок 2.3 – Диаграмма потоков данных

Для наглядности построения БД мы использовали Oracle SQL Developer Data Modeler – это комплексное ПО, позволяющее разработчикам проектировать реляционные модели взаимосвязей объектов для последующего преобразования их в полноценные БД. Продукт поддерживает логическое, реляционное, многомерное моделирование и моделирование типов данных, предлагая возможности многоуровневого проектирования и построения концептуальных диаграмм сущностей и связей. Спроектированные модели показаны ниже на рисунках 2.4-2.7.

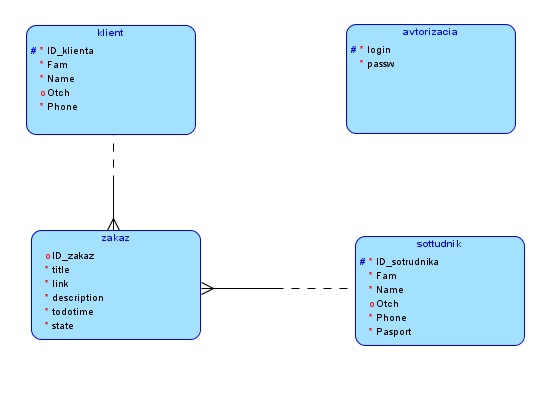


Рисунок 2.4 – Нотация Баркера

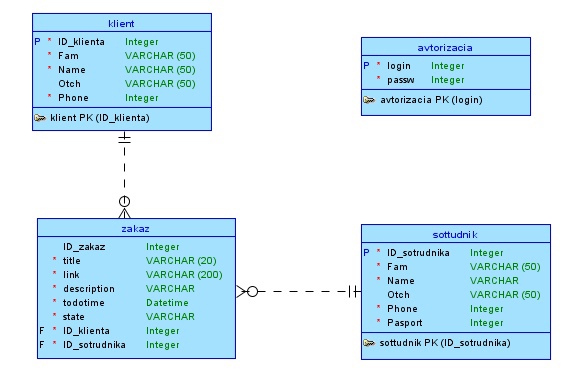


Рисунок 2.5 – Информационная модель БД

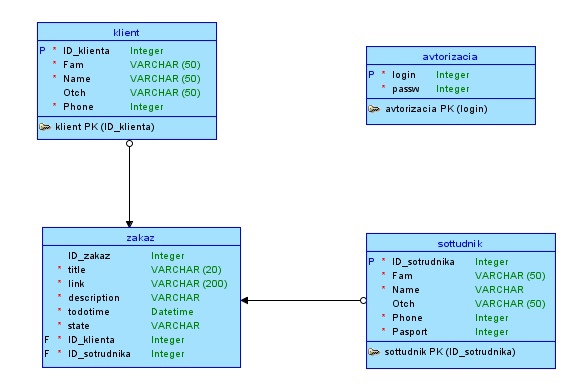


Рисунок 2.6 – Логическая модель БД

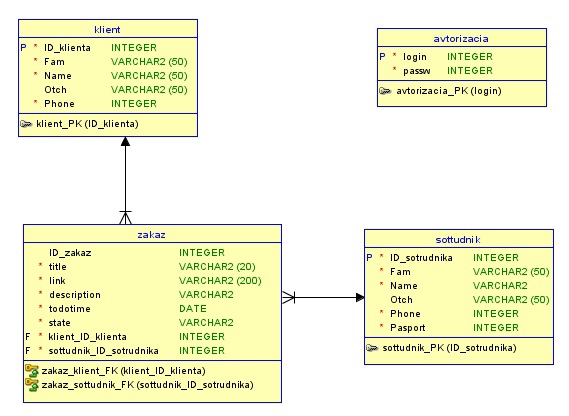


Рисунок 2.7 – Реляционная модель БД

2.2 Отладка программы

Отладка №1

Не найден файл с ui. Ошибка показана на рисунке 2.1.

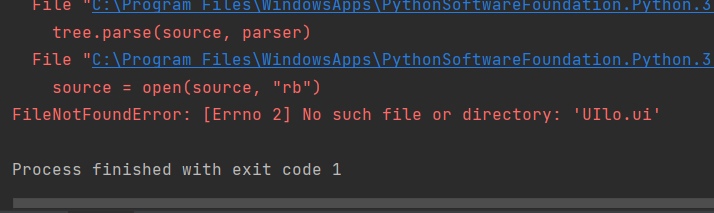


Рисунок 2.1 – Ошибка №1

Код с ошибкой показан ниже:

class WindowStart(QMainWindow):

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

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

uic.loadUi('UIlo.ui', self)

self.con = sqlite3.connect('todo.db')

self.cur = self.con.cursor()

Исправленная часть кода: uic.loadUi('UIlog.ui', self)

Ниже показан успешный запуск после исправления, рисунок 2.2.

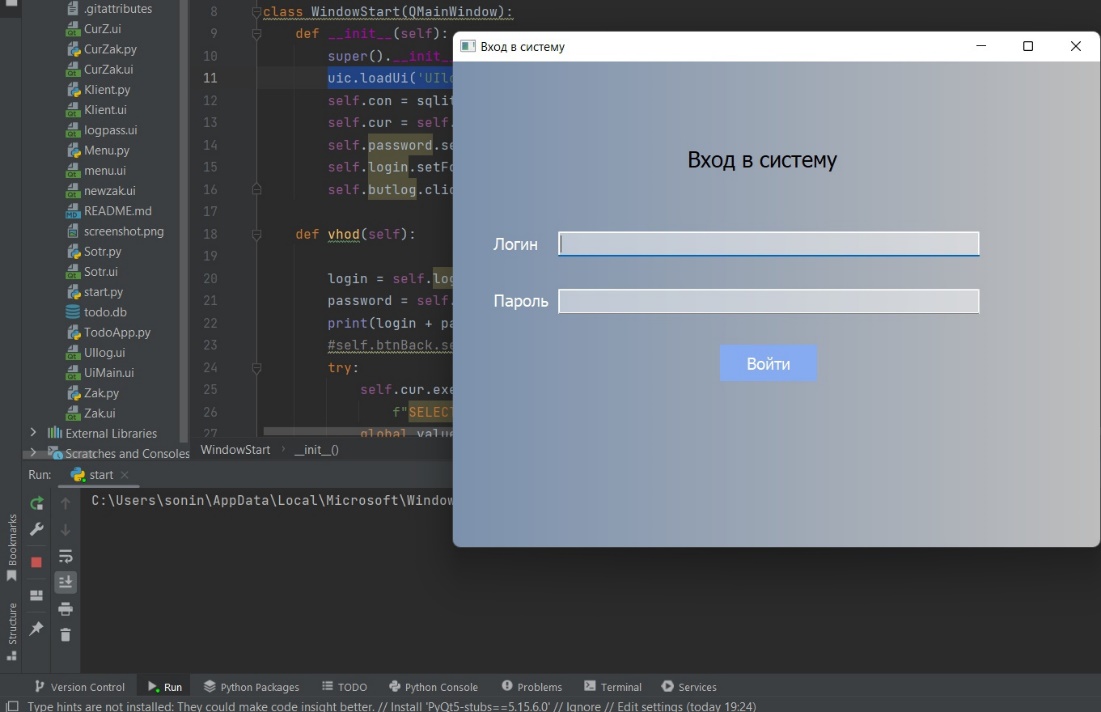


Рисунок 2.2 – Успешный запуск №1

Отладка №2

При входе с данными админа окно авторизации закрывается и вылетает сообщение об ошибке. должно открыться окно меню, но этого не происходит, и программа преждевременно завершается. Ошибка показана на рисунке 2.3.

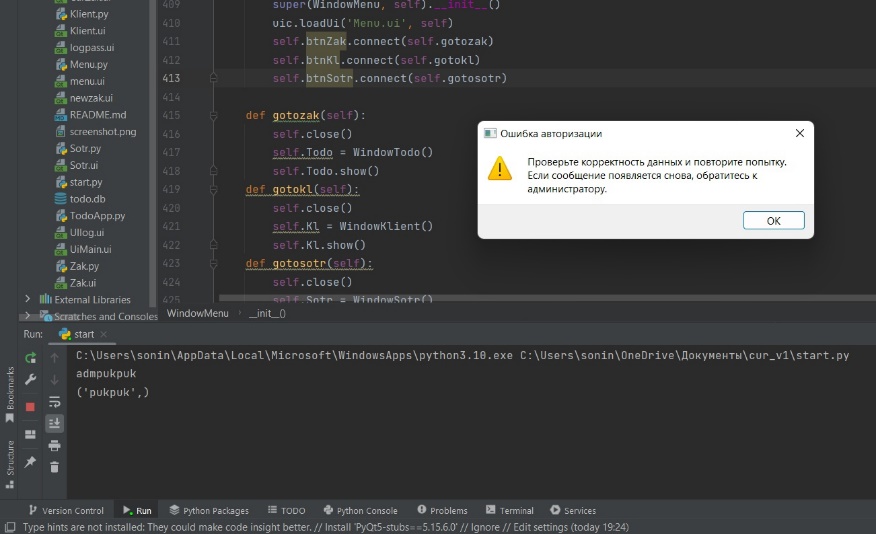


Рисунок 2.3 – Ошибка №2

Код с ошибкой показан ниже:

class WindowMenu(QDialog):  
def \_\_init\_\_(self):  
super(WindowMenu, self).\_\_init\_\_()  
uic.loadUi('Menu.ui', self)  
self.btnZak.connect(self.gotozak)  
self.btnKl.connect(self.gotokl)  
self.btnSotr.connect(self.gotosotr)

Ошибка - отстствует вызов свойства кнопки clicked.

Исправленный код:

self.btnZak.clicked.connect(self.gotozak)  
self.btnKl.clicked.connect(self.gotokl)  
self.btnSotr.clicked.connect(self.gotosotr)

Ниже показан успешный запуск после исправления, рисунок 2.4.

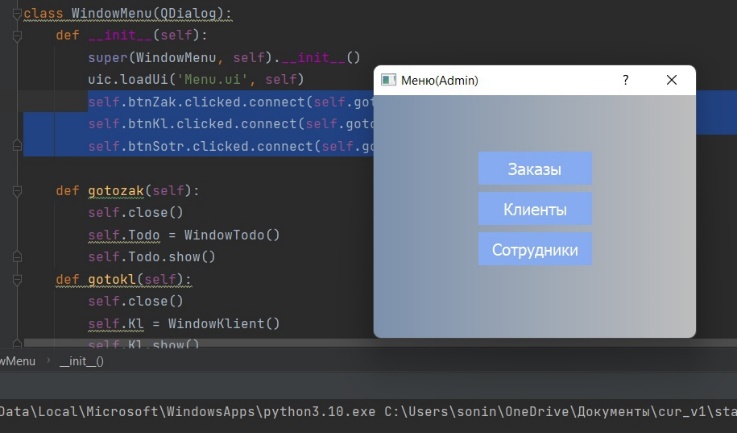


Рисунок 2.4 – Успешный запуск №2

Отладка №3

При попытке перехода в окно с заказами программа преждевременно завершается с ошибкой, это показано на рисунке 2.5.

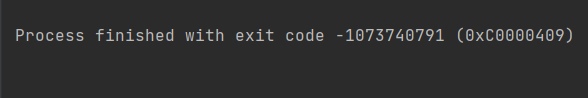


Рисунок 2.5 – Ошибка №3

Код с ошибкой показан ниже:

def FillTable(self):  
sort = self.stateslist.currentText()  
datesort = self.datesort.currentText()  
if sort == 'Все':  
# self.cur.execute("SELECT `ID\_zak`, `title`, `todotime`, `state` FROM zak ORDER BY `todotime` DESC")  
if datesort == 'Сначала новые':  
self.cur.execute("SELECT `ID\_zak`, `title`, `todotime`, `state` FROM zak ORDER BY `todotime` DECS")  
else:  
self.cur.execute("SELECT `ID\_zak`, `title`, `todotime`, `state` FROM zak ORDER BY `todotime`")

Решение - при загрузке окна с заказами выполняется процедура по заполнению таблицы данными из БД.

В sql-запросе ошибка в флажке DECS(последние две буквы перепутаны местами). из-за невозможности выполнить запрос выборку для заполнения таблицы возникает ошибка и окно не открывается.

Исправленый код: self.cur.execute("SELECT `ID\_zak`, `title`, `todotime`, `state` FROM zak ORDER BY `todotime` DESC ")

После исправления ошибки в запросе окно успешно открывается, таблица заполнена верными актуальными данными, показано на рисунке 2.6.

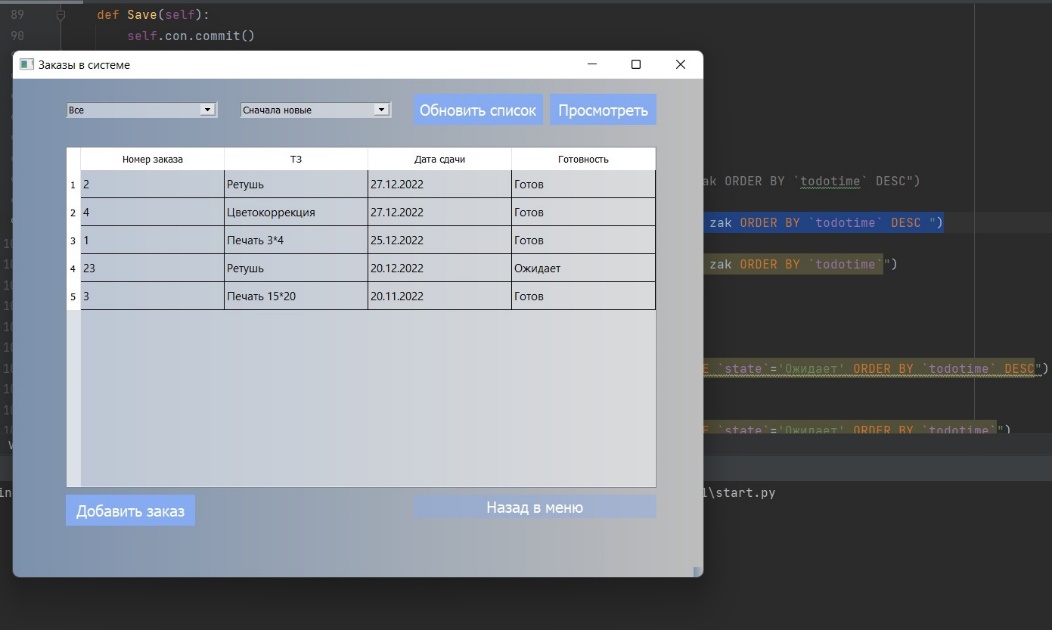


Рисунок 2.6 – Успешный запуск №3

2.3 Тестирование

Для тестирования программы мы использовали метод создания тест-кейсов, они представлены в таблицах 2.1-2.3.

Таблица 2.1 – Описание тест плана

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Результат теста |
| 1. Вход в систему   – Логин  – Пароль | – Окно открыто  – Поля заполнены  – Кнопка «Войти» активна  – Поля корректно заполнены:   * Успешный вход в систему   – Поля заполнены некорректно:   * Сообщение о проверке введенных данных |  |
| 1. Просмотр заказов в системе | – Открыто окно с базой данных  – Кнопка «Обновить список» активна  – Кнопка «Просмотреть» активна  – Данные в таблице обновляются |  |
| 1. Добавление сотрудника администратором:   – Имя  – Фамилия  – Отчество  – Дата рождения  – Телефон  – Паспорт | – Окно открыто  – Поля не заполнены  – Кнопка «Создать» не активна  – Кнопка «Удалить» не активна  – Кнопка «Сохранить» не активна |  |
| 1. Заполнение полей администратором для добавления сотрудника:   – Имя  – Фамилия  – Отчество  – Дата рождения  – Телефон  – Паспорт | – Окно открыто  – Поля заполнены  – Кнопка «Создать» активна  – Кнопка «Удалить» активна  – Кнопка «Сохранить» активна |  |
| 1. Нажимаем кнопку «Создать» | – Если введены корректные данные:   * Информация поступает в базу данных, отображение в системе после обновления   – Данные введены некорректно:   * Сообщение о проверке введенных данных |  |
| 1. Заполнение информации о клиенте:   – Имя  – Фамилия  – Отчество  – Телефон | – Окно открыто  – Поля не заполнены  – Кнопка «Создать» не активна  – Кнопка «Удалить» не активна  – Кнопка «Сохранить» не активна |  |
| 1. Заполнение информации о клиенте:   – Имя  – Фамилия  – Отчество  – Телефон | – Окно открыто  – Поля заполнены  – Кнопка «Создать» активна  – Кнопка «Удалить» активна  – Кнопка «Сохранить» активна |  |

Таблица 2.2 – Позитивный тест кейс

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Результат теста |
| 1. Вход в систему   – adm  – \*\*\* | – Окно открыто  – Поля заполнены  – Кнопка «Войти» активна  – Успешный вход в систему | Пройден |
| 1. Заполнение полей администратором для добавления сотрудника:   – Имя: Валерий  – Фамилия: Сидоров  – Отчество: Александрович  – Дата рождения: 26.05.2001  – Телефон: +7456738229  – Паспорт:12345678910 | – Окно открыто  – Поля заполнены  – Кнопка «Создать» активна  – Кнопка «Удалить» активна  – Кнопка «Сохранить» активна | Пройден |
| 1. Нажимаем кнопку «Создать» | –Информация поступает в базу данных, отображение в системе, после обновления | Пройден |
| 1. Заполнение информации о клиенте:   – Имя: Александра  – Фамилия: Смелая  – Отчество: Федоровна  – Телефон: +7866969686 | – Окно открыто  – Поля не заполнены  – Кнопка «Создать» активна  – Кнопка «Удалить» активна  – Кнопка «Сохранить» активна | Пройден |
| 1. Нажимаем кнопку «оформить заказ» | – Сообщение с успешным оформлением заказа, информация поступает в базу данных | Пройден |

Таблица 2.3 – Негативный тест кейс

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Результат теста |
| 1. Вход в систему   – ешщуцтмнцз  – \*\*\*йм4ещзцищш | – Окно открыто  – Поля заполнены  – Кнопка «Войти» неактивна  – Сообщение о проверке введенных данных | Пройден |
| 1. Заполнение информации о клиенте:   – Имя: 865968  – Фамилия: !»№434  – Отчество: 77  – Телефон: ннн | – Окно открыто  – Поля заполнены  – Кнопка «Создать» не активна  – Кнопка «Удалить» не активна  – Кнопка «Сохранить» не активна  – Сообщение о проверке введенных данных | Пройден |
| 1. Заполнение полей администратором для добавления сотрудника:   – Имя: ++»;;  – Фамилия: 03-203-  – Отчество: 898№  – Дата рождения: 26.05.2001  – Телефон: +!  – Паспорт:1 | – Окно открыто  – Поля заполнены  – Кнопка «Создать» не активна  – Кнопка «Удалить» не активна  – Кнопка «Сохранить» не активна  – Сообщение о проверке введенных данных | Пройден |

Также была проведена проверка на вирусы в нашей программе, рисунок 2.7 с результатом показан ниже.

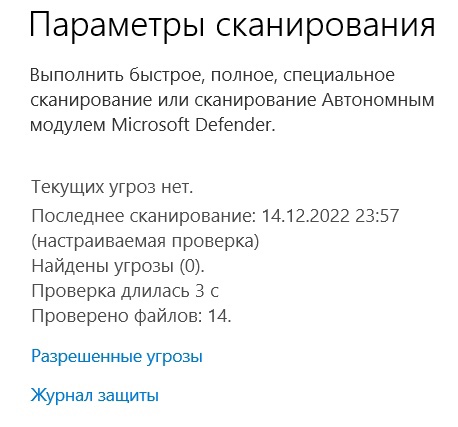


Рисунок 2.7 – Параметры сканирования

Заключение

Целью данного курсового проекта являлось написание программы по теме: «Разработка ИС «Фотоцентр»», которая была достигнута в ходе выполненной работы.

В ходе написания курсового проекта были получены практические навыки в области проектирования программы и создании БД. Программа разрабатывалась с использованием стратегии анализа сообщений, реализующей метод нисходящей разработки. Анализ основывался на изучении потоков данных, обрабатываемых программой. Опыт, полученный при написании работы, подтвердил необходимость в первую очередь правильной выработки требований к программе и грамотного подхода к её проектированию перед началом написания кода.

Так же были получены практические навыки при оптимизации и тестировании программы. Появилось понимание важности и сложности отладки, её затратности по времени относительно всего процесса разработки, необходимости знания аспектов из разных областей IT-технологий, как, например, знание аппаратной части, операционных систем, самих реализуемых процессов, природы и специфики различных ошибок и т.д.

Результатом проведенной работы является правильно спроектированное приложение и БД, а также работающий, эффективный, отлаженный и корректно отвечающий тестам соответствующих условий тестирования проект.

При разработке программы были изучены и применены методы проектирования БД. Программа имеет надлежащую структуру, схемы и необходимые комментарии.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

* + 1. Основы проектирования баз данных: Метод. указания/ Казан. гос.технол.ун-т; Сост.: Г.А. Гадельшина, А.В. Аксянова, А.Н. Валеева, Е.Н. Нуруллина. Казань, 2010. 60 с. 3.
    2. Разработка и реализация информационной модели предметной облсти: Метод. указания/ Казан. гос.технол.ун-т; Сост.: Г.А. Гадельшина, А.В. Аксянова, Л.Ю. Кошкина. Казань, 2009. 55 с.
    3. .Линн Бейли. Изучаем SQL. Питер. 2012.
    4. .К. Янк. PHP & MySQL: Novice to Ninja Пятое издание. Эксмо. 2013.
    5. Кумскова И.А. Базы данных (1-е изд.) М.: КноРус. 2016.
    6. Кумскова И.А. Базы данных (2-е изд.) М.: КноРус. 2016.
    7. Линн Бейли. Изучаем SQL. Питер. 2012.
    8. А.М. Вендров. Сase- технологии. Современные методы и средства проектирования информационных систем. – М.; Финансы и статистика, 1998.
    9. В.В. Коваленко. Проектирование ИС. РГРТУ, Рязань, 2006.
    10. Волков Ю.Г. Как написать диплом, курсовую, реферат? Ростов-на-Дону, 2001
    11. Проектирование экономических  информационных систем: Учебник. Г.Н. Смирнова, А.А. Сорокин, Ю.Ф. Тельнов; Под ред. Ю.Ф. Тельнова. –М.: Финансы и статистика, 2002.
    12. С.В. Маклаков. ERwin и Bpwin. CASE-средства разработки информационных систем.М.,1999.
    13. С.В. Маклаков. Создание информационных систем с ALLFusion Modeling Suite. М.,2003.

Приложение 1. Устав

Таблица 1.1 – Устав проекта

|  |  |
| --- | --- |
| Название проекта | ИС «Фотоцентр» |
| Бизнес-причина возникновения проекта | Наш проект станет уникальным продуктом на рынке фотобизнеса. Разрабатываемая ИС позволит новаторски автоматизировать процесс коммуникации с клиентом и процесс отправки файла, что позволит значительно увеличить прибыль, а также сократить издержки. |
| Бизнес-цель | * Увеличить прибыль; * Минимизация издержек; * Повышение спроса и повышение охватов; * Выделиться среди конкурентов; * Создать удобное в использовании приложение; |
| Требования, удовлетворяющие потребности, пожелания и ожидания заказчика, спонсора и других участников проекта | * ИС должна соответствовать следующим поставленным критериям: * прием заказов должен осуществляться в максимально удобном формате и для заказчика, и для сотрудника - формы заказа должны быть интуитивно понятными, а выходной файл кратким и информативным; * интерфейс перечня услуг компании должен соответствовать поставленной маркетинговой стратегии компании, быть легко редактируемым; * при оформлении брони, в системе должна быть предусмотрена система подтверждения и внесения предоплаты; * загрузка файлов на печать должна осуществляться во всех форматах без задержки; * у каждого клиента должен быть доступ к чату с сотрудником, ведущим его заказ; * отслеживать статус выполнения заказа можно круглосуточно; * ко всем фото применяется система резервного копирования во избежания потери файлов; * форма логинизации/регистрации должна содержать атомарные поля ввода, а также всплывающие подсказки на заполнение; * форма выбора файла должна содержать компоненты для множественного отображения файлов фото формата, интуитивно понятную навигацию, всплывающие подсказки; * форма прикрепления файла должна содержать объект для загрузки файлов, кнопки редактирования элементов; * форма отслеживания готовности заказа должна содержать объект вывода изображений, элемент вывода информации; * форма оплаты заказа должна содержать атомарные поля для реквизитов и всплывающие подсказки; * форма услуг должна содержать объекты отображения файлов разного формата для соответствия ее дизайн-проекту. |
| Расписание основных контрольных событий | Время начала проекта - 10.09.2022  Окончание разработки основного кода - 25.09.2022  Окончание предварительного тестирования с внесение поправок - 01.10.2022  Полное тестирование и оформление сопутствующей документации - до 15.10.2022  Время окончания проекта - 20.10.2022 |
| Участники проекта | Команда разработчиков:  Низамутдинова Анастасия - главный программист, разработчик интерфейсов.  Сонина Анастасия - второй программист, специалист по работе с данными, редактор интерфейсов.  Пугачева Алиса - специалист по работе с документацией, специалист по работе с пользователем.  Потенциальный покупатель ИС:  любое предприятие, оказывающее фотоуслуги |
| Окружение проекта | Данная ИС востребована на рынке, так как при проведении анализа рынка аналогов данному программному продукту обнаружено не было. Благоприятно влияет на проект востребованность отечественного ПО на фоне политической ситуации, а также льготы, распространяющиеся на проекты данного типа в сфере информационных технологий по текущему законодательству. Большая конкуренция среди разработчиков ограничивает сроки реализации проекта по причине его актуальности в данный период времени. Отсутствие первоначального капитала, влияет на выбор платформы разработки и состава команды, а также стоимость конечного продукта не в лучшую сторону. |
| Допущения относительно организации и окружения, а также внешние допущения | * команда проекта достаточно компетентна для выполнения проекта данного уровня; * работа в команде разработчика слаженная, задачи распределены справедливо, а все части проекта готовы к нужному сроку; * потенциальный заказчик платежеспособен и нуждается в данной ИС; * условия рынка позволяют продать готовый продукт выгодно, с достаточным процентом прибыли |
| Ограничения относительно организации и окружения, а также внешние ограничения | * процент прибыли от стартовой продажи должен покрыть не менее 50% расходов на реализацию проекта; * каждый этап реализации должен быть выполнен в срок с допустимым опозданием по сдаче не более 12 часов; * проект не должен противоречить введенному в процессе реализации законодательству; * рабочие условия команды разработчиков должны соответствовать требованиям трудового кодекса РФ: * потенциальный заказчик должен иметь всю необходимую документацию для законного оформления покупки ИС; * передача прав владения ИС после оплаты заказчиком производится в короткие сроки |
| Объем денежных средств, выделенных на достижение бизнес-цели | 200р (вложилась команда разработчиков и Горин Григорий на пару плиток шоколада для улучшения работы мозга) |
| Назначение руководителей проекта и общее определение полномочий ключевых членов проектной команды: РП, спонсор, координатор | Руководитель проекта - Сонина Анастасия. Связующее звено между пользователем, документацией и главным разработчиком, видит проект целиком.  Координатор - Низамутдинова Анастасия. Главный программист, имеет возможность оценить текущее состояние проекта и выявить слабые стороны, нуждающиеся в доработке.  Бизнес-менеджер - Пугачева Алиса.  Оценивает ситуацию на рынке, выявляет экономически сильные и слабые стороны проекта. Занимается анализом целевой аудитории и построение стратегии продвижения продукта. |

Приложение 2. Техническое задание

# Введение в техническое задание

Информационные процессы значительно облегчают работу во всех сферах обслуживания, в том числе и в сфере услуг. Автоматизирование этих процессов упорядочивает все стадии работы, повышает производительность и в конечном итоге, повышает качество обслуживания.

Анализ рынка фотоуслуг обосновывает вывод команды проекта об отсутствии качественного бюджетного продукта для базовой автоматизации процесса работы с клиентом при оказании услуг в сфере фотографии. Существующие продукты в основном обладают достаточно узким функционалом и требуют специального оборудования, дополнительного программного обеспечения и особых навыков для его использования.

Настоящее техническое задание распространяется на автоматизацию информационных процессов в фотоцентре.

# Основание для разработки

# 2.1 Программа разрабатывается на основе учебного плана специальности 09.02.07 «Информационные системы и программирование» 2.2 Наименование работы: ИС «Фотоцентр». 2.3 Исполнители: студентки 3 курса МТКП МГТУ им. Н.Э. Баумана Низамутдинова Анастасия Ринатовна, Пугачева Алиса Евгеньевна, Сонина Анастасия Сергеевна.

# Назначение разработки

Программа предназначена для автоматизации ведения учета заказов фотоцентра. Разрабатываемый программный продукт обеспечит быстрый и надежный обмен данными с клиентами, ускорит процесс реализации поступивших заказов, а также даст возможность формировать отчетность и анализировать данные без использования дополнительного программного обеспечения.

1. Требования к программному продукту

4.1 Требования к функциональным характеристикам:

* Гибко контролировать информационные процессы в организации;
* Снижать вероятность возникновения ошибок, вызванных человеческим фактором;
* Поддерживать разграничение прав доступа к ресурсам на основании пользовательских ролей, а также иметь защиту от несанкционированного доступа: добавлять новых сотрудников может только администратор, сотрудник может только просматривать заказы и изменять статус заказа, клиент может создать новый заказ и просмотреть запись;
* Информационная система должна иметь интуитивно-понятный графический пользовательский интерфейс;
* Фиксировать все изменения по заказам и другим информационным потокам в базе данных.

Основной задачей проектируемой системы является автоматизация процесса общения с клиентом с целью сокращения времени реализации заказа. Для достижения поставленной задачи необходимо выполнение описанных далее требований.

*Подсистема ввода данных должна реализовывать следующий набор функций:*

* + Наличие форм ввода информации в систему;
  + Наличие автоматического заполнения элементов форм;
  + Иметь возможность проверки данных перед записью;
  + Иметь возможность передачи данных в систему хранения.

*Система хранения данных должна реализовывать следующий набор функций:*

* Запись данных в базу данных
* Обеспечение защиты информации;
* Обеспечение целостности информации;
* Передача данных по запросу пользователя;
* Возможность журнализации изменений;
* Средства отображения информации в удобном для пользователя формате;
* Возможность вывода информации на печать;
* Расчет статистических данных;
* Возможность отмены внесенных изменений.

1. Подбор предложений

Работа фотоцентра направлена на оказание услуг в сфере фотографии и включает в себя печать фотографий, их обработку, реализацию фотографий на документы.

Реализуемая информационная система позволит клиенту познакомиться с перечнем услуг, выбрать необходимую и оформить заказ.

***Формирование услуги.*** Полученную в ходе процесса оформления заказа информацию о требуемых услугах система должна сопоставлять с анализом факторов работы, выявленными работниками организации. Под факторами подразумеваются критерии ценообразования (работа фотографа, расходный материал), нормативную документацию (сертификаты на работу), материальную часть(затраты на содержание штата, аренду и прочее).

***Хранение технической информации.*** Результат деятельности системы, все этапы принятия решений, факторы принятия решений должны журнализироваться для возможности выявить и исправить возникающие ошибки.

***Требования надежности.*** Система, управляющая автоматизацией и настройкой программного обеспечения должна иметь стандартные и продвинутые настройки безопасности. В их число входит создание пароля и логина администратора системы, защита от опасных изменений, журнализирование всех действий системы, регулярное обслуживание системы (в которое входит тестирование и выпуск исправлений).

Надежное функционирование информационной системы должно быть обеспечено выполнением организационно-технических мероприятий, таких как:

* использование лицензионного программного обеспечения;
* программное обеспечение должно быть совместимо с операционными системами macOS, Windows 7/10/11 в соответствии с требованиями заказчика и оборудованием организации.

**Требования к интерфейсу**. Система должна иметь удобный и подробный интерфейс. Интерфейс должен быть выполнен в минималистичном стиле, иметь однозначно интерпретируемые кнопки.

# Требования к составу и параметрам технических средств

Минимально допустимая конфигурация клиентского рабочего места администратора системы:

* Платформа х86, процессор INTEL Pentium Dual-Core G4400 или аналогичный;
* Оперативная память объемом 4 ГБ;
* 1 ГБ памяти на жестком диске;
* Видеоадаптер SVGA;
* Монитор с разрешением 1024 Х 768;
* Клавиатура, компьютерная мышь;
* Наличие доступа в Интернет.

# Требования к составу программных средств

Для корректной работы подсистемы необходимо наличие следующих программных средств:

* Операционная система Windows 7/8/10/11, macOS;
* Программное обеспечение для развертывания и работы базы данных;
* Драйвера устройств.

# Требования к программной документации

Документация должна быть разделена на описание интерфейсной части системы и низкоуровневой, а также содержать требования к конфигурации системы, руководство по установке программного обеспечения, объяснение нюансов работы с ним и с системой в целом (ограничения, предупреждения). Разъяснения сообщений, которые система может оставлять при журнализации, отображении ошибок и иных уведомлений.

Перечень сопроводительной технической документации:

* Руководство пользователя;
* Руководство программиста.

***Стандарты.***  
При проектировании системы должны быть учтены следующие стандарты:  
ГОСТ 19.102-77 – Стадия разработки;  
ГОСТ 19.401-78- Требования к содержанию документов;  
ГОСТ 19.201-78- Требования к содержанию и оформлению документов;  
ГОСТ 34.601-90-Стадия создания автоматизированной системы;  
ГОСТ 34.698-90- Требования к содержанию и оформлению;  
ГОСТ 19.507-79- Ведомость эксплуатационных документов;  
ГОСТ 19.401.78- Требования к содержанию;  
ГОСТ 2.101-68- Единая система конструкторской документации;  
ГОСТ 34.603-92- Виды испытаний автоматизированных систем;  
ГОСТ 24.208-80- Требования к содержанию документов стадии «Ввод в эксплуатацию»;  
ГОСТ 19.301-79- Программа и методика испытаний. Требования к содержанию и оформлению;  
ГОСТ 15.309-88- Испытание и приемка выпускаемой продукции.

1. Календарный план работ

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название этапа | Сроки этапа | Чем заканчивается этап |
| 1 | Разработка технического задания | 21.10.2022 г. –  25.10.2022 г. | Готовое ТЗ |
| 2 | Обоснование необходимости разработки программы. Постановка задачи | 26.10.2022 г. –  28.10.2022 г. | Четко сформулированные требования для программы |
| 3 | Сбор исходных материалов | 20.10.2022 г.- 11.11.2022 г. | Готовые материалы для разработки АИС |
| 4 | Определение требований к программе | 24.10.2022 г.- 12.11.2022 г. | Готовые требования к программе |
| 5 | Согласование и утверждение ТЗ | 12.10.2022 г. | Согласованное и утвержденное ТЗ |

Приложение 3. Руководство по установке

Руководство по установке

(Инструкция)

1. Открываем файл PhotoCenter.exe.

2. Выбираем язык установки, как показано на рисунке 3.1.

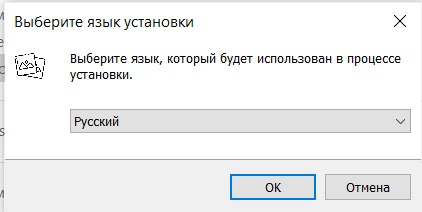


Рисунок 3.1 – Выбор языка

3.Принимаем условия соглашения, как на рисунке 3.2.

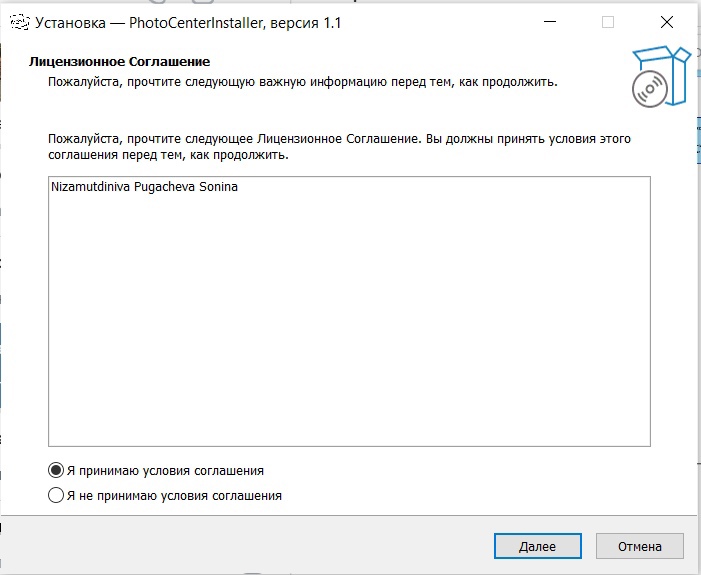


Рисунок 3.2 – Лицензионное соглашение

1. Ввод пароля показан на рисунке 3.3.

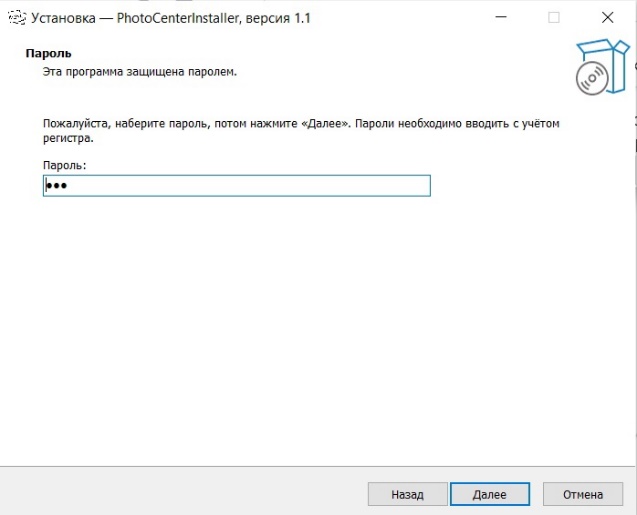


Рисунок 3.3 – Ввод пароля

1. Выбор места загрузки программы на компьютере показан на рисунке 3.4.

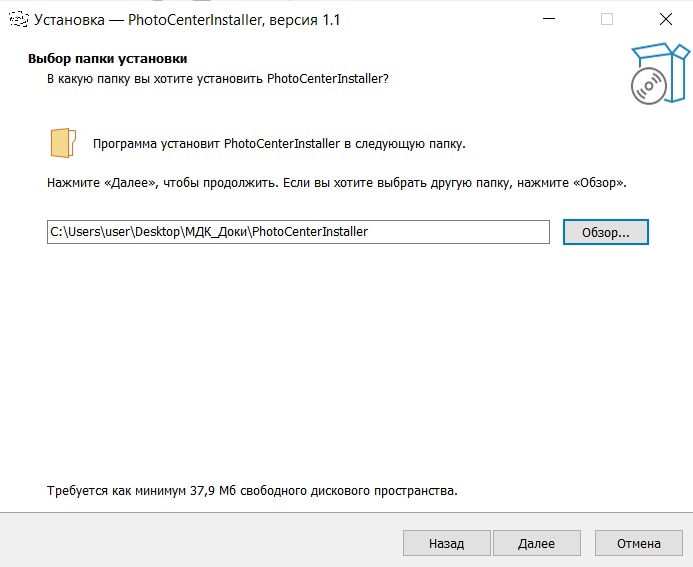


Рисунок 3.4 – Выбор места установки

1. Программа установлена, результат показан на рисунке 3.5.

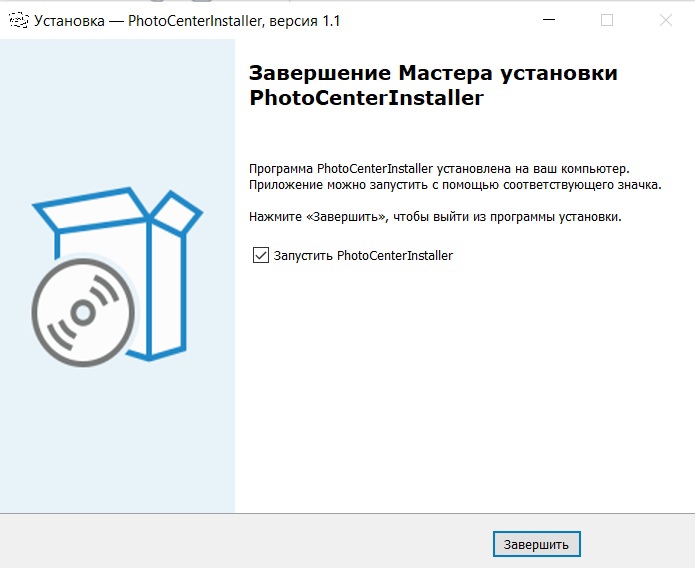


Рисунок 3.5 – Завершение установки

Приложение 4. Руководство пользователя

Руководство пользователя

(Инструкция)

1. Установка приложения «Фотоцентр»
2. Стандартный запуск Приложения «Фотоцентр»
3. Вход в приложение осуществляется посредством ввода логина и пароля, созданные администратором. На рисунках 4.1-4.2 изображён интерфейс входа в систему.

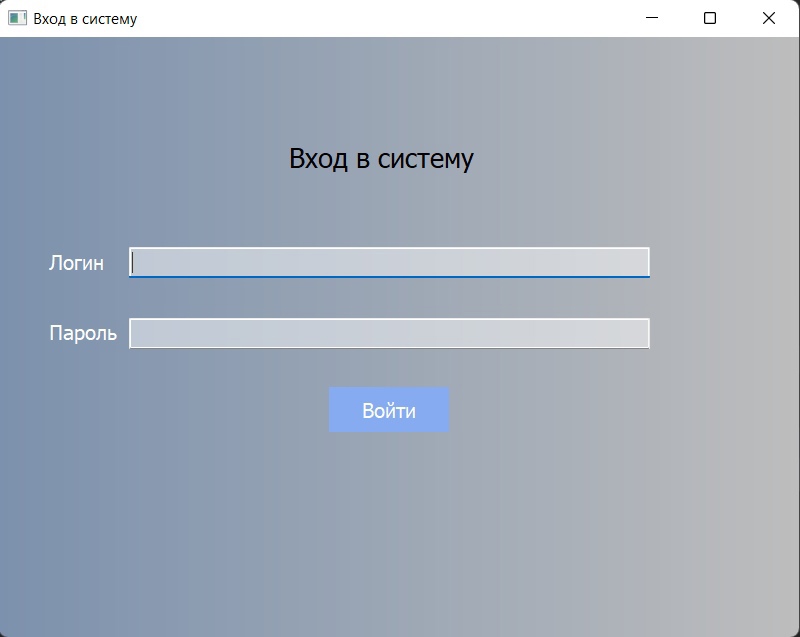


Рисунок 4.1 – Вход (поля не заполнены)

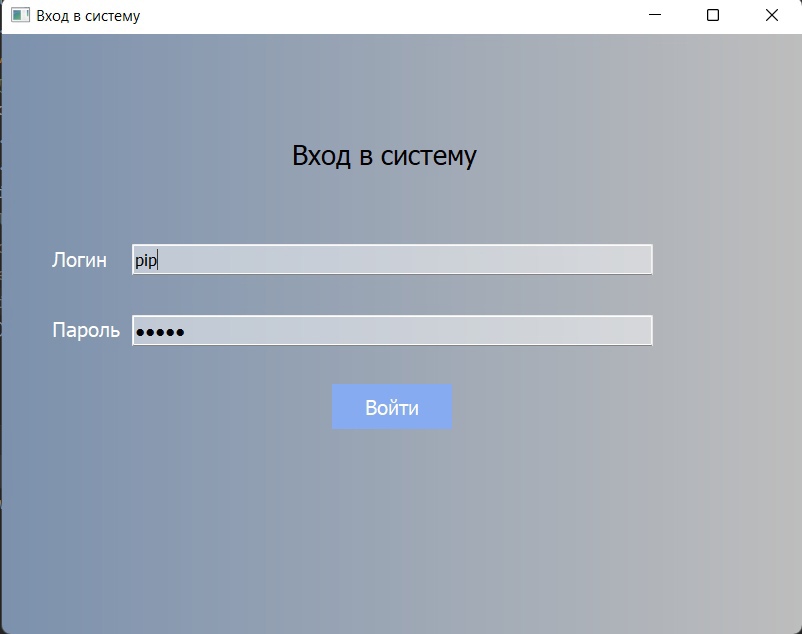


Рисунок 4.2 – Вход (поля заполнены)

1. После заполнения полей, нажмите на кнопку «Войти», после нажатия приложение перенесет вас на страницу с услугами фотосалона. Если Вас нет в системе, то вы обнаружите уведомление, показанное на рисунке 4.3.

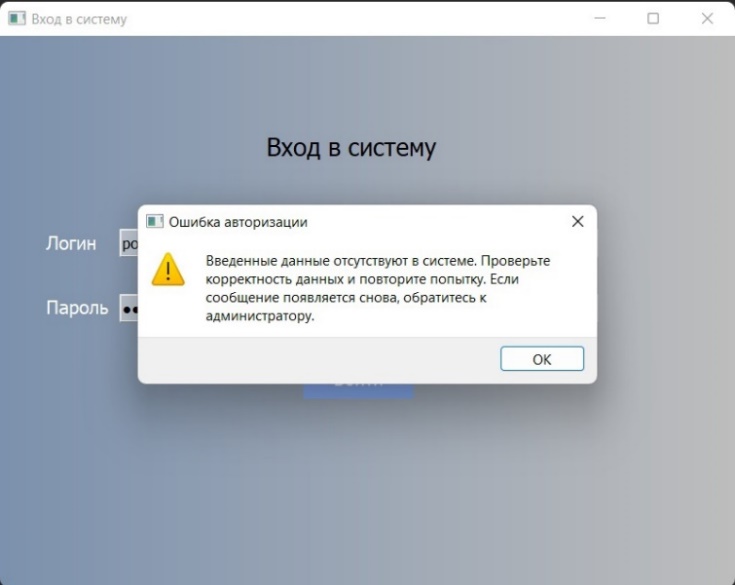


Рисунок 4.3 – Уведомление об ошибке

1. После входа в систему, пользователю (сотруднику) открывается доступ к просмотру и добавлению заказов. Рисунок 4.4, изображающий меню с заказами, показан ниже.

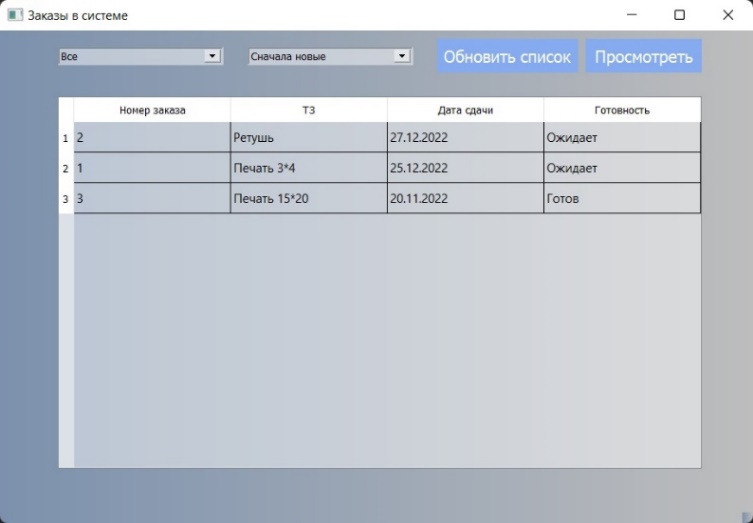


Рисунок 4.4 – Окно заказы в системе

1. В этом меню можно обновить список заказов, посмотреть закрытые(готовые) и открытые (в процессе) заказы, также отсортировать по новизне. Также при выделении заказа и нажатии кнопки «Просмотреть» у пользователя пояляется возможность изучить детали заказа и изменить его статус кнопкой «Заказ выполнен». Подробнее показано на рисунках 4.5 - 4.7.

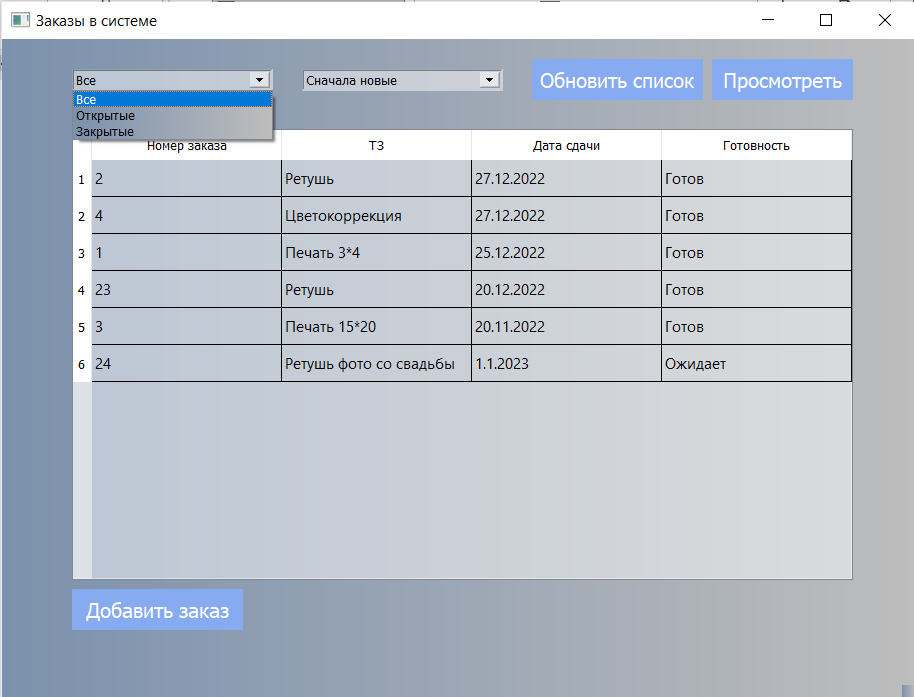


Рисунок 4.5 – Выбор статуса заказа для просмотра

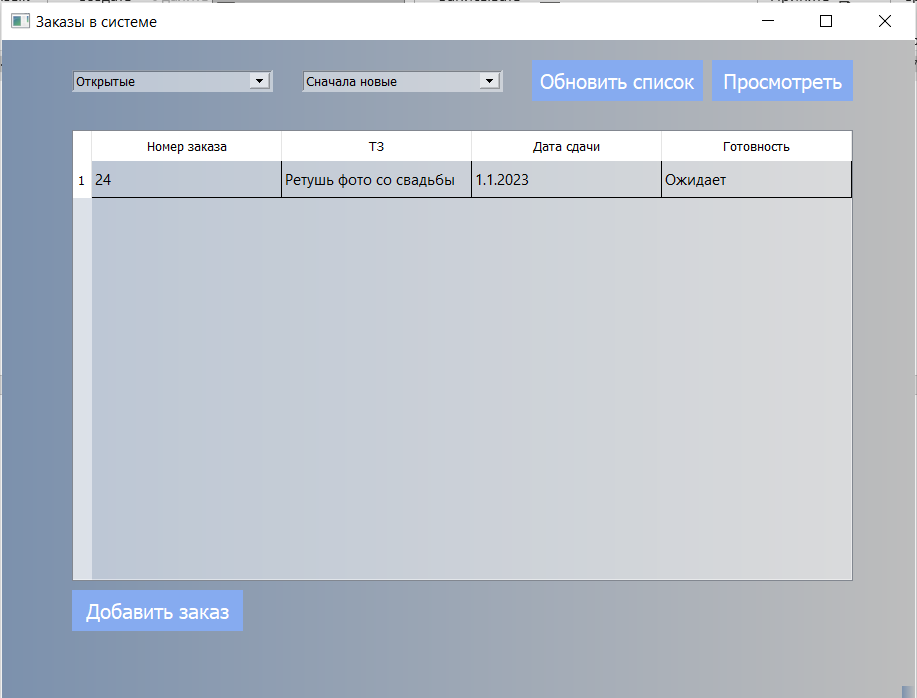


Рисунок 4.6 – Демонстрация открытых заказов

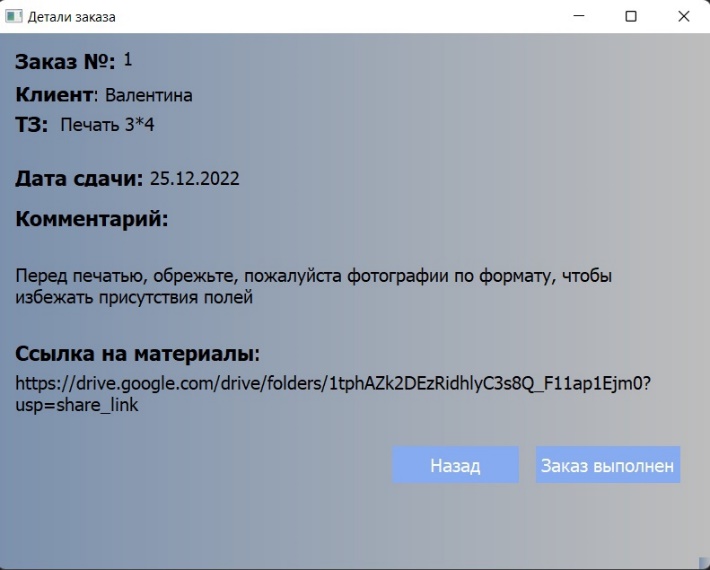


Рисунок 4.7 – Подробности заказа

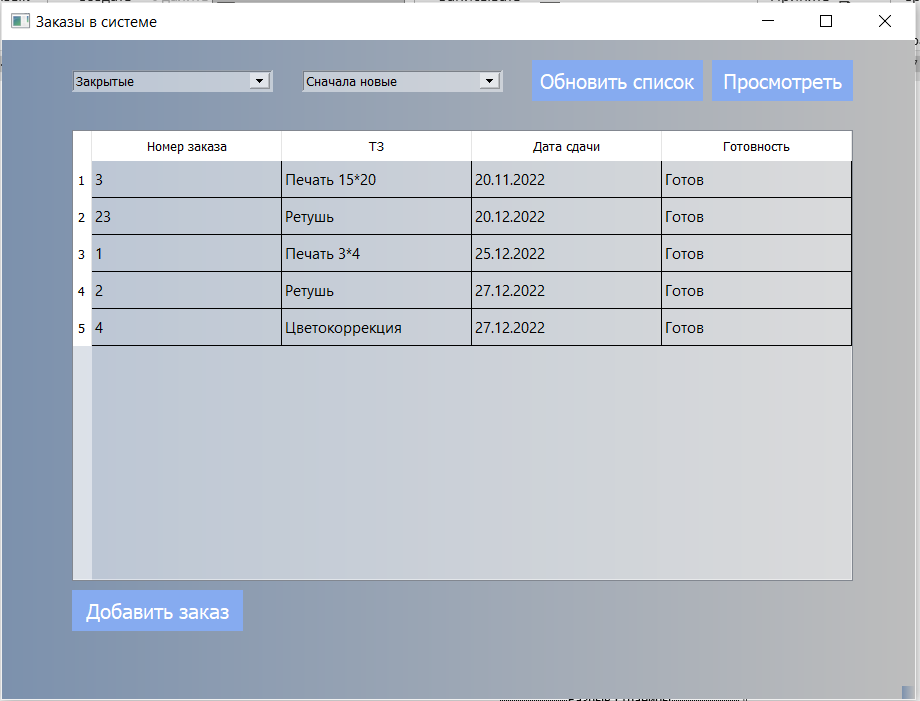


Рисунок 4.8 – Окно «Заказы в системе» с новым статусом просматриваемого заказа(была нажата кнопка «Заказ выполнен»

Приложение 5. Руководство администратора

Руководство администратора

(Инструкция)

1. Установка приложения «Фотоцентр».
2. Стандартный запуск Приложения «Фотоцентр»
3. Вход в приложение осуществляется посредством ввода логина и пароля. На рисунках 5.1-5.2 изображён интерфейс входа в систему.

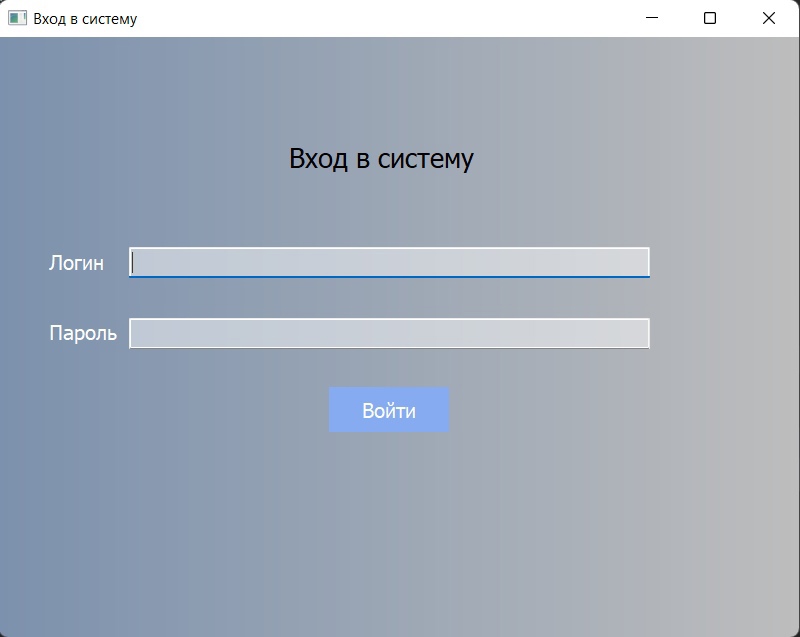


Рисунок 5.1 – Вход (поля не заполнены)

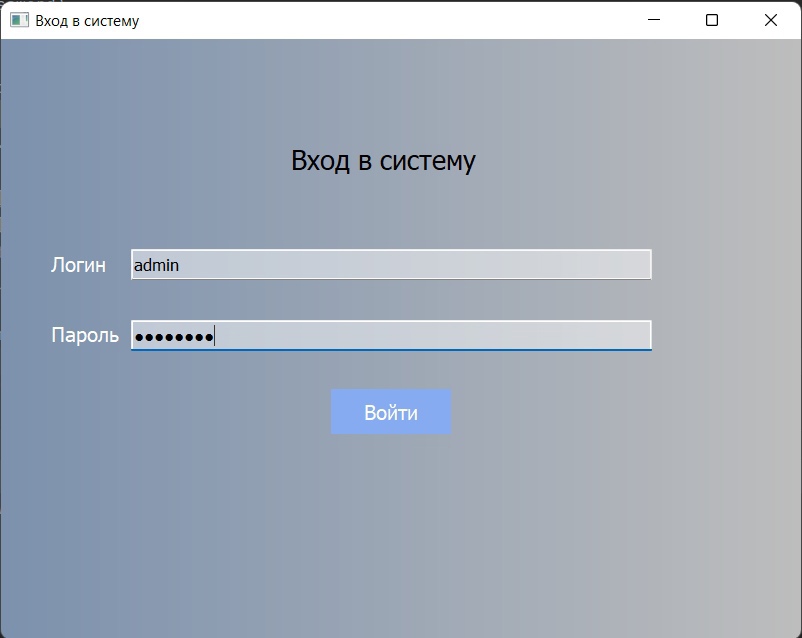


Рисунок 5.2 – Вход (поля заполнены)

1. После заполнения полей, нажмите на кнопку «Войти», после нажатия приложение перенесет вас на страницу с меню. В меню можно выбрать вкладку о заказах, сотрудниках и клиентах, показано оно на рисунке 5.3.

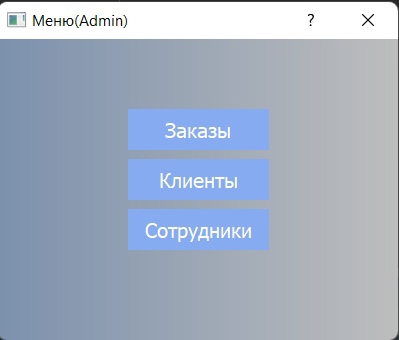


Рисунок 5.3 – Демонстрация меню администратора

1. В каждом окне есть возможность добавить данные, изменить или удалить. Окно «клиенты» изображено на рисунке 5.4.

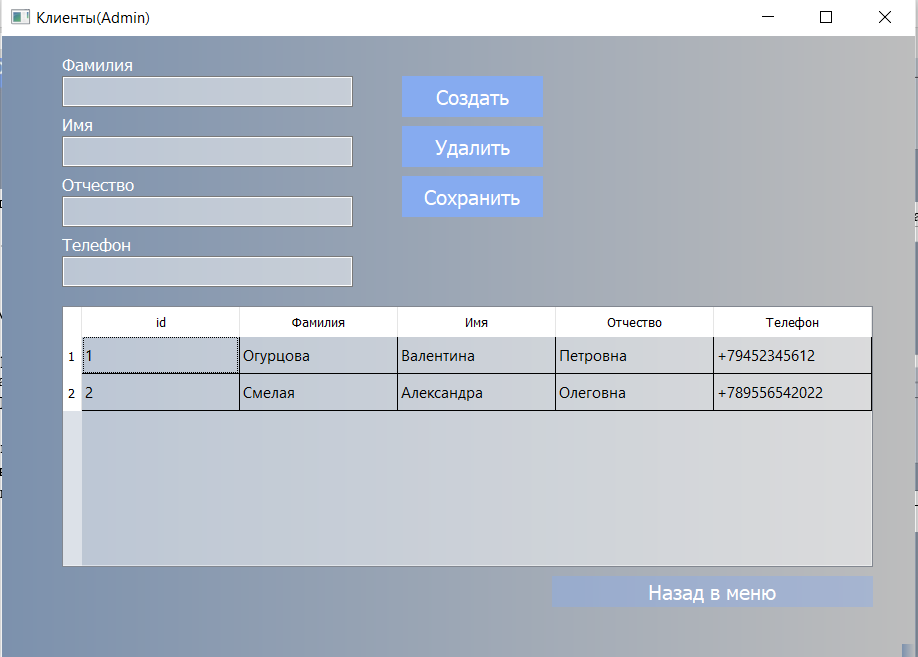


Рисунок 5.4 – Демонстрация окна «клиенты»

1. Заполнение полей о клиенте изображено на рисунке 5.5. Чтобы добавить нового клиента, нужно нажать на кнопку «сохранить». Если выбрать клиента и нажать на кнопку «удалить», то клиент будет удален из БД.

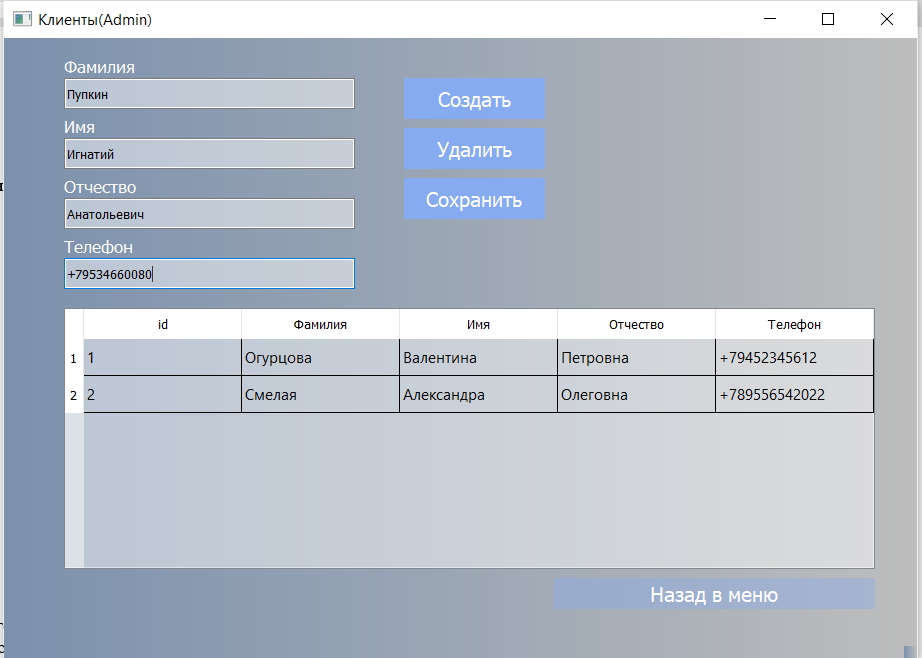


Рисунок 5.5 – Заполнение полей в окне «клиенты»

1. Сохранение данных о клиенте показано на рисунке 5.6.

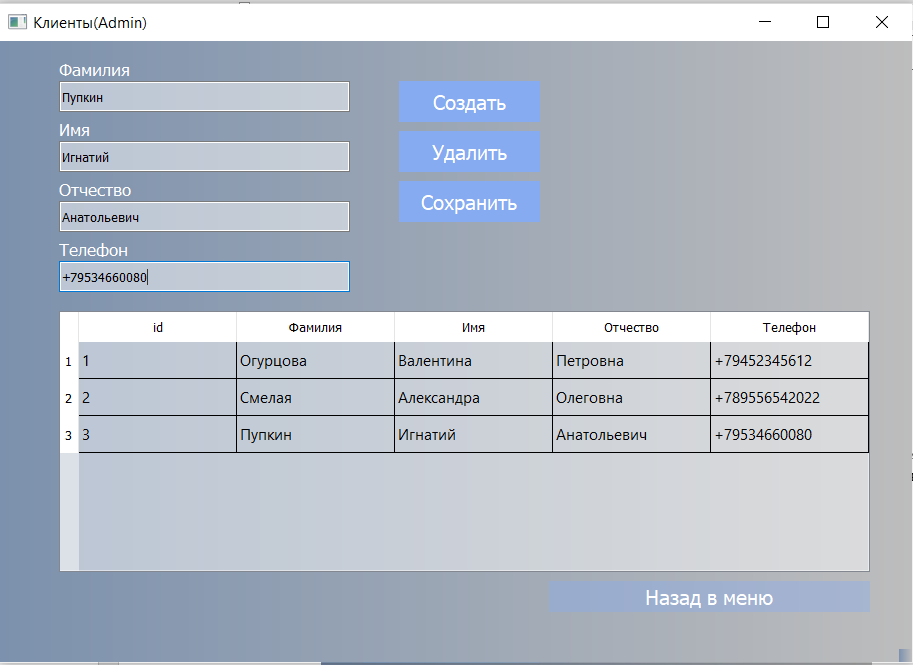


Рисунок 5.6 – Сохраненные данные о клиенте

1. Если нажать на поле, которое нужно изменить и заполнить его другими данными, затем нажать на кнопку «сохранить», то данные обновятся, это показано на рисунках 5.7-5.8.

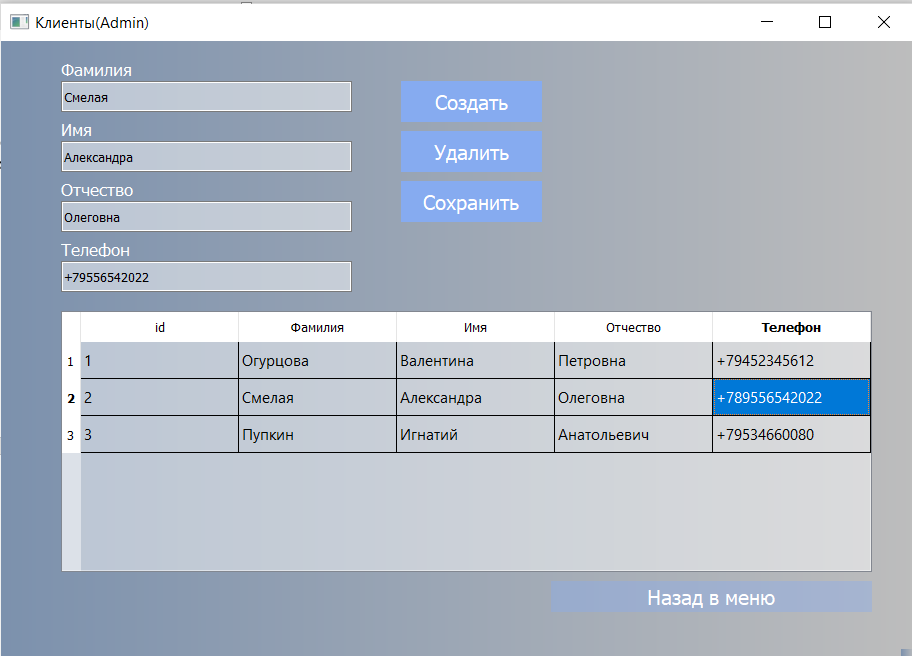


Рисунок 5.7 – Изменение данных

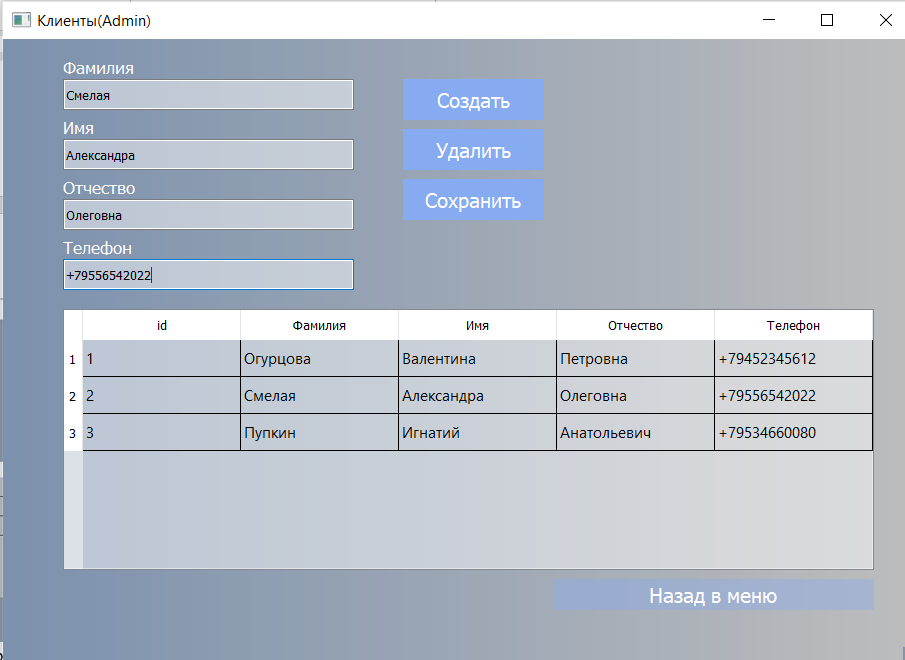


Рисунок 5.8 – Обновленные данные

Чтобы выйти из окна, нужно нажать на кнопку **«**Назад в меню**».**

1. Окно «сотрудники» изображено на рисунке 5.9. Здесь тоже можно обновлять, добавлять и удалять пользователей (сотрудников).

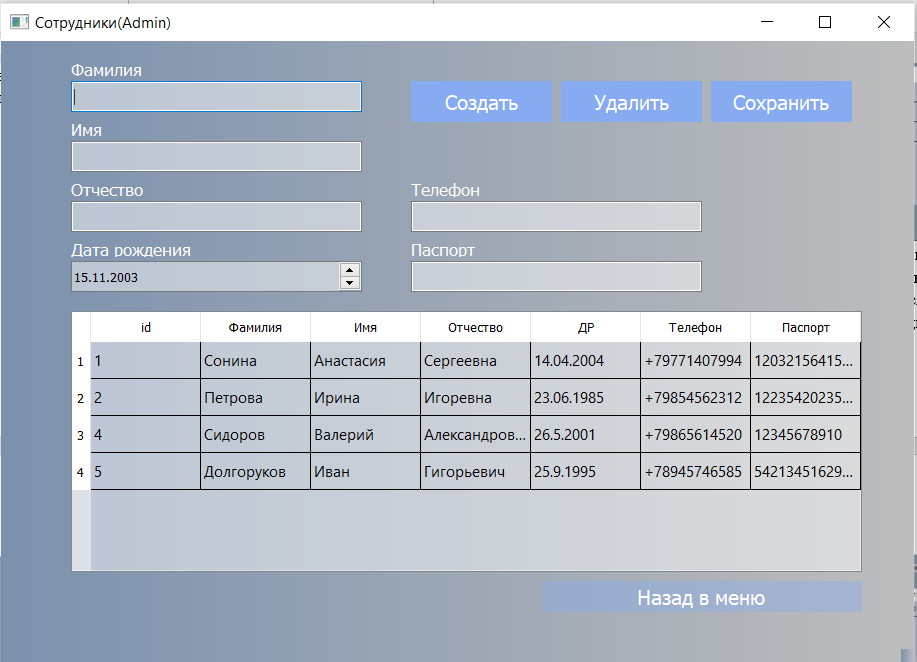


Рисунок 5.9 – Окно «Сотрудники»

1. При добавлении нового сотрудника приложение перемещает нас на новое окно, которое предлагает создать пароль и логин для пользователя системы, показано это на рисунке 5.10.

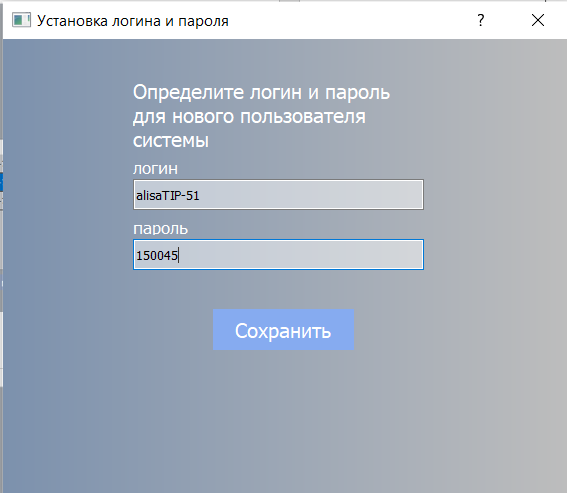


Рисунок 5.10 – Окно с определением логина/пароля для нового сотрудника

1. После наделения пароля и логина для нового сотрудника, в БД сохраняются данные о пользователе, результат можно увидеть на рисунке 5.11.

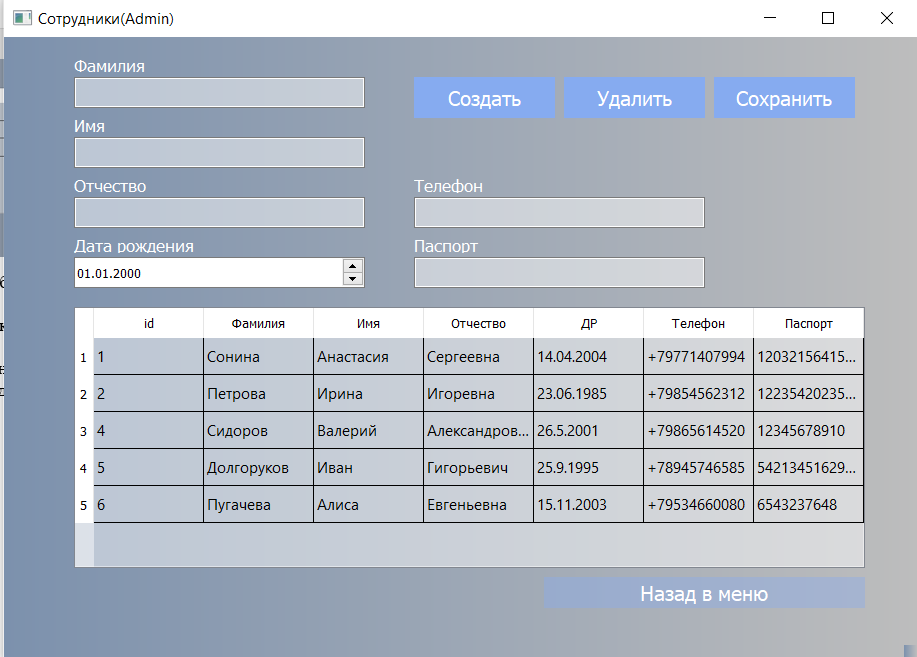


Рисунок 5.11 – Сохранение нового сотрудника

1. Во вкладке заказ можно просматривать актуальные и завершенные заказы, а также добавлять новые. Окно «заказ в системе» показано на рисунке 5.12.

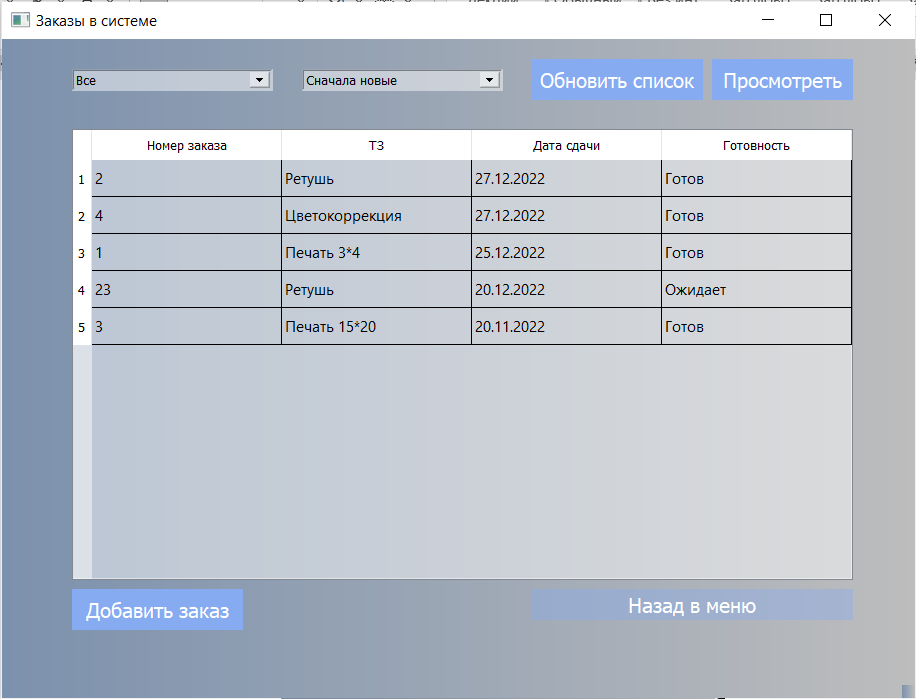


Рисунок 5.12 – Окно «заказ в системе»

1. В этом окне можно сортировать заказы по новизне (показано на рисунке 5.13) и по статусу готовности (показано на рисунке 5.14). Чтобы заказы отсортировались, нужно нажать на конку «Обновить список».

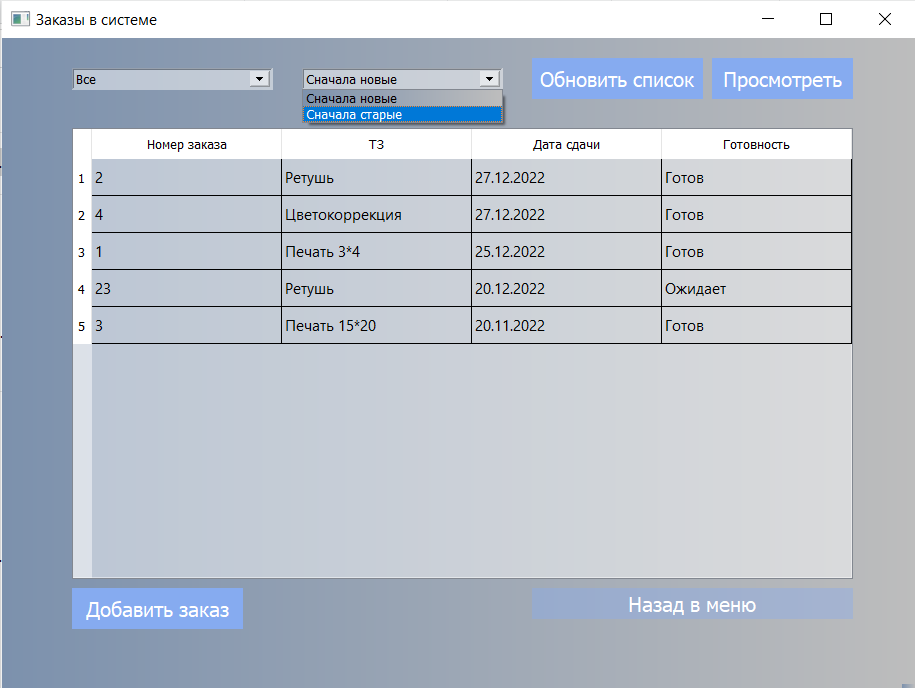


Рисунок 5.13 – Сортировка заказов по новизне

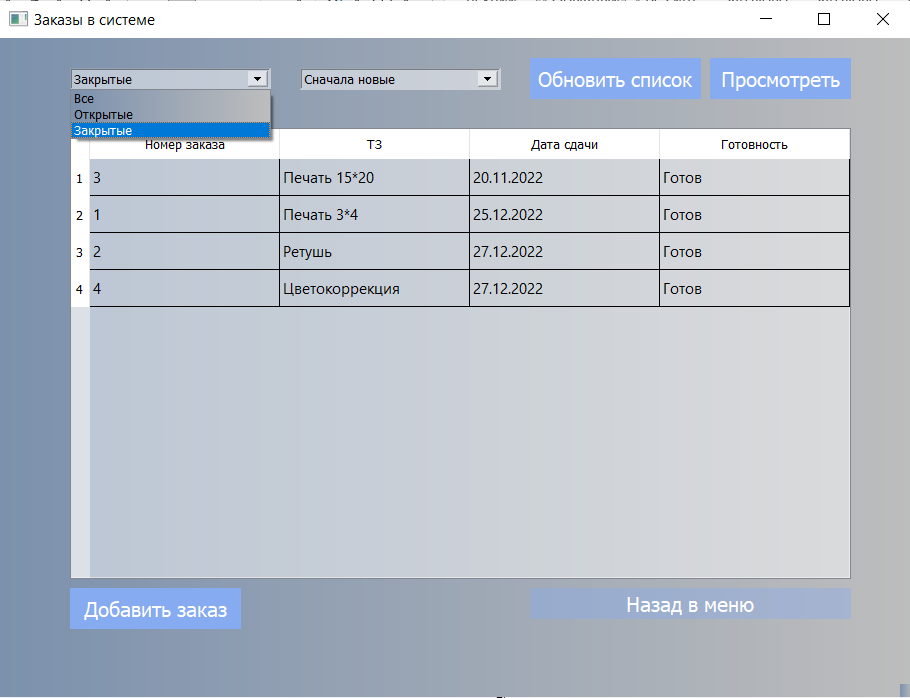


Рисунок 5.14 – Сортировка заказов по статусу готовности

1. В окне «Заказы в системе» можно посмотреть подробную информацию о заказе и изменить статус готовности, нажав на кнопку «Посмотреть», показано на рисунках 5.15-5.17.

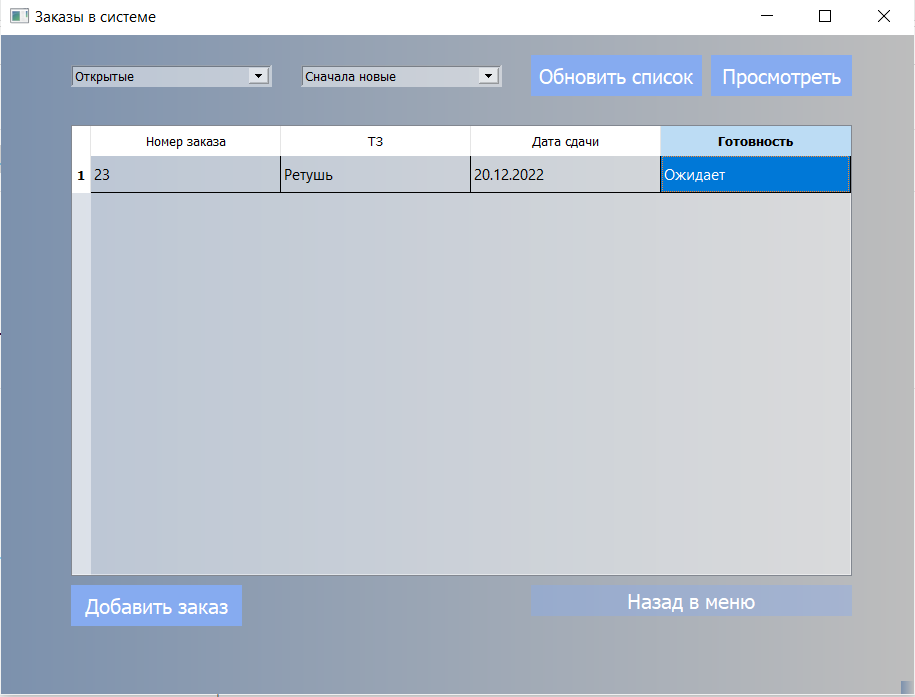


Рисунок 5.15 – Выбор заказа для просмотра и изменения

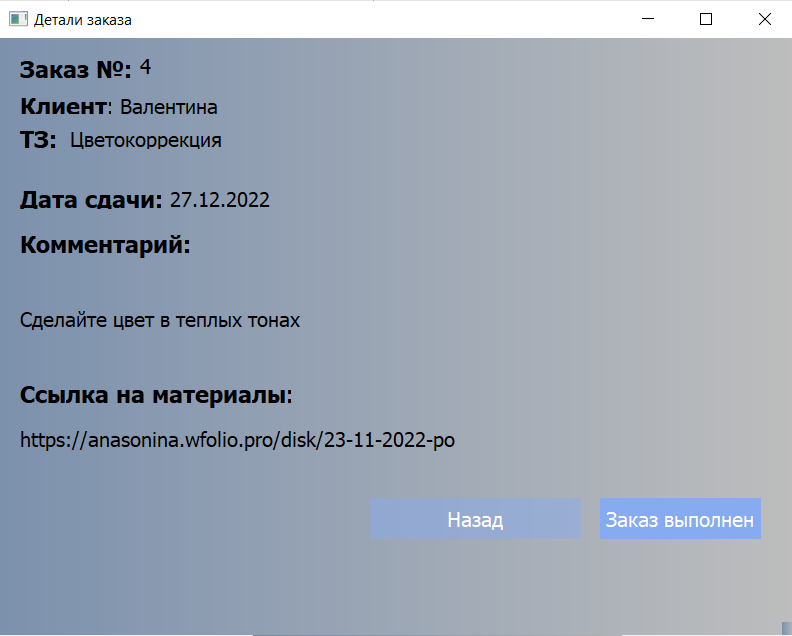


Рисунок 5.16 – Отображение информации о заказе

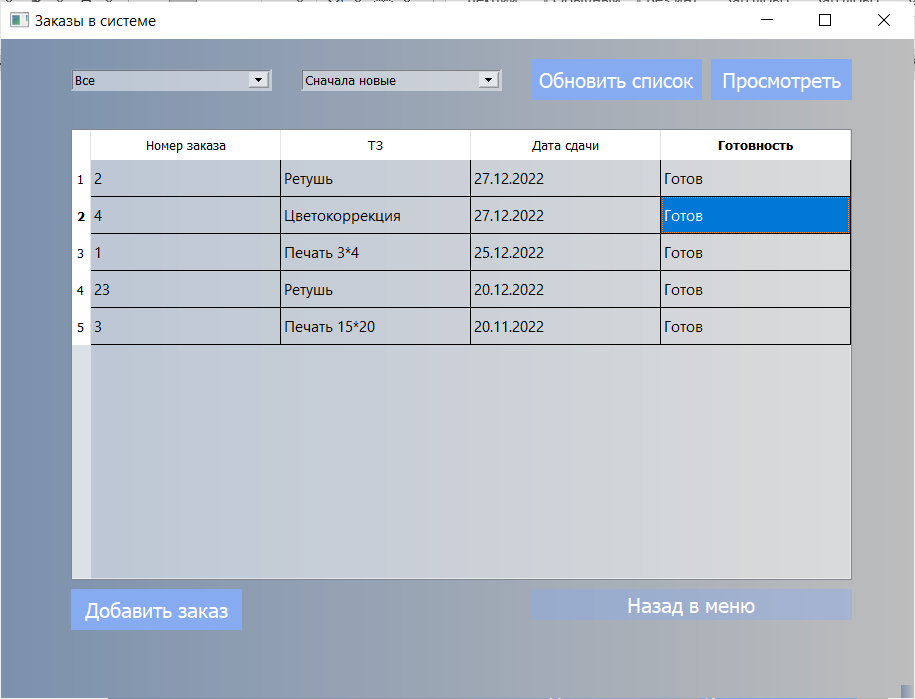


Рисунок 5.17 – Обновленная информация

Приложение 6. Программный код

Программный код

import sys

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

from PyQt5 import uic, QtWidgets

import sqlite3

value = ''

class WindowStart(QMainWindow):

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

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

uic.loadUi('UIlog.ui', self)

self.con = sqlite3.connect('todo.db')

self.cur = self.con.cursor()

self.password.setEchoMode(QtWidgets.QLineEdit.Password)

self.login.setFocus()

self.butlog.clicked.connect(self.vhod)

def vhod(self):

login = self.login.text()

password = self.password.text()

print(login + password)

#self.btnBack.setVisible(False)

try:

self.cur.execute(

f"SELECT `passw` FROM `logpass` WHERE `logg`='{login}';")

global value

value = self.cur.fetchone()

print(value)

if value != []:

if value[0] == password:

if value == ('pukpuk',):

self.close()

self.Menu = WindowMenu()

self.Menu.show()

else:

self.close()

self.Zak = WindowTodo()

self.Zak.show()

else:

error = QMessageBox()

error.setWindowTitle('Ошибка авторизации')

error.setText('Проверьте корректность данных и повторите попытку. Если сообщение появляется снова, обратитесь к администратору.')

error.setIcon(QMessageBox.Icon.Warning)

error.setStandardButtons(QMessageBox.Ok)

error.exec\_()

self.con.commit()

except:

error = QMessageBox()

error.setWindowTitle('Ошибка авторизации')

error.setText('Проверьте корректность данных и повторите попытку. Если сообщение появляется снова, обратитесь к администратору.')

error.setIcon(QMessageBox.Icon.Warning)

error.setStandardButtons(QMessageBox.Ok)

error.exec\_()

class WindowTodo(QMainWindow):

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

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

uic.loadUi('Zak.ui', self)

self.state = "new"

self.con = sqlite3.connect('todo.db')

self.cur = self.con.cursor()

states = ('Все', 'Открытые', 'Закрытые')

sorts = ('Сначала новые', 'Сначала старые')

self.datesort.addItems(sorts)

self.stateslist.addItems(states)

global value

if value != ('pukpuk',):

self.btnBack.hide()

self.btnBack.clicked.connect(self.GoBack)

self.btnRefresh.clicked.connect(self.ref)

self.btnShow.clicked.connect(self.show\_zak)

self.btnNew.clicked.connect(self.Tocreate)

self.FillTable()

def ref(self):

self.FillTable()

def show\_zak(self):

self.close()

self.Zak = WindowZak()

self.Zak.show()

def Tocreate(self):

self.close()

self.new = WindowNewzak()

self.new.show()

def Save(self):

self.con.commit()

self.FillTable()

def FillTable(self):

sort = self.stateslist.currentText()

datesort = self.datesort.currentText()

if sort == 'Все':

# self.cur.execute("SELECT `ID\_zak`, `title`, `todotime`, `state` FROM zak ORDER BY `todotime` DESC")

if datesort == 'Сначала новые':

self.cur.execute("SELECT `ID\_zak`, `title`, `todotime`, `state` FROM zak ORDER BY `todotime` DESC ")

else:

self.cur.execute("SELECT `ID\_zak`, `title`, `todotime`, `state` FROM zak ORDER BY `todotime`")

else:

if sort == 'Открытые':

if datesort == 'Сначала новые':

self.cur.execute(

"SELECT `ID\_zak`, `title`, `todotime`, `state` FROM zak WHERE `state`='Ожидает' ORDER BY `todotime` DESC")

else:

self.cur.execute(

"SELECT `ID\_zak`, `title`, `todotime`, `state` FROM zak WHERE `state`='Ожидает' ORDER BY `todotime`")

else:

if datesort == 'Сначала старые':

self.cur.execute(

"SELECT `ID\_zak`, `title`, `todotime`, `state` FROM zak WHERE `state`='Готов' ORDER BY `todotime` DESC")

else:

self.cur.execute(

"SELECT `ID\_zak`, `title`, `todotime`, `state` FROM zak WHERE `state`='Готов' ORDER BY `todotime`")

rows = self.cur.fetchall()

self.tblTodo.setColumnCount(4)

self.tblTodo.setHorizontalHeaderLabels(['Номер заказа', 'ТЗ', 'Дата сдачи', 'Готовность'])

self.tblTodo.setRowCount(0)

for row in rows:

inx = rows.index(row)

self.tblTodo.insertRow(inx)

# add more if there is more columns in the database.

self.tblTodo.setItem(inx, 0, QTableWidgetItem(str(row[0])))

self.tblTodo.setItem(inx, 1, QTableWidgetItem(str(row[1])))

self.tblTodo.setItem(inx, 2, QTableWidgetItem(str(row[2])))

self.tblTodo.setItem(inx, 3, QTableWidgetItem(str(row[3])))

header = self.tblTodo.horizontalHeader()

header.setSectionResizeMode(0, QtWidgets.QHeaderView.Stretch)

header.setSectionResizeMode(1, QtWidgets.QHeaderView.Stretch)

header.setSectionResizeMode(2, QtWidgets.QHeaderView.Stretch)

header.setSectionResizeMode(3, QtWidgets.QHeaderView.Stretch)

def GoBack(self):

self.close()

self.Menu = WindowMenu()

self.Menu.show()

class WindowNewzak(QDialog):

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

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

uic.loadUi('newzak.ui', self)

self.state = "new"

self.con = sqlite3.connect('todo.db')

self.cur = self.con.cursor()

self.cur.execute("SELECT `Surname` FROM klient")

kl\_surn = self.cur.fetchall()

self.cur.execute("SELECT `Name` FROM klient")

kl\_nam = self.cur.fetchall()

self.cur.execute("SELECT `Otch` FROM klient")

kl\_otc = self.cur.fetchall()

kolkl = len(kl\_surn)

print(kl\_surn, kl\_nam, kl\_otc, kolkl)

i = 0

klients = []

while i < kolkl:

fullname = ((str(kl\_surn[i])).replace('(', '')).replace(',)', '') +' '+ ((str(kl\_nam[i])).replace('(', '')).replace(',)', '') +' '+ ((str(kl\_otc[i])).replace('(', '')).replace(',)', '')

klients.append(fullname)

i = i+1

self.klist.addItems(klients)

self.btnSave.clicked.connect(self.Create)

def Create(self):

try:

widgetDate = self.dateDate.date()

selectedDate = str(widgetDate.day()) + "." + str(widgetDate.month()) + "." + str(widgetDate.year())

self.cur.execute("insert into zak (title, link, description, todotime, `state`, ID\_k, ID\_s) values (?, ?, ?, ?, ?, ?, ?)",

(self.txttitle.text(), self.txtlink.text(), self.txtdesc.text(), selectedDate, 'Ожидает', '1','1'))

self.con.commit()

self.close()

self.back = WindowTodo()

self.back.show()

except:

print('error')

class WindowPass(QDialog):

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

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

uic.loadUi('logpass.ui', self)

self.con = sqlite3.connect('todo.db')

self.cur = self.con.cursor()

self.txtlog.setText("")

self.txtpass.setText("")

self.btnSave.clicked.connect(self.Savelog)

def Savelog(self):

self.cur.execute("insert into logpass (logg, passw) values (?, ?)", (self.txtlog.text(), self.txtpass.text()))

self.con.commit()

self.close()

self.saveback = QMainWindow

self.saveback = WindowSotr()

self.saveback.show()

class WindowSotr(QMainWindow):

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

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

uic.loadUi('Sotr.ui', self)

self.state = "new"

self.con = sqlite3.connect('todo.db')

self.cur = self.con.cursor()

self.btnSave.setEnabled(False)

self.btnDelete.setEnabled(False)

self.btnBack.clicked.connect(self.GoBack)

self.btnNew.clicked.connect(self.AddNew)

self.btnSave.clicked.connect(self.Save)

self.btnDelete.clicked.connect(self.Delete)

self.tblTodo.itemSelectionChanged.connect(self.selectedRow)

self.FillTable()

def GoBack(self):

self.close()

self.Menu = WindowMenu()

self.Menu.show()

def GoNew(self):

self.close()

self.new = WindowNewzak()

self.new.show()

def AddNew(self):

self.state = "new"

self.txtSurname.setText("")

self.txtName.setText("")

self.txtOtch.setText("")

self.txtPhone.setText("")

self.txtPassport.setText("")

self.btnSave.setEnabled(True)

self.txtSurname.setFocus()

def Save(self):

widgetDate = self.dateDate.date()

selectedDate = str(widgetDate.day()) + "." + str(widgetDate.month()) + "." + str(widgetDate.year())

if (self.state == "new"):

self.cur.execute("insert into sotr (Surname, Name, Otch, Dater, Phone, Passport) values (?, ?, ?, ?, ?, ?)",

(self.txtSurname.text(), self.txtName.text(), self.txtOtch.text(), selectedDate,

self.txtPhone.text(), self.txtPassport.text()))

self.close()

self.logpass = WindowPass()

self.logpass.show()

if (self.state == "edit"):

self.cur.execute("update sotr set Surname=?, Name=?, Otch=?, Dater=?, Phone=?, Passport=? where ID\_sotr=?", (

self.txtSurname.text(), self.txtName.text(), self.txtOtch.text(), selectedDate, self.txtPhone.text(),

self.txtPassport.text(), self.id))

self.con.commit()

self.FillTable()

self.btnSave.setEnabled(False)

self.btnDelete.setEnabled(False)

def Delete(self):

try:

row = self.tblTodo.currentRow()

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

self.cur.execute("delete from sotr where ID\_sotr=?", (id,))

self.con.commit()

self.FillTable()

self.btnSave.setEnabled(False)

self.btnDelete.setEnabled(False)

except:

print("error")

def FillTable(self):

self.cur.execute("SELECT \* FROM sotr")

rows = self.cur.fetchall()

self.tblTodo.setColumnCount(7)

self.tblTodo.setHorizontalHeaderLabels(['id', 'Фамилия', 'Имя', 'Отчество', 'ДР', 'Телефон', 'Паспорт'])

self.tblTodo.setRowCount(0)

for row in rows:

inx = rows.index(row)

self.tblTodo.insertRow(inx)

# add more if there is more columns in the database.

self.tblTodo.setItem(inx, 0, QTableWidgetItem(str(row[0])))

self.tblTodo.setItem(inx, 1, QTableWidgetItem(str(row[1])))

self.tblTodo.setItem(inx, 2, QTableWidgetItem(str(row[2])))

self.tblTodo.setItem(inx, 3, QTableWidgetItem(str(row[3])))

self.tblTodo.setItem(inx, 4, QTableWidgetItem(str(row[4])))

self.tblTodo.setItem(inx, 5, QTableWidgetItem(str(row[5])))

self.tblTodo.setItem(inx, 6, QTableWidgetItem(str(row[6])))

header = self.tblTodo.horizontalHeader()

header.setSectionResizeMode(0, QtWidgets.QHeaderView.Stretch)

header.setSectionResizeMode(1, QtWidgets.QHeaderView.Stretch)

header.setSectionResizeMode(2, QtWidgets.QHeaderView.Stretch)

header.setSectionResizeMode(3, QtWidgets.QHeaderView.Stretch)

header.setSectionResizeMode(4, QtWidgets.QHeaderView.Stretch)

header.setSectionResizeMode(5, QtWidgets.QHeaderView.Stretch)

header.setSectionResizeMode(6, QtWidgets.QHeaderView.Stretch)

def selectedRow(self):

try:

self.state = "edit"

row = self.tblTodo.currentRow()

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

self.txtSurname.setText(self.tblTodo.item(row, 1).text())

self.txtName.setText(self.tblTodo.item(row, 2).text())

self.txtOtch.setText(self.tblTodo.item(row, 3).text())

self.txtPhone.setText(self.tblTodo.item(row, 5).text())

self.txtPassport.setText(self.tblTodo.item(row, 6).text())

self.btnSave.setEnabled(True)

self.btnDelete.setEnabled(True)

except:

print("error")

class WindowKlient(QMainWindow):

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

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

uic.loadUi('Klient.ui', self)

self.state = "new"

self.con = sqlite3.connect('todo.db')

self.cur = self.con.cursor()

self.btnSave.setEnabled(False)

self.btnDelete.setEnabled(False)

self.btnBack.clicked.connect(self.GoBack)

self.btnNew.clicked.connect(self.AddNew)

self.btnSave.clicked.connect(self.Save)

self.btnDelete.clicked.connect(self.Delete)

self.tblTodo.itemSelectionChanged.connect(self.selectedRow)

self.FillTable()

def GoBack(self):

self.close()

self.Menu = WindowMenu()

self.Menu.show()

def AddNew(self):

self.state = "new"

self.txtSurname.setText("")

self.txtName.setText("")

self.txtOtch.setText("")

self.txtPhone.setText("")

self.btnSave.setEnabled(True)

self.txtSurname.setFocus()

def Save(self):

if (self.state == "new"):

self.cur.execute("insert into klient (Surname, Name, Otch, Phone) values (?, ?, ?, ?)",

(self.txtSurname.text(), self.txtName.text(), self.txtOtch.text(), self.txtPhone.text()))

if (self.state == "edit"):

self.cur.execute("update klient set Surname=?,Name=?,Otch=?, Phone=? where ID\_kl=?", (

self.txtSurname.text(), self.txtName.text(), self.txtOtch.text(), self.txtPhone.text(), self.id))

self.con.commit()

self.FillTable()

self.btnSave.setEnabled(False)

self.btnDelete.setEnabled(False)

def Delete(self):

try:

row = self.tblTodo.currentRow()

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

print(id)

self.cur.execute("delete from klient where ID\_kl=?", (id,))

self.con.commit()

self.FillTable()

self.btnSave.setEnabled(False)

self.btnDelete.setEnabled(False)

except:

print("error")

def FillTable(self):

self.cur.execute("SELECT \* FROM klient")

rows = self.cur.fetchall()

self.tblTodo.setColumnCount(5)

self.tblTodo.setHorizontalHeaderLabels(['id', 'Фамилия', 'Имя', 'Отчество', 'Телефон'])

self.tblTodo.setRowCount(0)

for row in rows:

inx = rows.index(row)

self.tblTodo.insertRow(inx)

# add more if there is more columns in the database.

self.tblTodo.setItem(inx, 0, QTableWidgetItem(str(row[0])))

self.tblTodo.setItem(inx, 1, QTableWidgetItem(str(row[1])))

self.tblTodo.setItem(inx, 2, QTableWidgetItem(str(row[2])))

self.tblTodo.setItem(inx, 3, QTableWidgetItem(str(row[3])))

self.tblTodo.setItem(inx, 4, QTableWidgetItem(str(row[4])))

header = self.tblTodo.horizontalHeader()

header.setSectionResizeMode(0, QtWidgets.QHeaderView.Stretch)

header.setSectionResizeMode(1, QtWidgets.QHeaderView.Stretch)

header.setSectionResizeMode(2, QtWidgets.QHeaderView.Stretch)

header.setSectionResizeMode(3, QtWidgets.QHeaderView.Stretch)

header.setSectionResizeMode(4, QtWidgets.QHeaderView.Stretch)

def selectedRow(self):

try:

self.state = "edit"

row = self.tblTodo.currentRow()

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

self.txtSurname.setText(self.tblTodo.item(row, 1).text())

self.txtName.setText(self.tblTodo.item(row, 2).text())

self.txtOtch.setText(self.tblTodo.item(row, 3).text())

self.txtPhone.setText(self.tblTodo.item(row, 4).text())

self.btnSave.setEnabled(True)

self.btnDelete.setEnabled(True)

except:

print("error")

class WindowMenu(QDialog):

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

super(WindowMenu, self).\_\_init\_\_()

uic.loadUi('Menu.ui', self)

self.btnZak.clicked.connect(self.gotozak)

self.btnKl.clicked.connect(self.gotokl)

self.btnSotr.clicked.connect(self.gotosotr)

def gotozak(self):

self.close()

self.Todo = WindowTodo()

self.Todo.show()

def gotokl(self):

self.close()

self.Kl = WindowKlient()

self.Kl.show()

def gotosotr(self):

self.close()

self.Sotr = WindowSotr()

self.Sotr.show()

class WindowZak(QMainWindow):

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

super(WindowZak, self).\_\_init\_\_()

uic.loadUi('CurZ.ui', self)

self.con = sqlite3.connect('todo.db')

self.cur = self.con.cursor()

self.cur.execute("SELECT `zak`.`ID\_zak`, `klient`.`Name`, `zak`.`title`, `zak`.`todotime`, `zak`.`description`, `zak`.`link` FROM zak INNER JOIN klient ON `zak`.`ID\_k`=`klient`.`ID\_kl` WHERE `zak`.`ID\_zak`=4")

rows = self.cur.fetchone()

self.lbl\_num.setText(str(rows[0]))

self.lbl\_kl.setText(str(rows[1]))

self.lbl\_title.setText(str(rows[2]))

self.lbl\_tododate.setText(str(rows[3]))

self.lbl\_description.setWordWrap(True)

self.lbl\_description.setText(str(rows[4]))

self.lbl\_link.setWordWrap(True)

self.lbl\_link.setText(str(rows[5]))

self.btnback.clicked.connect(self.GoBack)

self.btnDone.clicked.connect(self.zakdone)

def GoBack(self):

self.close()

self.Todo = WindowTodo()

self.Todo.show()

def zakdone(self):

self.cur.execute("update zak set state='Готов'")

self.con.commit()

self.close()

self.Todo = WindowTodo()

self.Todo.show()

app = QApplication(sys.argv)

window = WindowStart()

window.show()

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