Санкт-Петербургское государственное бюджетное профессиональное образовательное учреждение

«Политехнический колледж городского хозяйства»

**ОТЧЕТ**

**Учебная практика**

по профессиональному модулю

**ПМ.02 Осуществление интеграции программных модулей**

|  |  |
| --- | --- |
| Руководитель практики: | |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Солопова Елена Владиславовна  ФИО |
|  |  |
| Отчет выполнил студент группы: ИП-21-3 | |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Пономарев Андрей Викторович |
|  | ФИО |

2025

**Содержание**

[1. Введение 3](#_Toc190262597)

[2. Общая часть 4](#_Toc190262598)

[2.1. Разработка базы данных, объектов БД и импорт данных 4](#_Toc190262599)

[2.1.1. Схема базы данных 16](#_Toc190262600)

[2.2. Разработка модулей программного обеспечения для компьютерных систем 16](#_Toc190262601)

[2.2.1. Алгоритм решения задачи 18](#_Toc190262602)

[2.2.3. Тестирование приложения 31](#_Toc190262603)

[2.3. Сопровождение и обслуживание программного обеспечения компьютерных систем 34](#_Toc190262604)

[2.3.1. Программный модуль для работы с заказами 34](#_Toc190262605)

[2.3.2. Осуществление интеграции программных модулей 37](#_Toc190262606)

[2.3.3. Разработка библиотеки и подключение её к проекту 41](#_Toc190262607)

[2.4. Представление результатов работы 45](#_Toc190262608)

[3. Заключение 48](#_Toc190262609)

[Приложение А 49](#_Toc190262610)

# 1. Введение

ООО «Игора» - Прокат спортивного инвентаря.

В рамках выполнения задания необходимо разработать основные модули информационной системы для точки проката горнолыжного курорта «Игора» в Ленинградской области:

* неавторизованный клиент и авторизованный клиент может просматривать предоставляемые услуги и формировать заказы;
* менеджер может просматривать услуги, добавлять и редактировать клиентов, подтверждать и редактировать заказы;
* администратор может просматривать/добавлять/редактировать/удалять услуги.

Кроме того, разрабатываемая информационная система предполагает установку на терминалах в точках прокатах. На терминале клиент (авторизованный и неавторизованный) может просмотреть услуги и сформировать заказ.

# 2. Общая часть

## 2.1. Разработка базы данных, объектов БД и импорт данных

СУБД для создания базы данных: SQL Server Management studio.

Логин для входа: 975-21;

Пароль для входа: 7snhf12K;

Пошаговая инструкция запуска и просмотра БД:

1. Запустить SSMS;
2. Войти используя данные, указанные выше;
3. Раскрыть пункт «gogs.wsr.ru»;
4. Раскрыть пункт «Базы данных»;
5. Найти базу данных с названием IgoraDemo и раскрыть;
6. Раскрыть пункт «Таблицы»;
7. Выбрать интересующую таблицу, нажать ПКМ по ней и выбрать пункт «Выбрать первые 1000 строк».

Пошаговая инструкция импорта данных БД:

Для корректного переноса в базу данных необходимо преобразовать входные данные в третью нормальную форму.

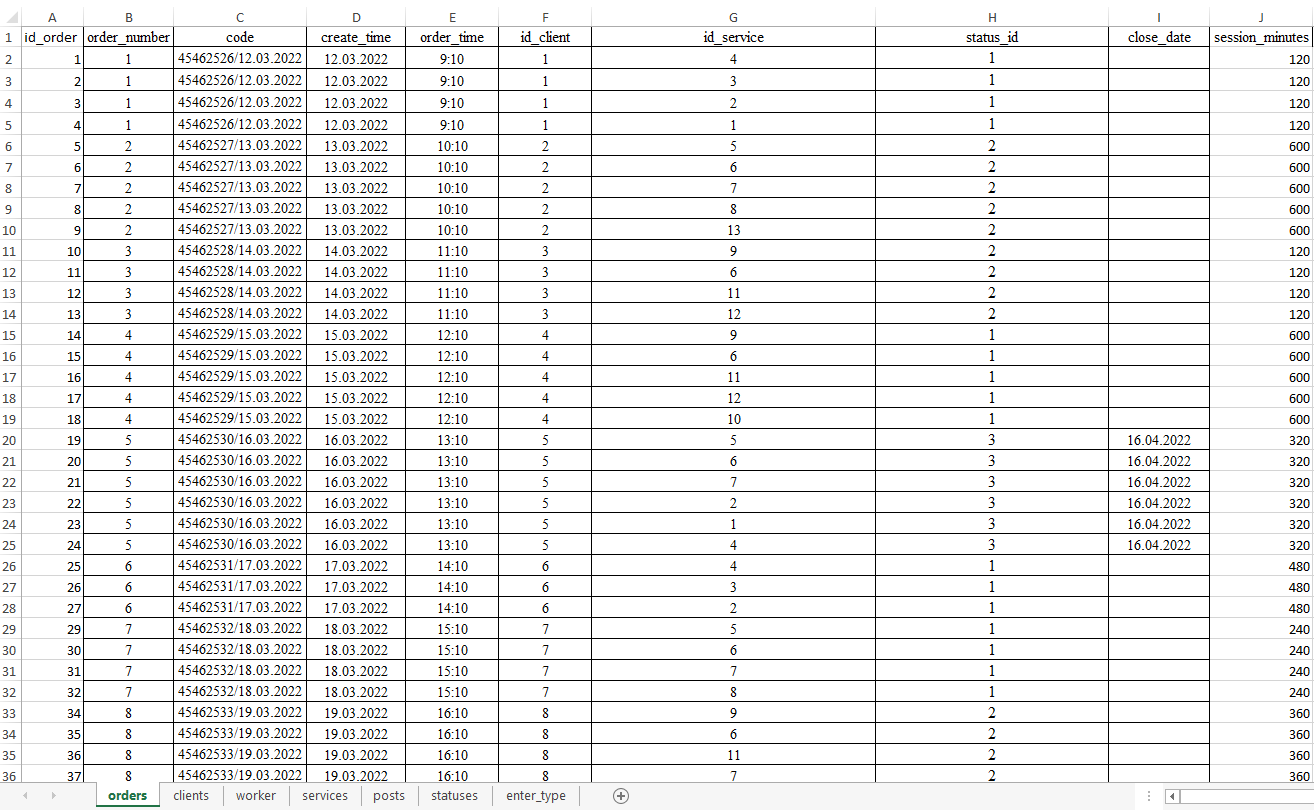


Рисунок 1. Таблица заказы для импорта

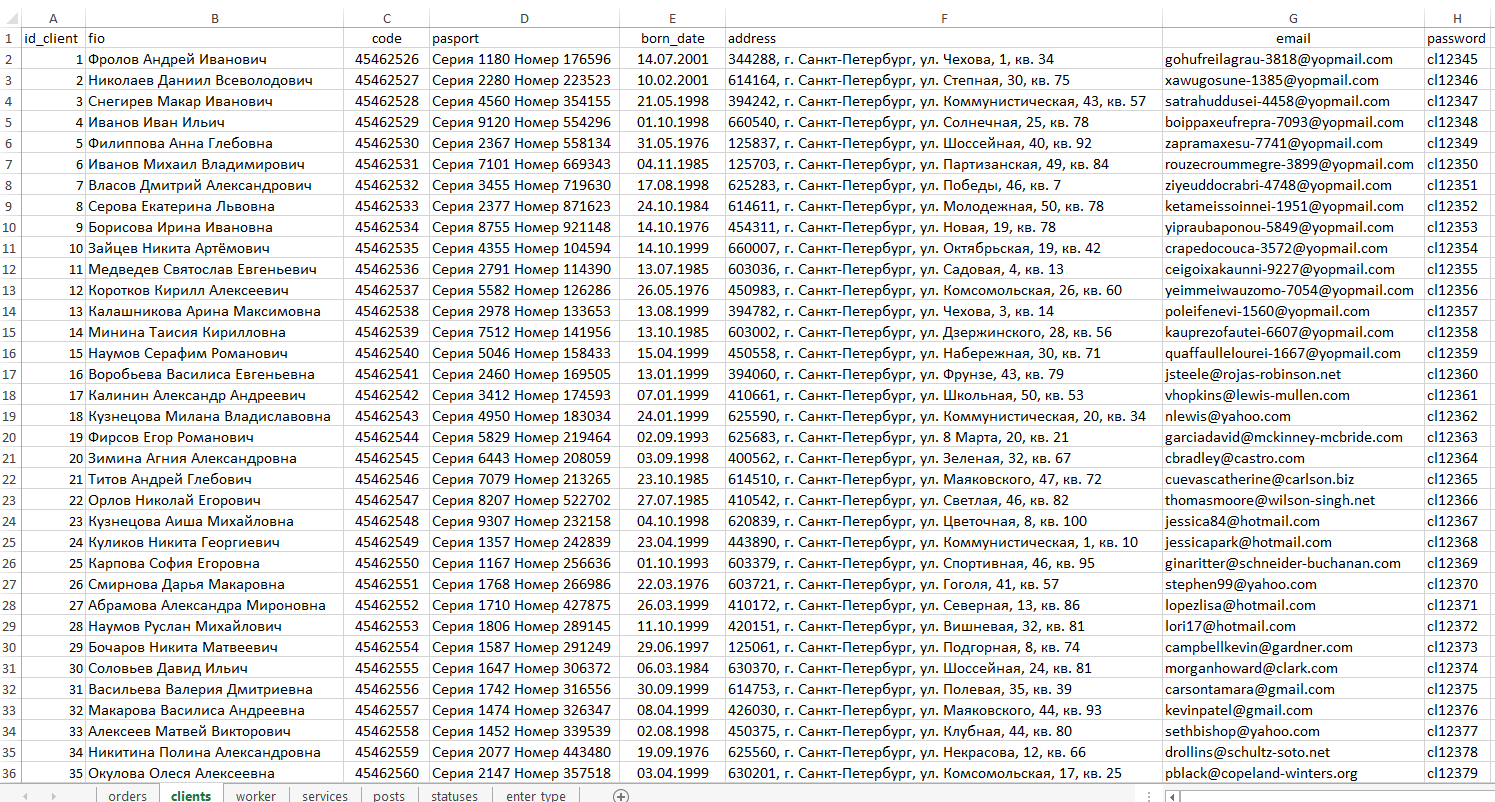


Рисунок 2. Таблица клиенты для импорта

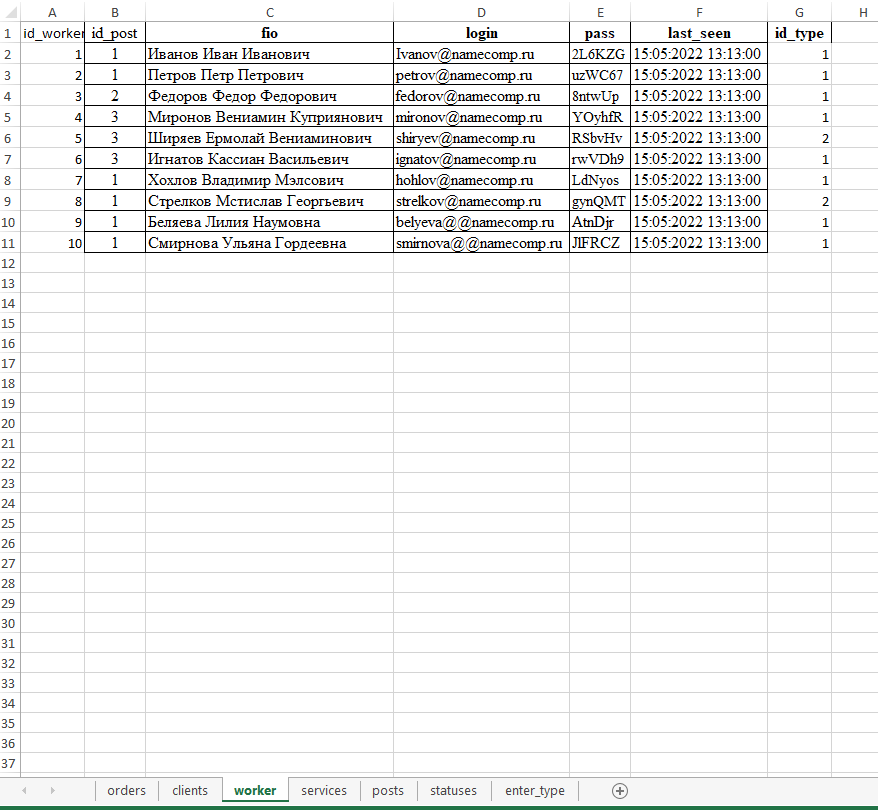


Рисунок 3. Таблица работники для импорта

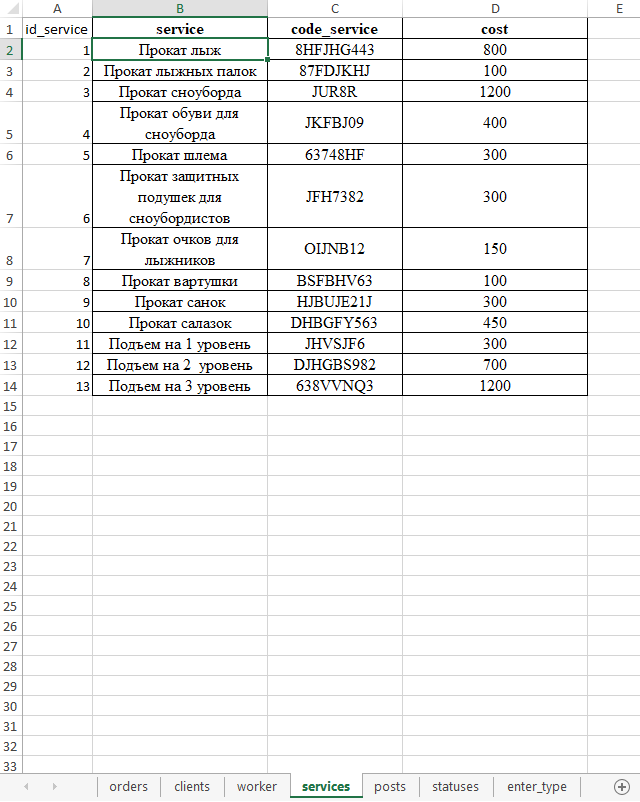


Рисунок 4. Таблица услуги для импорта

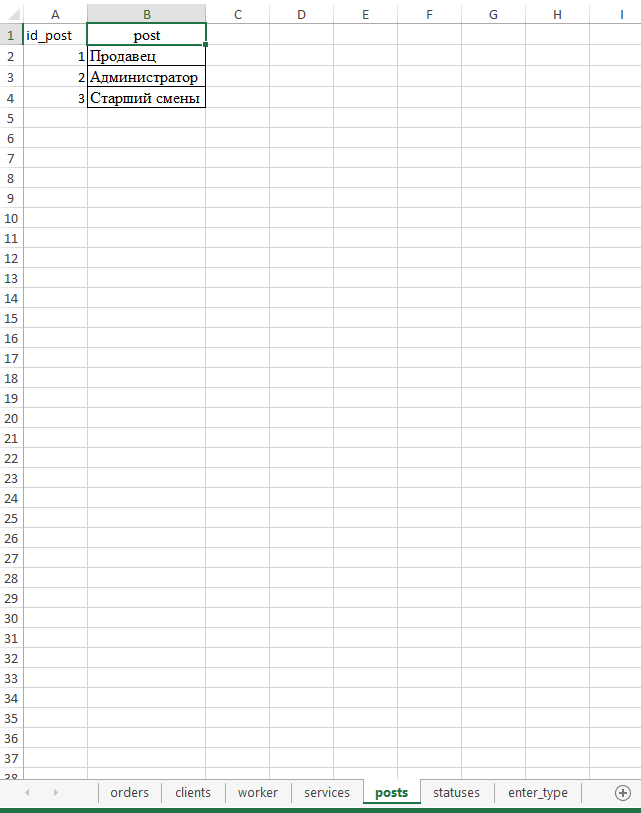


Рисунок 5. Таблица должности для импорта

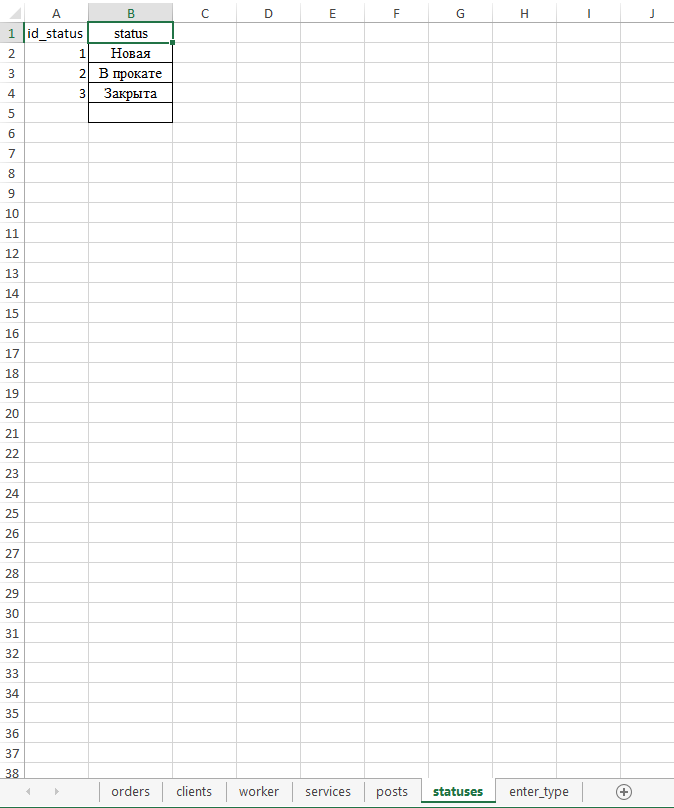


Рисунок 6. Таблица статусы для импорта



Рисунок 7. Таблица тип входа для импорта

Когда таблицы приведены к нужному виду необходимо создать базу данных в Microsoft management studio. Название базы данных соответствует предметной области данных:

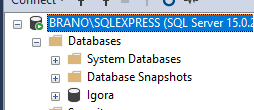


Рисунок 8. Новая база данных

Далее необходимо нажать «Задачи» -> «Импорт данных»:

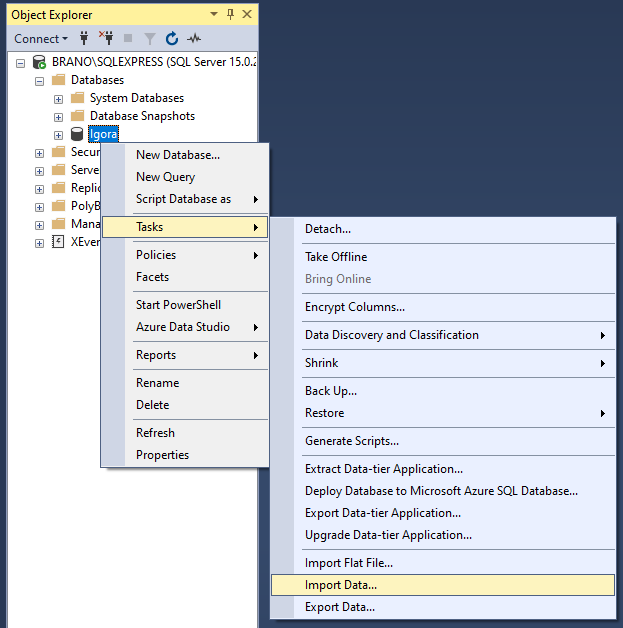


Рисунок 9. Импорт данных

Далее выбрать файл excel для импорта:

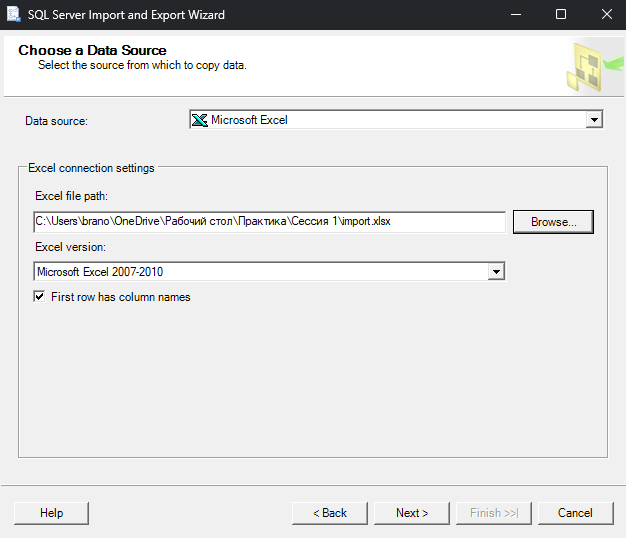


Рисунок 10. Импорт данных из excel

Нажать «Далее», выбрать откуда с помощью чего будут переносится данные:

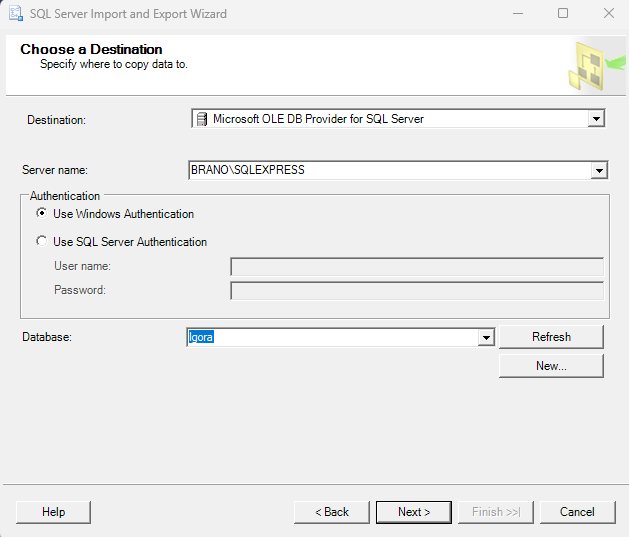


Рисунок 11. Импорт данных

Нажать «Далее»:

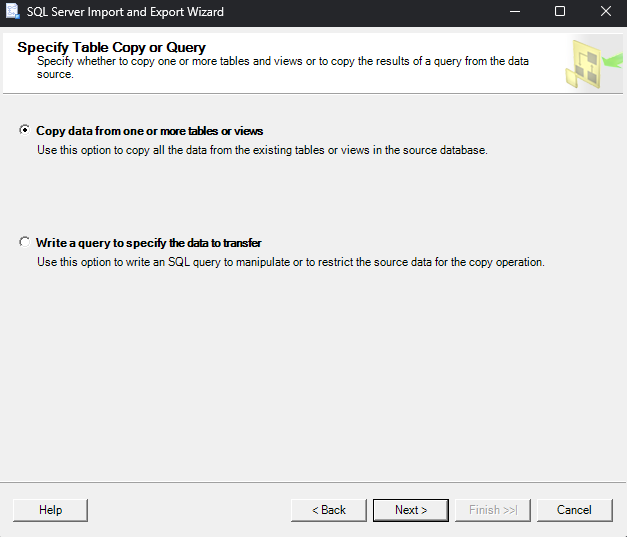


Рисунок 12. Импорт данных из таблиц

Выбрать все таблицы для переноса:

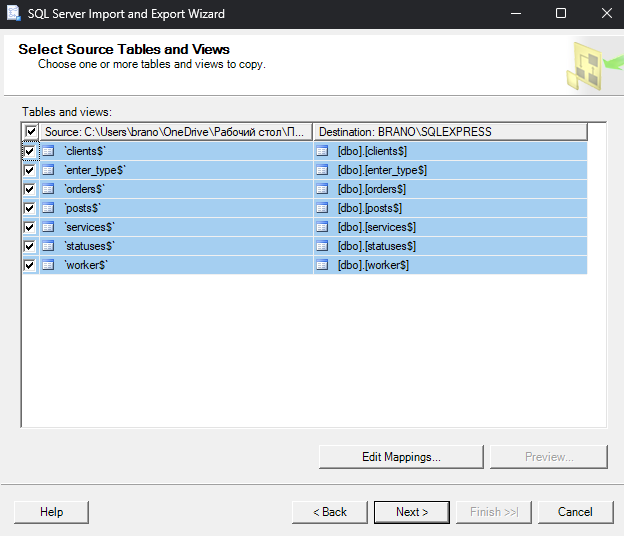


Рисунок 13. Импорт таблиц

И завершить импорт данных в базу данных.

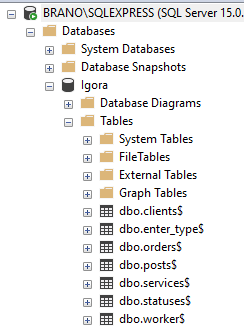


Рисунок 14. Импортированная база данных

Пошаговая инструкция экспорта данных БД:

1. Выполнить выход, найти нужную БД;
2. Нажать ПКМ по названию БД;
3. Нажать «Задачи»;
4. Нажать «Сгенерировать скрипты»;
5. Нажать «Далее», затем «Далее»;
6. В настройках выбрать «Сохранить как скрипт»;
7. Нажать «Продвинутые», там найти пункт «Типы данных скрипт» - выбрать «Данные и таблицы», нажать «ОК»;
8. Выбрать место куда сохранится скрипт;
9. Завершить импорт.

## 2.1.1. Схема базы данных

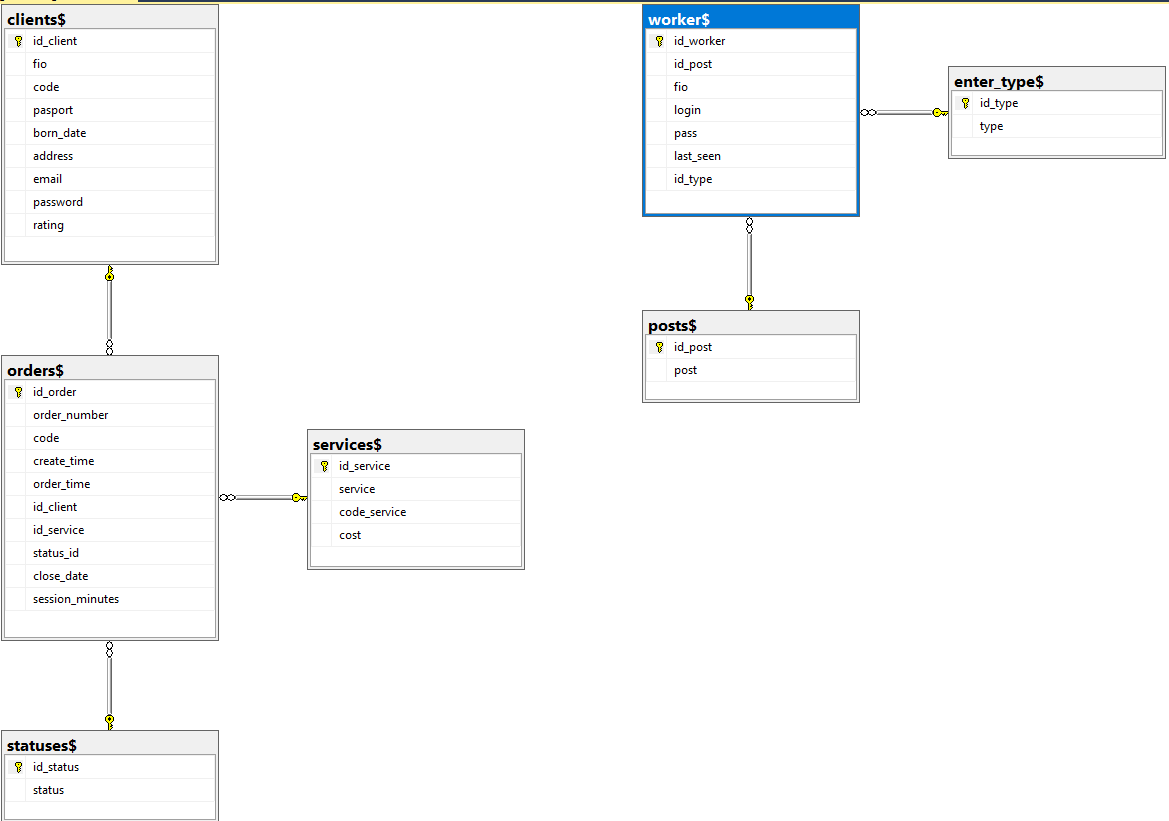


Рисунок 15. Схема базы данных

## 2.2. Разработка модулей программного обеспечения для компьютерных систем

Среда разработки: Microsoft Visual Studio, C# .NET Framework 4.8, WinForms;

Данные для входа как клиент: логин – client; пароль – client.

Данные для входа как менеджер: логин – manager; пароль – manager;

Данные для входа как администратор: логин – admin; пароль – admin;

Запуск приложения:

1. Зайти в папку с проектом.
2. Открыть папку IgoraDemo.
3. Открыть папку bin.
4. Открыть папку Debug.
5. Запустить приложение «IgoraDemo.exe»

## 2.2.1. Алгоритм решения задачи

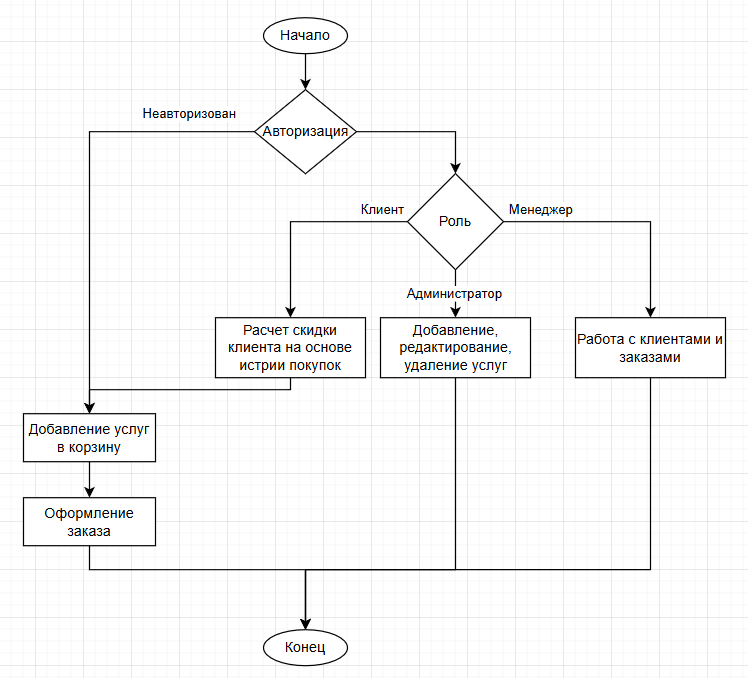
****

Рисунок 16. Алгоритм работы программы

Функция расчета индивидуальной скидки:

public static int calculateDiscount(int sum)

{

if (sum < 50\_000)

{

return 0;

}

else if (sum >= 50\_000 && sum < 100\_000)

{

return 1;

}

else if (sum >= 100\_000 && sum < 300\_000)

{

return 3;

}

else

{

return 5;

}

}

На вход подается количество потраченных денег клиентом, возвращается скидка величиной от 0% до 5% в виде целочисленного значения.

Скидка вычисляется в зависимости от потраченных денег: до 50000 – 0%, от 50000 – до 100000 – 1%, от 100000 – до 300000 – 3%, более 300000 – 5%.

**2.2.2. Разработка desktop-приложения**

Ссылка на репозиторий - <http://gogs.wsr.ru:3000/975-21/Ponomarev02Igora>.

**Функциональность:** Главная форма открывается при запуске приложения. На ней пользователь может просмотреть услуги и сформировать заказ. Также возможен поиск по названию услуги и сортировка по цене. После добавления в корзину появляется кнопка «Корзина».

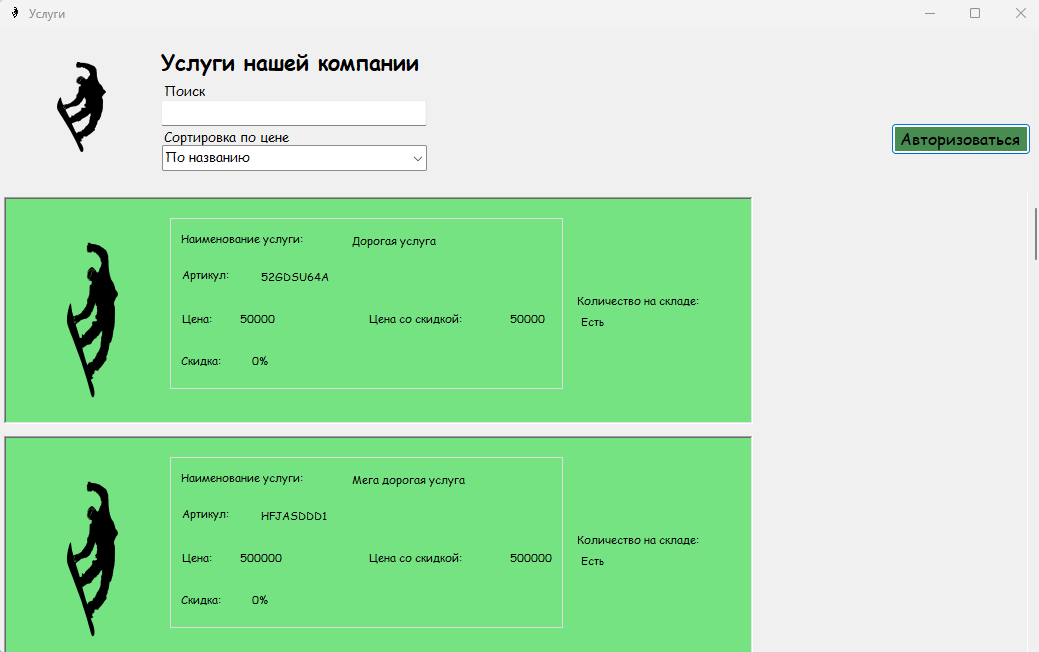
****

Рисунок 17. Главная форма

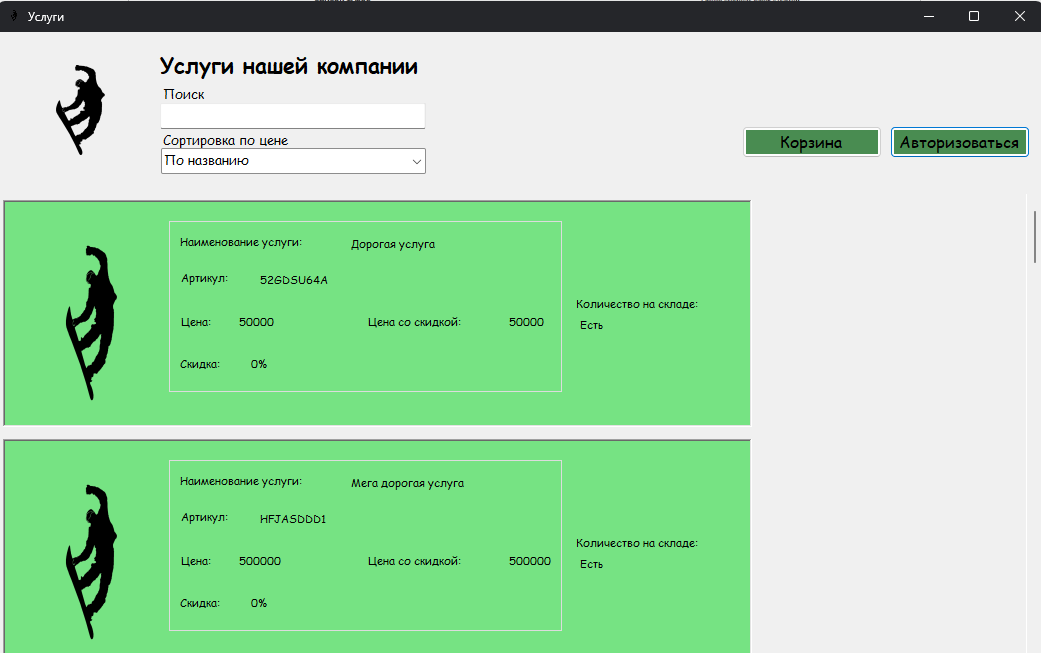
****

Рисунок 18. Главная форма с корзиной

**Функциональность:** В корзине отображаются все услуги, которые были добавлены в корзину, а также общая скидка и сумма заказа. По нажатии кнопки «Оформить заказ» можно оформить заказ и данные о нем занесутся в базу данных

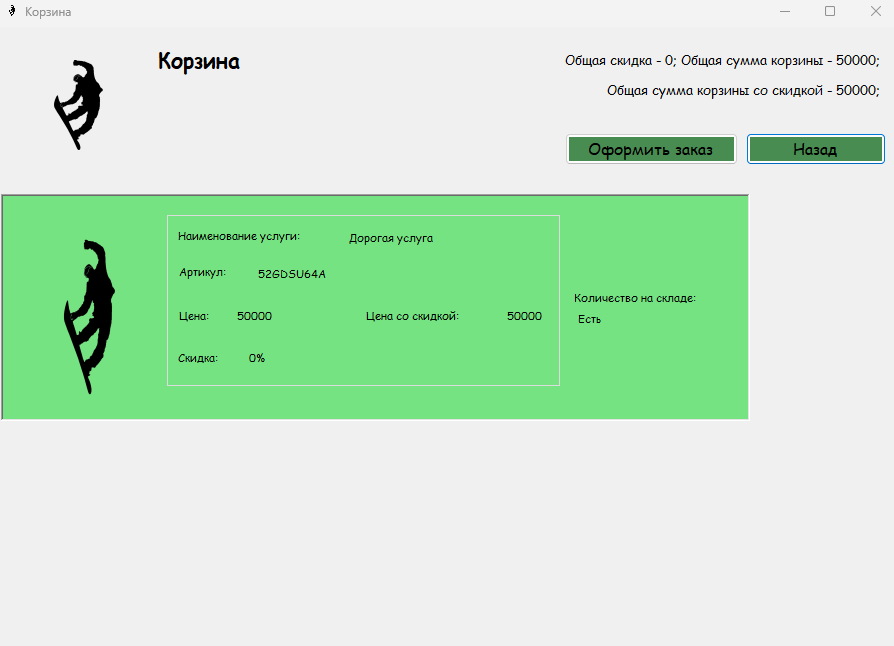
****

Рисунок 19. Форма корзины

**Функциональность:** Окно авторизации позволяет заполнить данные для входа.

Данные для входа **клиента**: логин - client; пароль - client;

Данные для входа **менеджера**: логин - manager; пароль - manager;

Данные для входа **администратора**: логин – admin; пароль – admin;

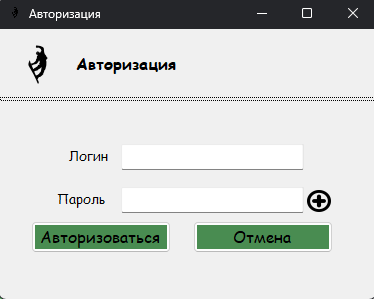


Рисунок 20. Форма авторизации

**Функциональность:** При входе как клиент, на всех услугах появляется персональная скидка и расчетная цена, а также кнопка «История»:

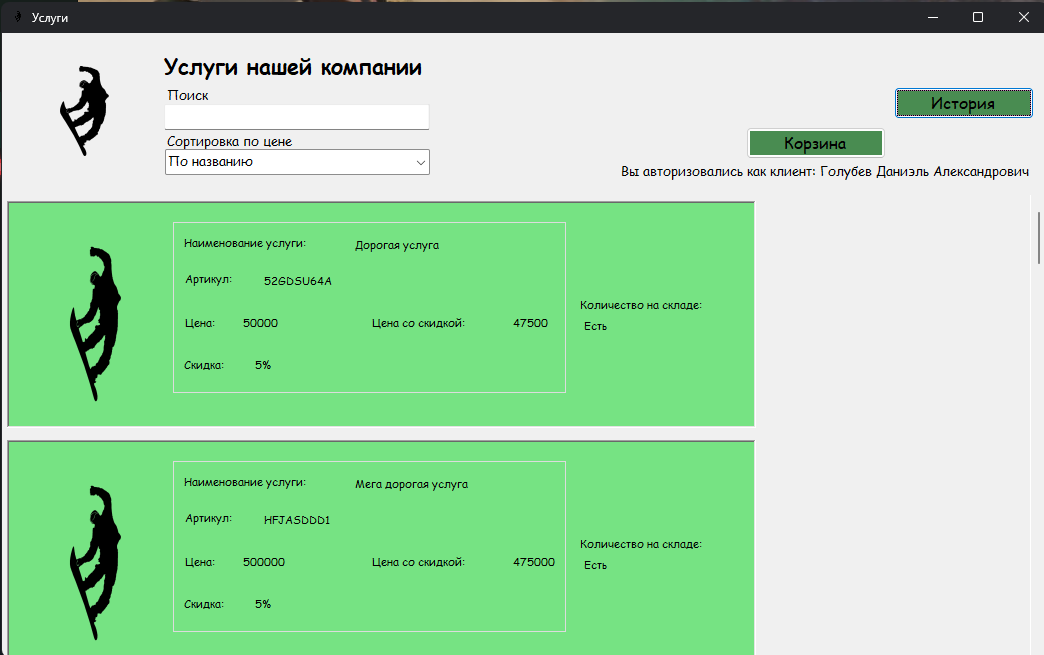


Рисунок 21. Главная форма для авторизованного пользователя

**Функциональность:** На форме отображения истории авторизованного клиента можно посмотреть все заказы (ФИО, рейтинг, дата оформления, сумма заказа, статус). Также есть возможность выбрать заказ и отменить его (если не была внесена предоплата или заказ не был завершен).

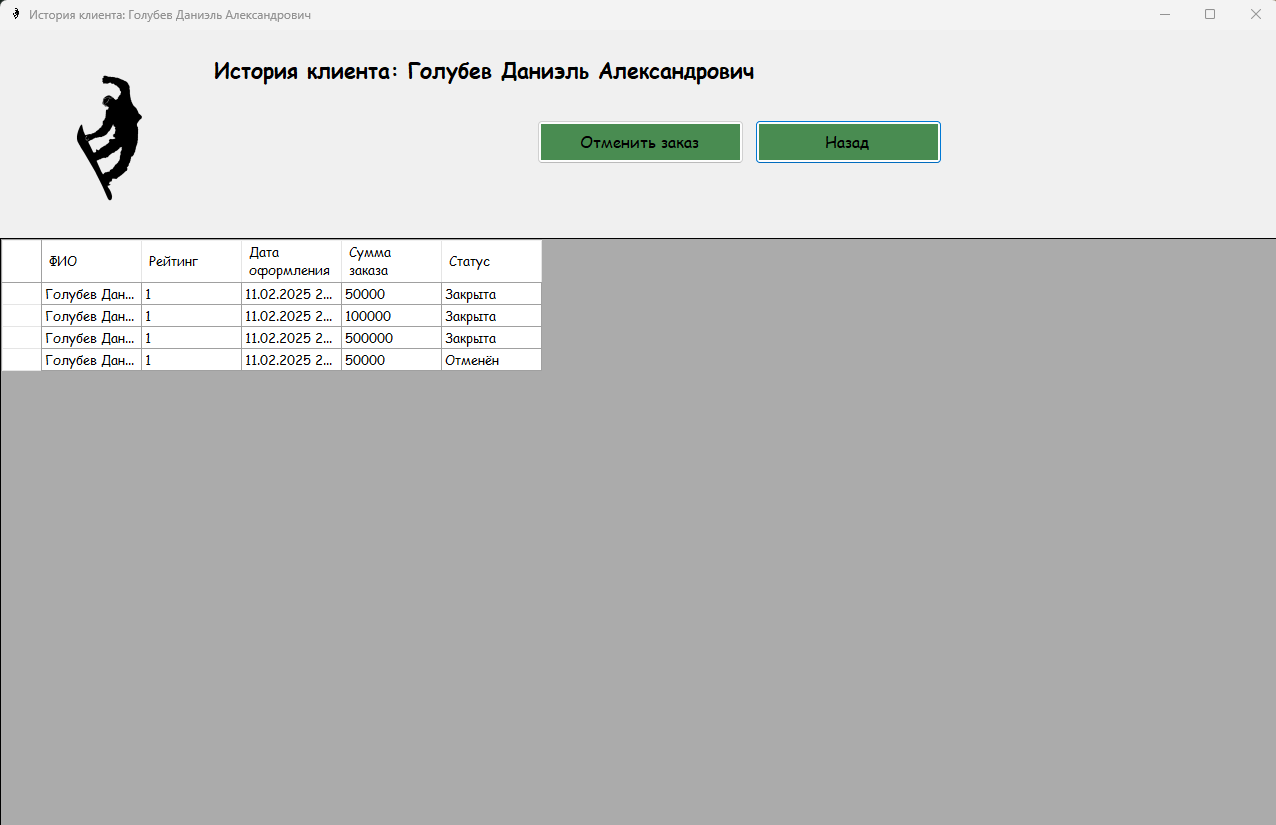


Рисунок 22. История авторизованного пользователя

**Функциональность:** Корзина также меняется для авторизованного пользователя: добавляется персональная скидка, а заказ будет оформляться от лица авторизованного клиента.

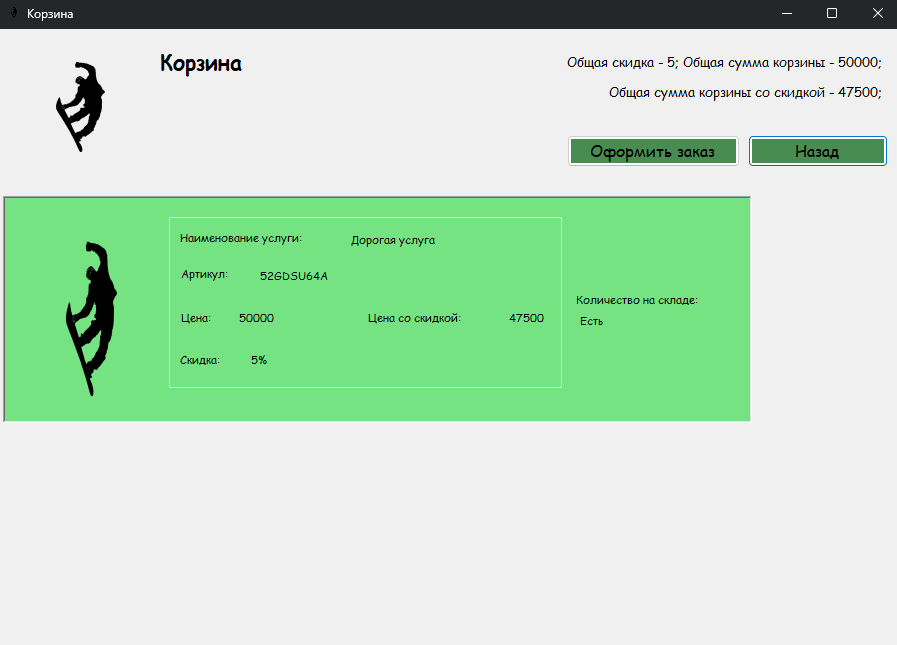


Рисунок 23. Корзина авторизованного пользователя

**Функциональность:** При авторизации как менеджер на главную форму добавляются кнопки «Заказы» и «Клиенты». Строка состояния меняется соответственно.

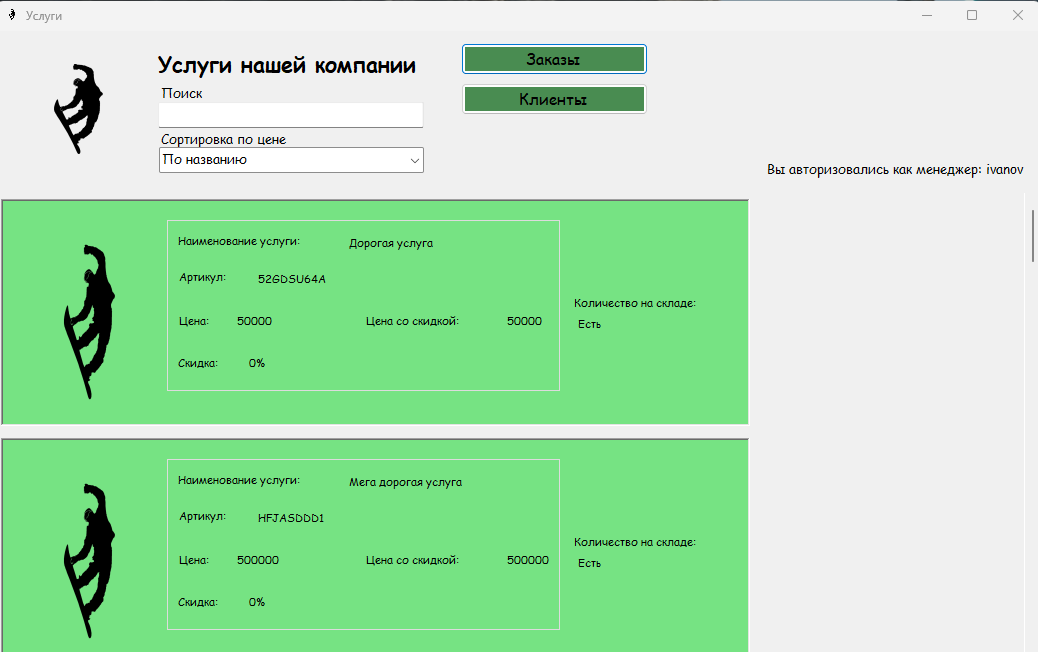


Рисунок 24. Главная форма менеджера

**Функциональность:** Окно просмотра заказов отображает все заказы (ФИО, рейтинг, дата оформления, сумма заказа, статус). Позволяет сделать поиск по ФИО. Кнопка «Назад» обеспечивает целостность опыта использования программы. При нажатии на кнопку «Подтвердить заказ» Статус будет меняться последовательно: Новая -> В прокате -> Закрыта. Закрытый заказ нельзя подтвердить.

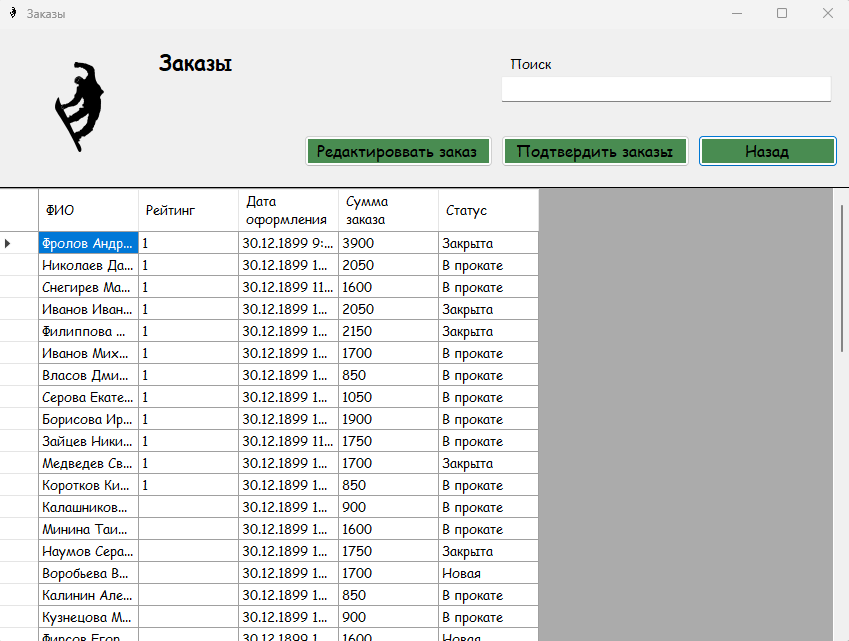


Рисунок 25. Форма просмотра заказов для менеджера

**Функциональность:** При нажатии на кнопку «Редактировать заказ» появляется форма редактирования выбранного заказа. Пользователь может изменить значения прям в таблице в удобном формате, сделать поиск по ключевым словам, сохранить изменения или откатить их.

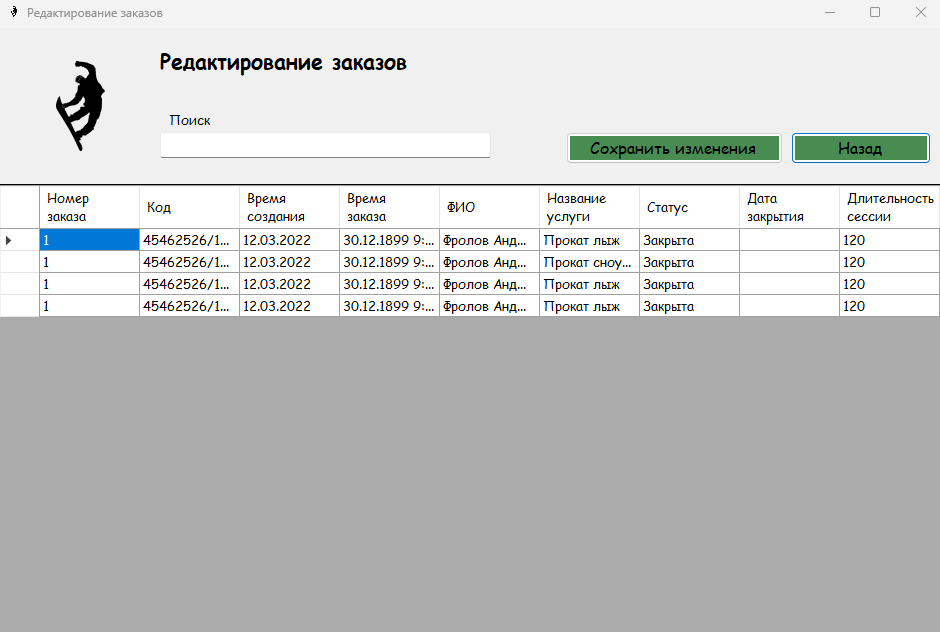


Рисунок 26. Форма редактирования заказов для менеджера

**Функциональность:** Форма просмотра и редактирования клиентов отображает всех клиентов компании (ФИО, код, паспорт, дата рождения, адрес, почта, пароль, рейтинг). Возможно сделать поиск по ФИО. Кнопка «Назад» обеспечивает целостность опыта использования приложения.

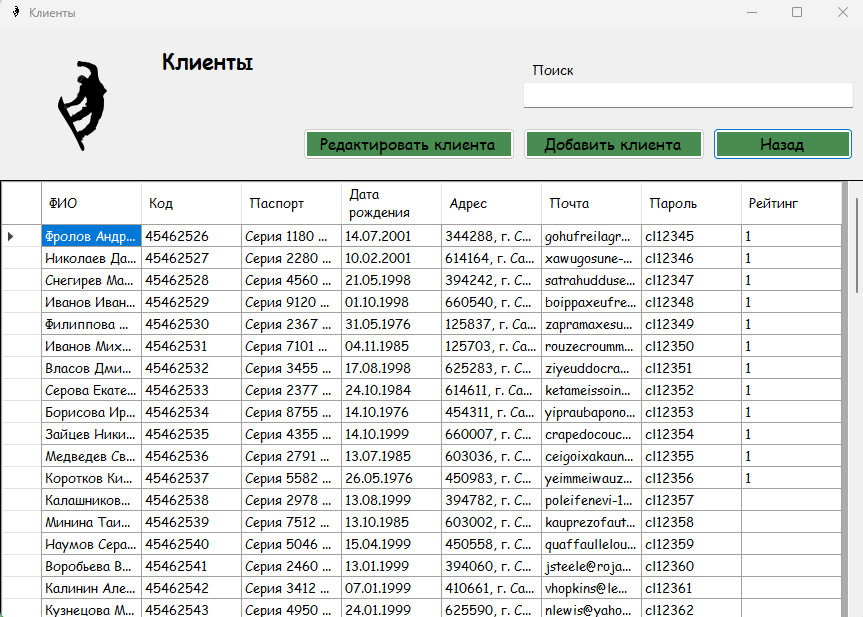


Рисунок 27. Форма просмотра клиентов для менеджера

**Функциональность:** Форма редактирования клиента позволяет изменить все данные о клиенте с валидацией данных для предотвращения ошибок.

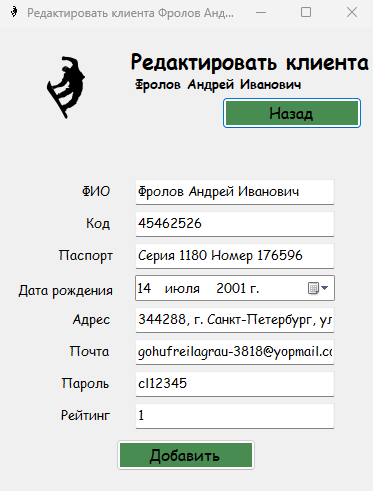


Рисунок 28. Форма редактирования клиента

**Функциональность:** Добавление клиента позволяет ввести данные нового клиента для добавления его в систему. Валидация данных не даст совершить ошибку при пользовательском вводе.

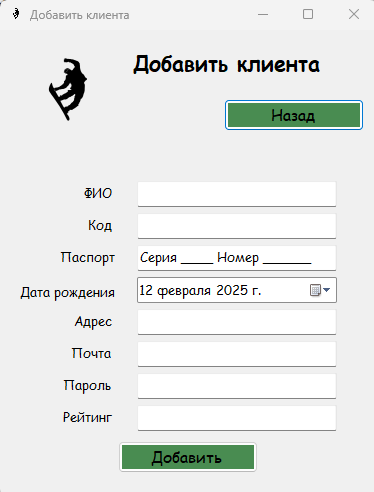


Рисунок 29. Форма добавления клиента для менеджера

**Функциональность:** При авторизации как администратор появляется панель администратора, позволяющая просмотреть все услуги, выполнить поиск и сортировку, и взаимодействовать с ними (добавить, редактировать и удалить). Чтобы удалить строку необходимо выделить нужную строчку и нажать «Удалить строку».

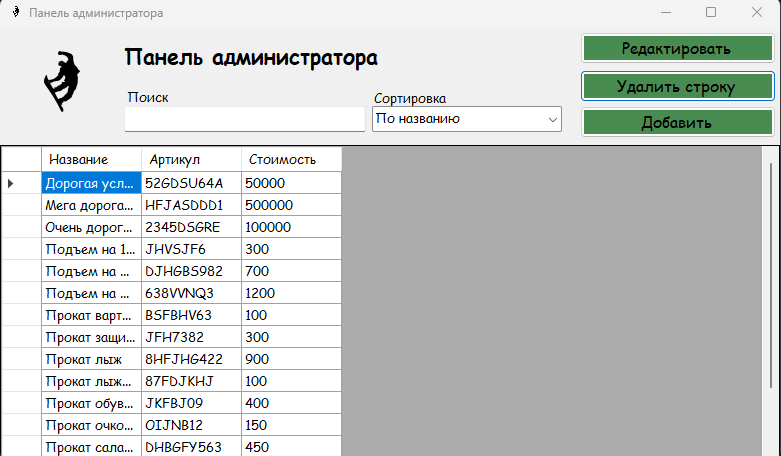


Рисунок 30. Главная форма для администратора

**Функциональность:** Форма редактирования услуги, поддерживающая валидацию данных:

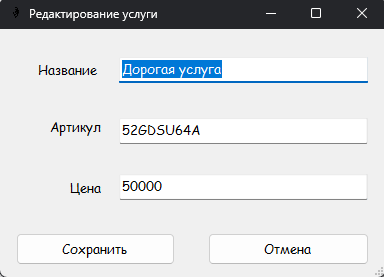


Рисунок 31. Редактирование услуги для администратора

**Функциональность:** Форма добавления услуги с валидацией данных. Невозможно добавить услугу с одинаковым артикулом.

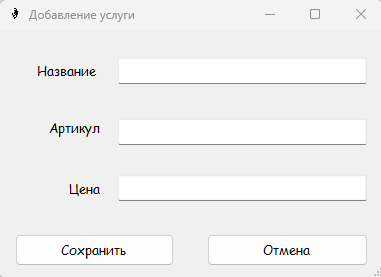


Рисунок 32. Форма добавления услуги для администратора

## 2.2.3. Тестирование приложения

Тестирование валидации данных создания нового клиента:

**Тест №1:** Добавление пользователя проходит успешно, без потери данных и с валидацией пользовательского ввода.

Входные данные:

* ФИО – qwe qwe qwe
* Код - 12312312
* Паспорт – 1212121212
* Дата рождения – текущая дата
* Адрес – йцу
* Почта – qwe
* Пароль – qwe123QWE123
* Рейтинг - 5

Ожидаемый результат: сообщение о неверном формате введённой почты без аварийного завершения.

Фактический результат: сообщение о неверном формате введённой почты без аварийного завершения.

Результат: Тест пройден.

**Тест №2:** Добавление пользователя проходит успешно, без потери данных и с валидацией пользовательского ввода.

Входные данные:

* ФИО – qwe
* Код - 12312312
* Паспорт – 1212121212
* Дата рождения – текущая дата
* Адрес – йцу
* Почта – qwe@bk.ru
* Пароль – qwe123QWE123
* Рейтинг - 5

Ожидаемый результат: сообщение о неверном формате введённого ФИО без аварийного завершения.

Фактический результат: сообщение о неверном формате введённого ФИО без аварийного завершения.

Результат: Тест пройден.

**Тест №3:** Добавление пользователя не проходит если в БД уже есть такой клиент.

Входные данные:

* ФИО – qwe qwe qwe
* Код - 12312312
* Паспорт – 1212121212
* Дата рождения – текущая дата
* Адрес – йцу
* Почта – qwe@bk.ru
* Пароль – qwe123QWE123
* Рейтинг – 5

Примечание: в базе данных уже должна быть запись с кодом 12312312 и паспортом 1212121212.

Ожидаемый результат: сообщение о невозможности повтора одинаковых клиентов.

Фактический результат: сообщение о невозможности повтора одинаковых клиентов.

Результат: Тест пройден.

**Тест №4:** Добавление услуги не проходит, если указанный артикул существует в БД.

Входные данные:

* Название – Прокат красного сноуборда
* Артикул – SHDDT123A
* Цена - 300

Примечание: в базе данных уже должна быть запись с артикулом SHDDT123A.

Ожидаемый результат: сообщение о невозможности повтора одинаковых артикулов.

Фактический результат: сообщение о невозможности повтора одинаковых артикулов.

Результат: Тест пройден.

Тестирование подсистемы расчета скидки:

**Тест №1:** Скидка для клиента с суммой выкупа 40000 рублей

Входные данные: Сумма выкупа 40000.

Ожидаемый результат: функция вернёт 0.

Фактический результат: функция вернула 0.

Результат: Тест пройден.

**Тест №2:** Скидка для клиента с суммой выкупа 51000 рублей

Входные данные: Сумма выкупа 51000.

Ожидаемый результат: функция вернёт 1.

Фактический результат: функция вернула 1.

Результат: Тест пройден.

**Тест №3:** Скидка для клиента с суммой выкупа 150000 рублей

Входные данные: Сумма выкупа 150000.

Ожидаемый результат: функция вернёт 3.

Фактический результат: функция вернула 3.

Результат: Тест пройден.

**Тест №4:** Скидка для клиента с суммой выкупа 200000 рублей

Входные данные: Сумма выкупа 200000.

Ожидаемый результат: функция вернёт 5.

Фактический результат: функция вернула 5.

Результат: Тест пройден.

# 2.3. Сопровождение и обслуживание программного обеспечения компьютерных систем

## 2.3.1. Программный модуль для работы с заказами

Возможность работы с заказами представляет форма «История». На главной форме необходимо авторизоваться как клиент и отобразится кнопка для перехода на соответствующую форму.

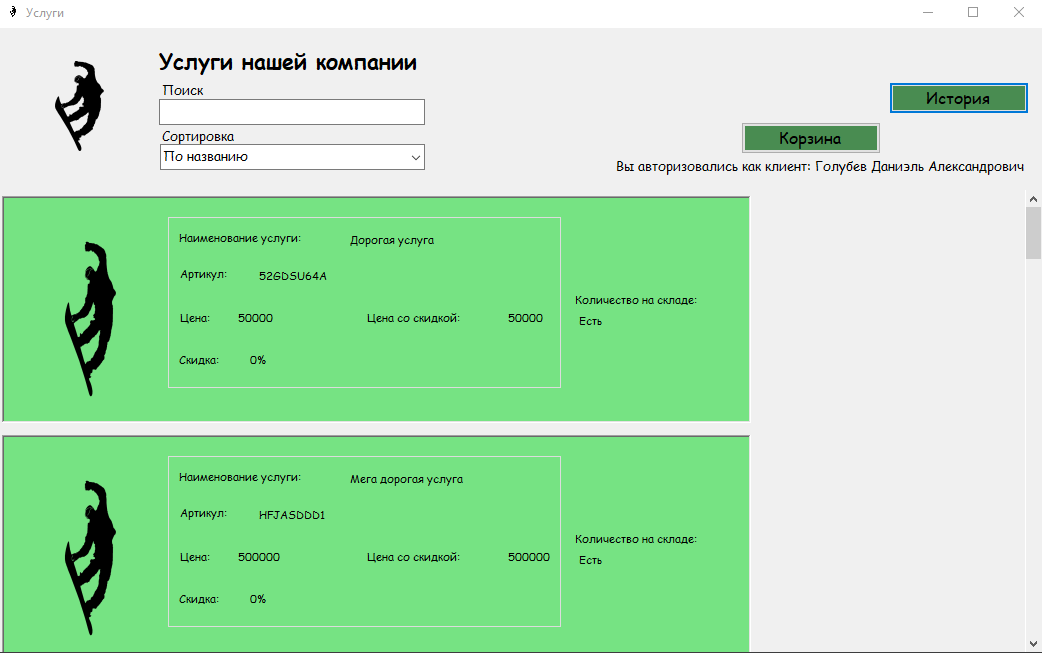


Рисунок 33. Главная форма с историей

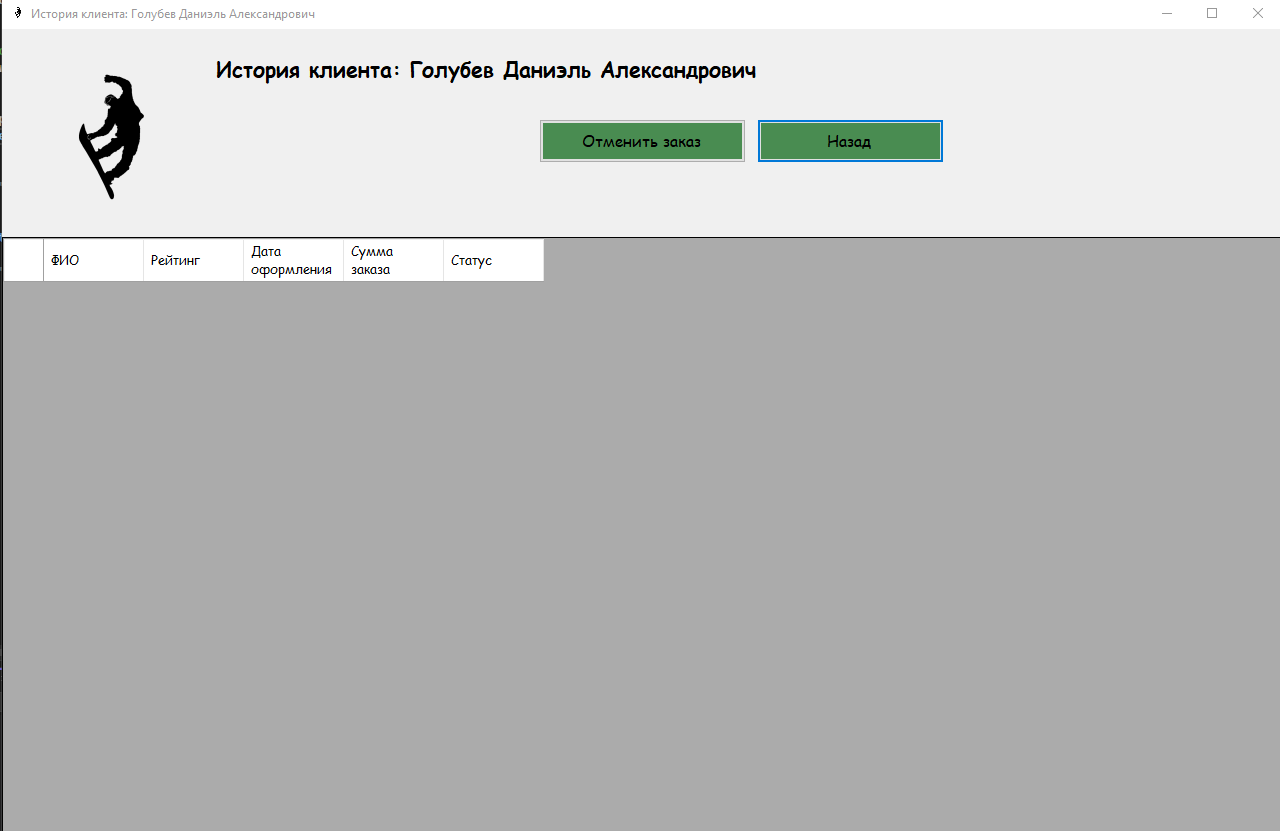


Рисунок 34. Форма история

Чтобы отобразились заказы необходимо оформить заказ.

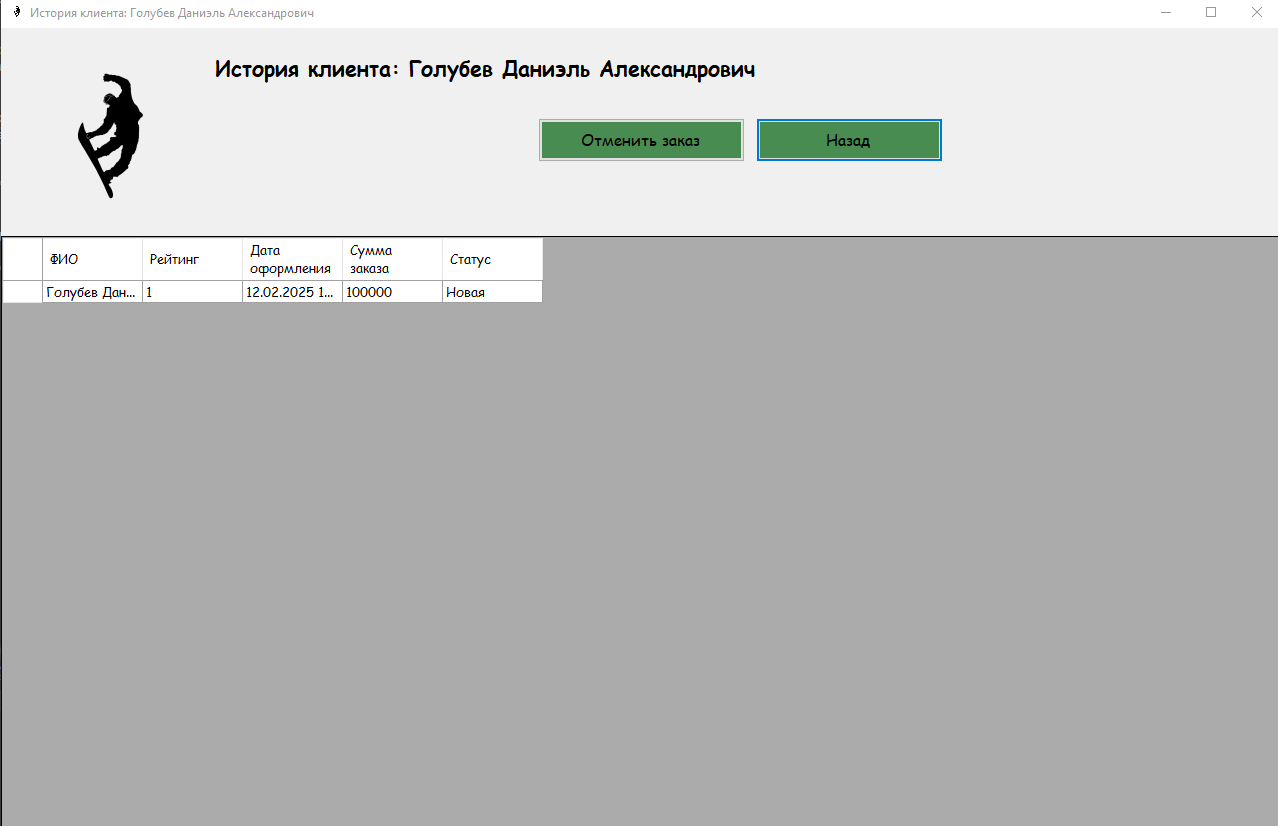


Рисунок 35. История с заказом

Комплекс предоставляет возможность отменить заказ, если он не был оплачен и не был закончен. Для этого необходимо выделить заказ и нажать «Отменить заказ». Его статус изменится на «Отменен».

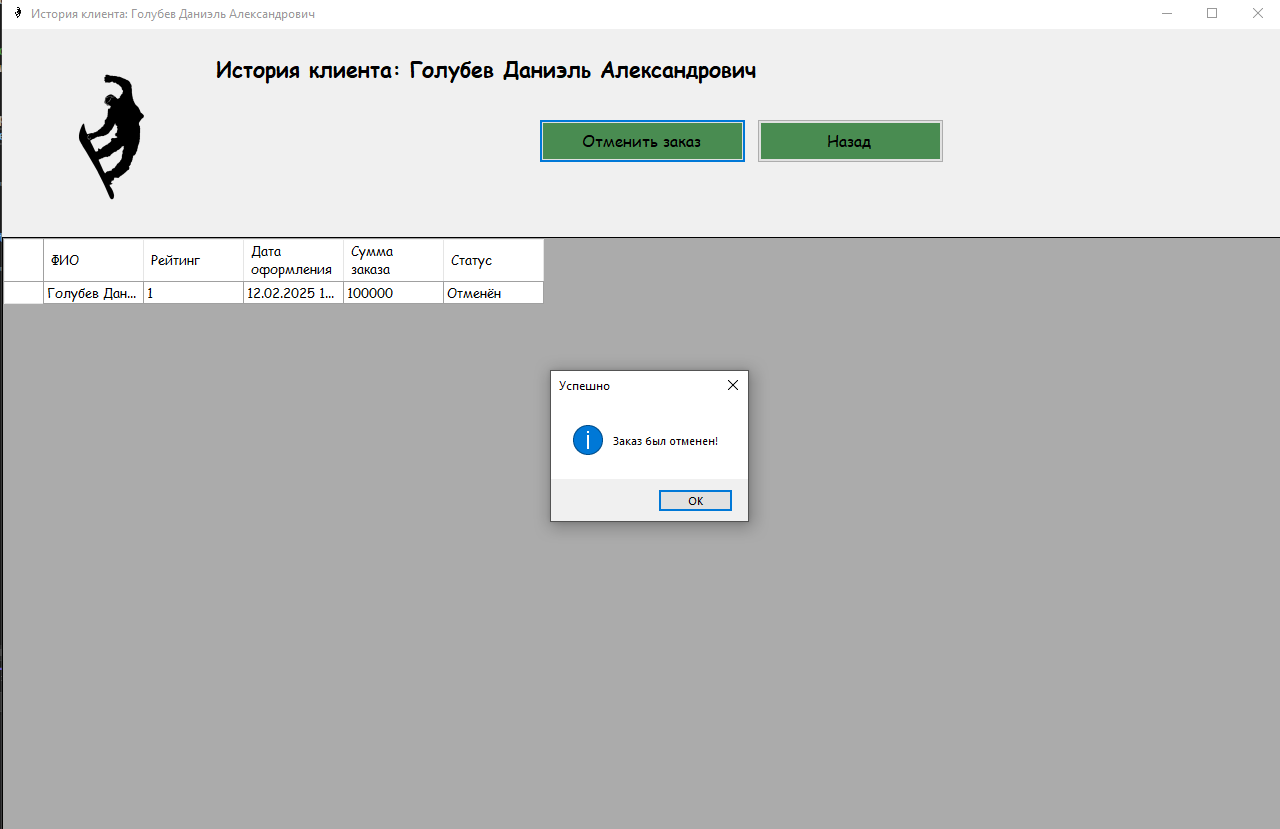


Рисунок 36. Отменённый заказ в истории

При оформлении нового заказа он отобразится после отменённого.

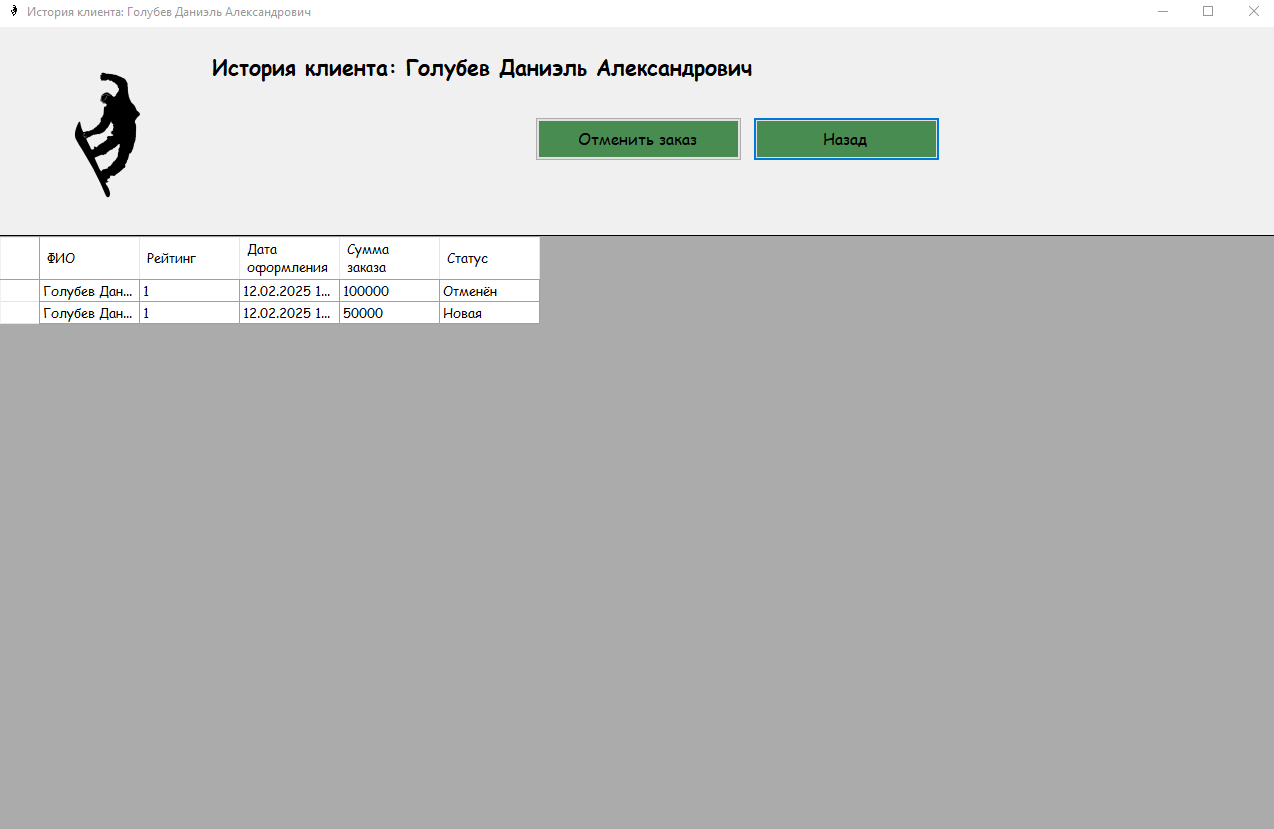


Рисунок 37. Новый заказ

## 2.3.2. Осуществление интеграции программных модулей

Работа с заказами предполагает две стороны взаимодействия: клиент и менеджер.

**Клиент** может добавлять услуги в корзину и формировать заказ, а также отменить его до подтверждения менеджером.

Клиент нажимает ПКМ по понравившейся услуге и нажимает «Добавить в корзину».

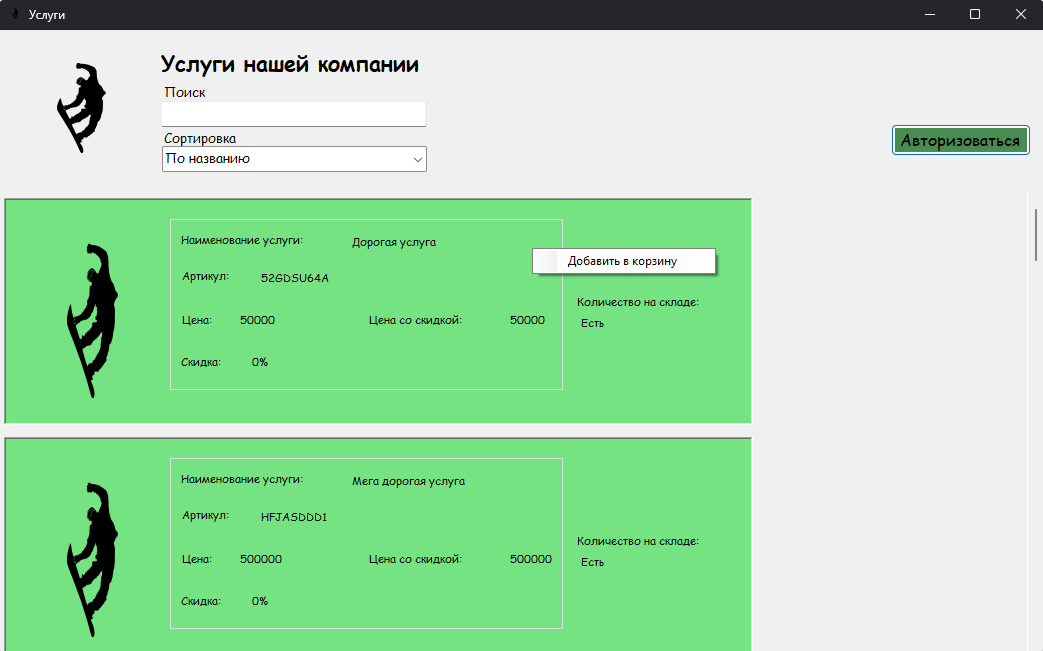


Рисунок 38. Добавление услуги в корзину

Появляется кнопка «Корзина» в которой хранятся выбранные клиентом товары, где он может сделать свой заказ. Отмена заказа может быть выполнена только авторизованным пользователем в форме «История».

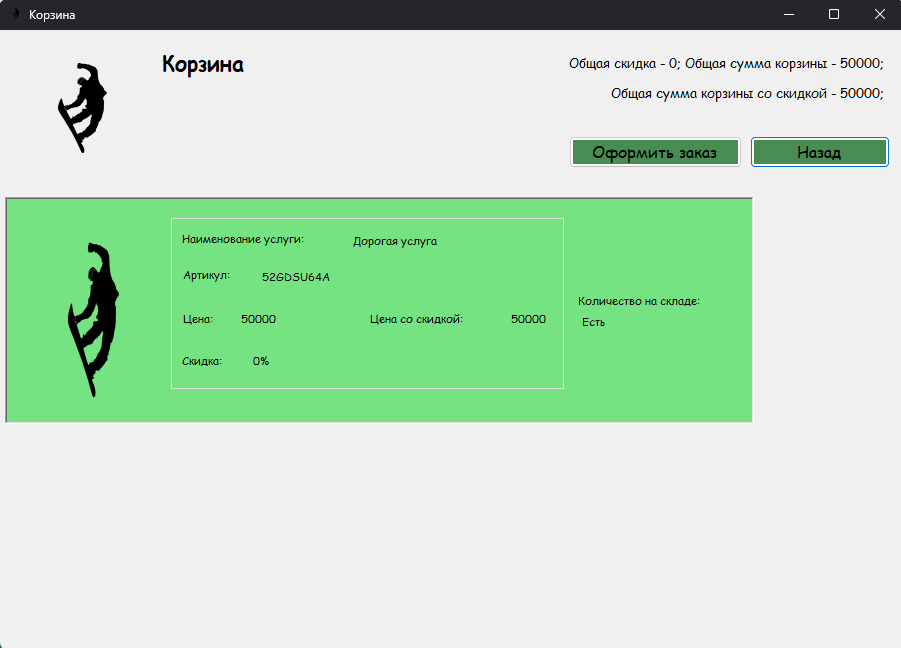


Рисунок 39. Товар в корзине

Код отвечающий за добавление товаров в корзину:

private void добавитьВКорзинуToolStripMenuItem\_Click(object sender, EventArgs e)

{

MainForm.cartOrders.Add(\_service);

}

Код отвечающий за передачу коллекции заказов в форму корзины и подсчета скидки:

private void CartBtn\_Click(object sender, System.EventArgs e)

{

var count = Program.context.orders\_.Where(o => o.id\_client == clientId).Sum(o => o.services\_.cost) ?? 0;

var discount = DiscountCalculation.calculateDiscount((int)count);

var cartForm = new CartForm(clientId, discount);

cartForm.Owner = this;

cartForm.ShowDialog();

CheckCollection();

UpdateFlowPanel();

}

Со стороны менеджера взаимодействие с услугами происходит следующим образом: после входа в аккаунт менеджера на главной форме появляется кнопка «Заказы». После перехода на форму заказов, менеджер может выбрать одно из следующих действий: просмотреть все заказы за всё время с возможностью поиска; выбрать из списка заказ и отредактировать его; выбрать из списка заказ и подтвердить его (подтверждение подразумевает собой смену статуса на следующий: новая -> в прокате -> закрыта).

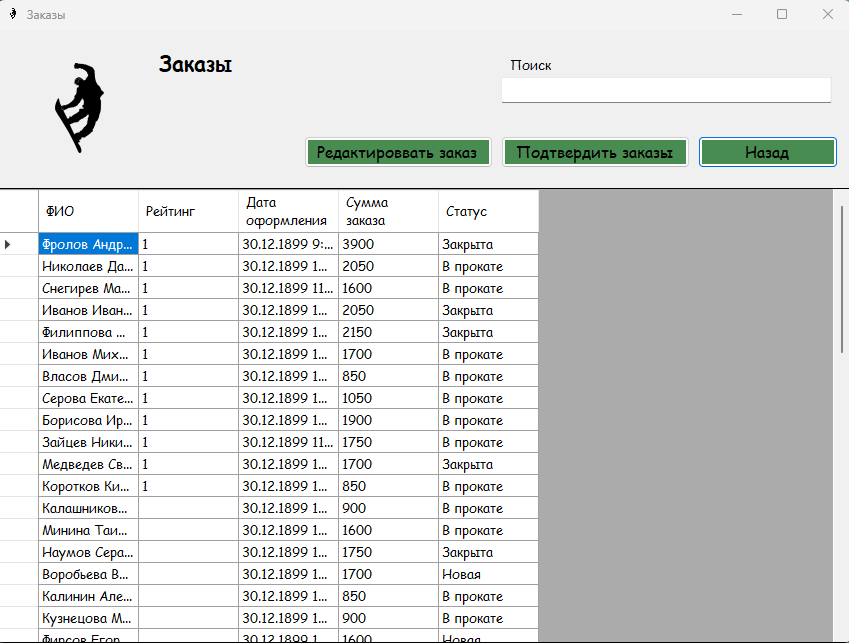


Рисунок 40. Просмотр заказов для менеджера

В форме редактирования заказа менеджер может изменить все составляющие выбранного заказа и сохранить изменения. Также предусматривается возможность поиска по ключевым словам.

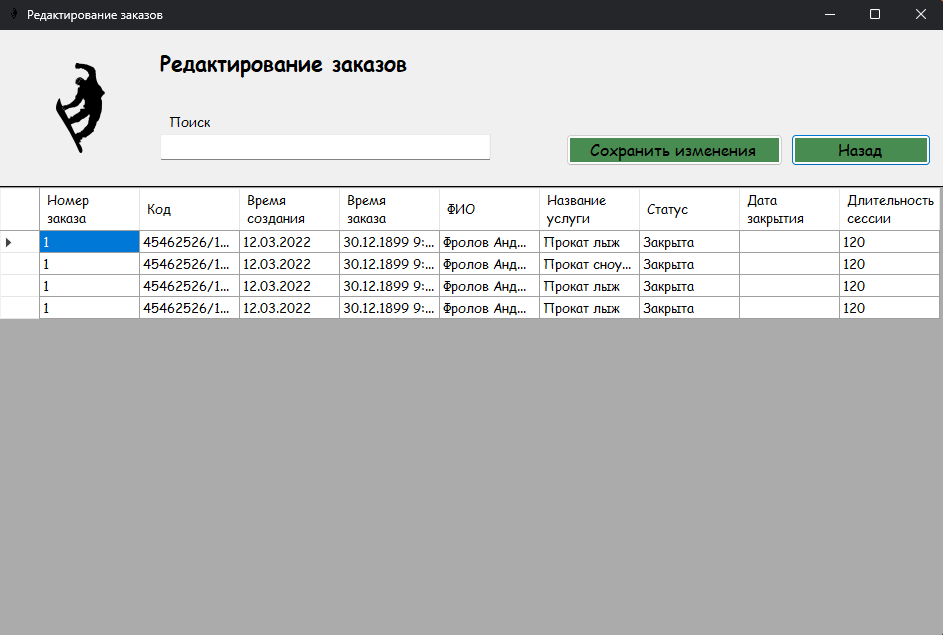


Рисунок 41. Редактирование заказа для менеджера

Код отвечающий за поиск заказов в списке:

private void UpdateData()

{

ordersRows.Clear();

var orders = Program.context.orders\_.GroupBy(or => or.order\_number).ToList();

foreach (var o in orders)

{

OrdersRow ordersRow = new OrdersRow();

var statusId = o.First().status\_id;

var status = Program.context.statuses\_.FirstOrDefault(s => s.id\_status == statusId);

ordersRow.OrderNumber = o.First().order\_number;

ordersRow.Fio = o.First().clients\_.fio;

ordersRow.Date = o.First().order\_time;

ordersRow.Sum = o.Sum(order => order.services\_.cost);

ordersRow.rating = o.First().clients\_.rating;

ordersRow.status = status.status;

if (ordersRow.Fio.ToLower().Contains(SearchTb.Text.ToLower()))

{

ordersRows.Add(ordersRow);

}

}

dataGridView1.DataSource = null;

dataGridView1.DataSource = ordersRows;

dataGridView1.Columns["orderNumber"].Visible = false;

}

Код отвечающий за подтверждение заказов:

private void AcceptBtn\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count <= 0)

{

MessageBox.Show("Необходимо выбрать строку для подтверждения", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

var selectedRow = dataGridView1.SelectedRows[0];

if (selectedRow == null)

{

MessageBox.Show("Необходимо выбрать строку для подтверждения", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

var id = (int)selectedRow.Cells["orderNumber"].Value;

var editOrders = Program.context.orders\_.Where(o => o.order\_number == id).ToList();

foreach (var order in editOrders)

{

if (order.status\_id == 1)

{

order.status\_id = 2;

}

else if (order.status\_id == 2)

{

order.status\_id = 3;

}

else

{

MessageBox.Show("Статус заказа \"Закрыт\". Изменить нельзя", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

Program.context.SaveChanges();

}

UpdateData();

MessageBox.Show("Статус был обновлён!", "Статус", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

## 2.3.3. Разработка библиотеки и подключение её к проекту

Для создания библиотеки необходимо нажать ПКМ по решению -> выбрать «Добавить» -> «Создать проект» -> «Библиотека классов (.NET Framework)» -> Выбрать название и создать проект.

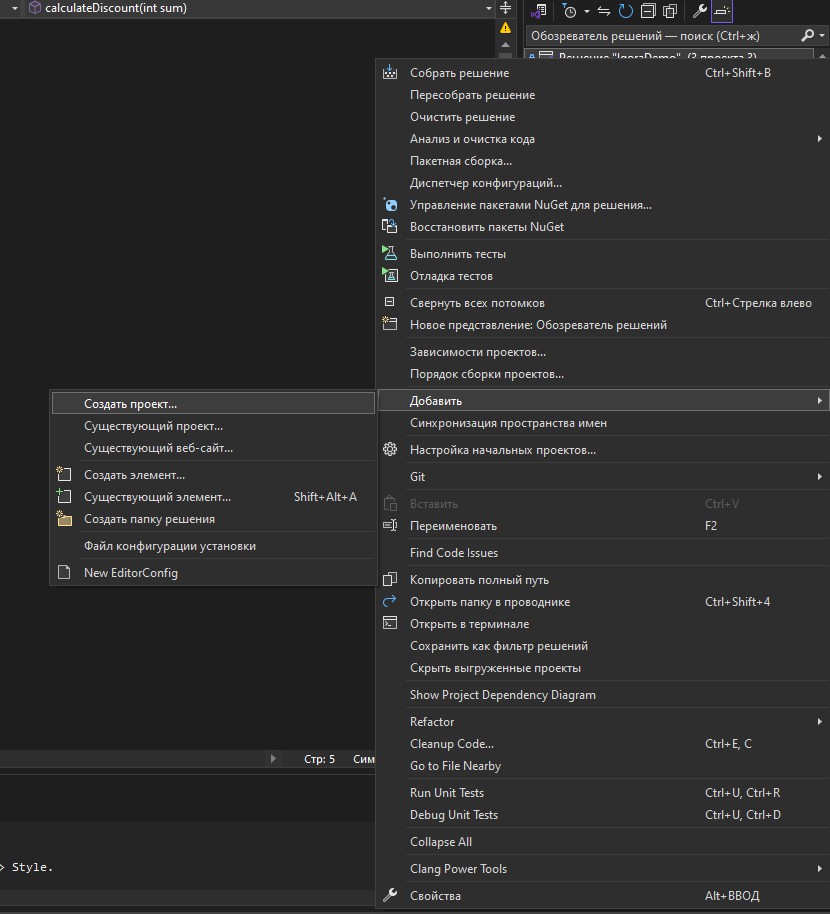


Рисунок 42. Добавление проекта библиотеки в решение

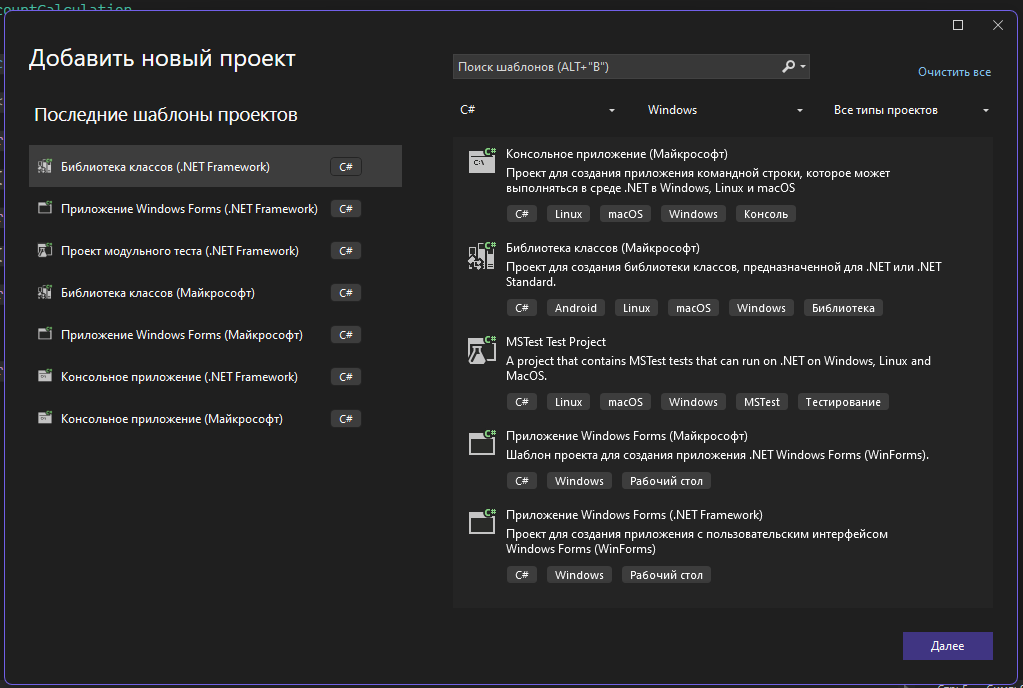


Рисунок 43. Выбор типа проекта

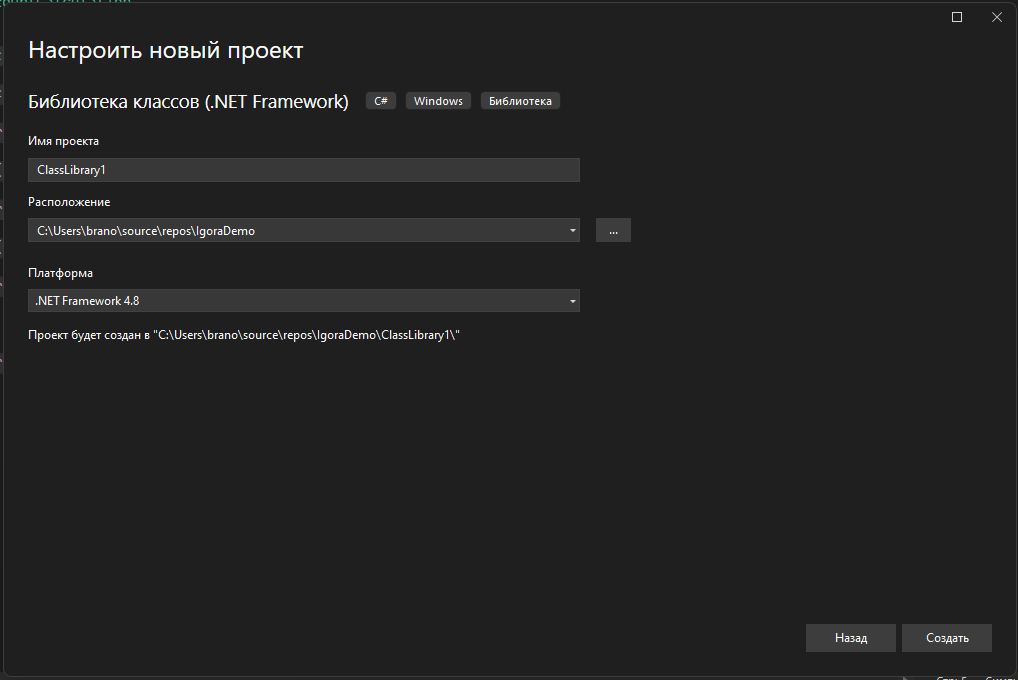


Рисунок 44. Выбор имени для библиотеки

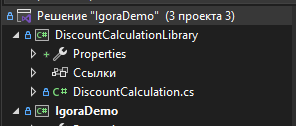


Рисунок 45. Обозреватель решений с двумя проектами

Для создания ссылки на проект (связывания двух проектов) необходимо в другом проекте нажать ПКМ по кнопку «Ссылки» -> «Добавить ссылку» -> «Проекты» -> Пометить галочкой нужный проект для создания ссылки. Далее можно будет использовать библиотеку так, словно она часть проекта.

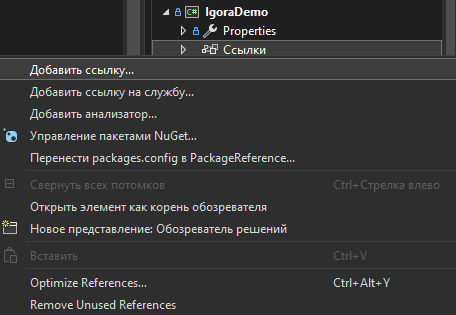


Рисунок 46. Добавление ссылки на проект

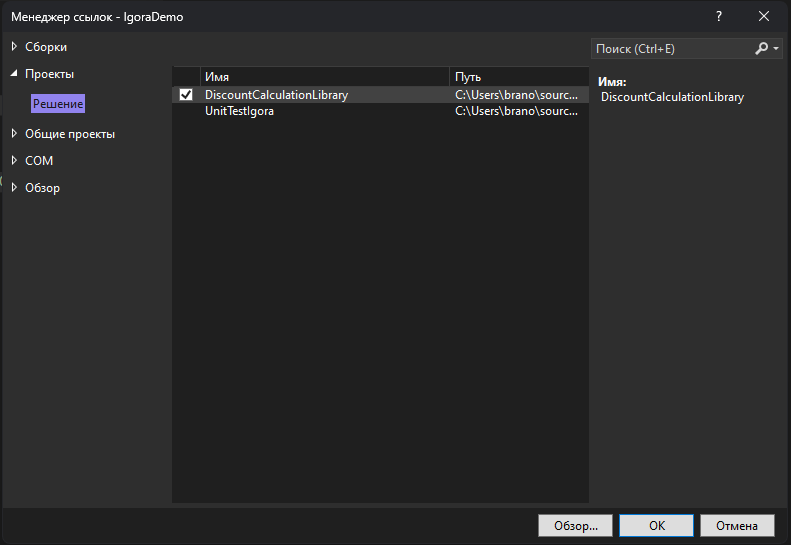


Рисунок 47. Выбор проекта для добавления ссылки

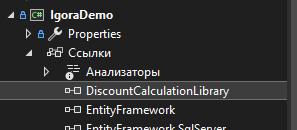


Рисунок 48. Добавленная ссылка на проект в обозревателе решений

# 2.4. Представление результатов работы

Для представления результатов работы использовался git в сочетании с gogs-сервером.

Ссылка на репозиторий - <http://gogs.wsr.ru:3000/975-21/Ponomarev02Igora>.

Для начала необходимо создать репозиторий на сервере gogs: Нажать на «плюсик» -> «Новый репозиторий» -> указать название репозитория и нажать «Создать репозиторий».

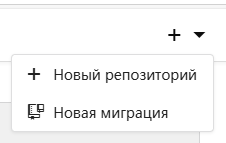


Рисунок 49. Новый репозиторий

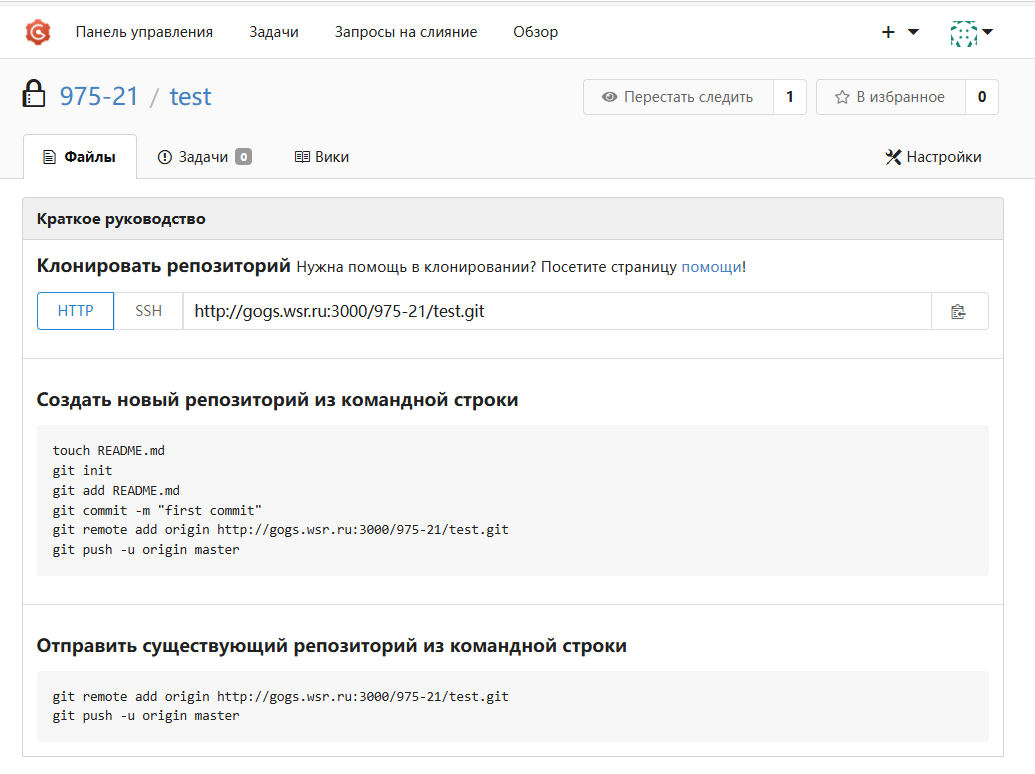


Рисунок 50. Пустой удаленный репозиторий

Далее необходимо занести на удалённый репозиторий проект: Открыть git bash в корне проекта -> Написать команду «git init» для инициализации git-репозитория -> Написать команду «git add .» для добавления всех файлов в индекс -> Написать команду «git commit –m “message”» для фиксации всех изменений -> Написать команду «git remote add origin http://gogs.wsr.ru:3000/975-21/test.git» для добавление удалённого репозитория в локальный -> написать команду "git push –u origin master» для отправки данных на удаленный репозиторий gogs.

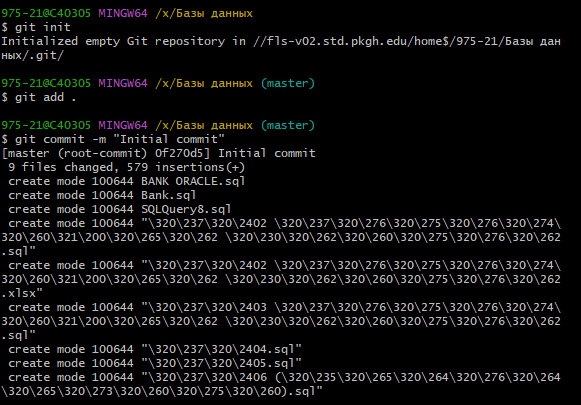


Рисунок 51. Команды git для сохранения файлов

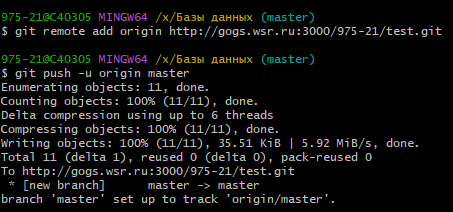


Рисунок 52. Команды git для отправки файлов на удалённый репозиторий

Далее можно проверить результат на сайте gogs: отправленные файлы отобразились в удаленном репозитории.

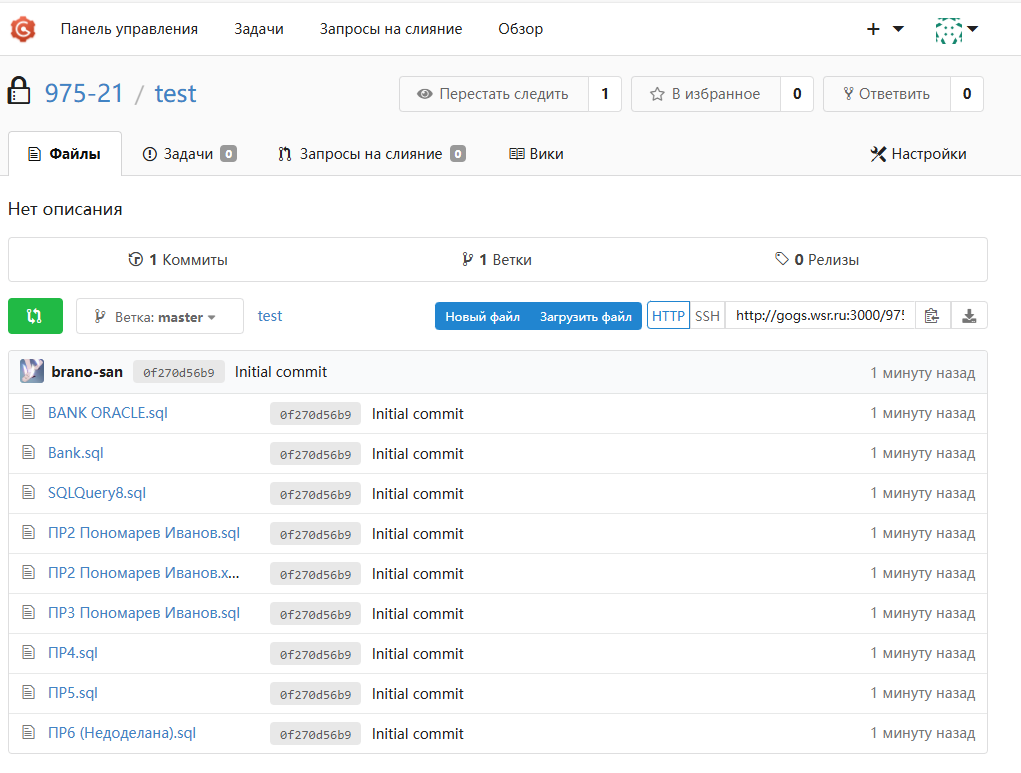
****

Рисунок 53. Файлы на удалённом репозитории

# 3. Заключение

Разработка информационной системы для точки проката спортивного инвентаря ООО «Игора» позволит оптимизировать процессы взаимодействия с клиентами и повысить эффективность работы персонала. Система обеспечит удобный доступ к информации о предоставляемых услугах как для авторизованных, так и для неавторизованных пользователей, а также упростит процесс формирования заказов через веб-интерфейс и терминалы на точках проката.

Для сотрудников компании система предоставит необходимые инструменты для управления клиентской базой, обработки заказов и контроля услуг. Возможность администрирования контента позволит гибко адаптировать ассортимент и актуализировать данные о доступных услугах.

Внедрение данной информационной системы станет важным шагом в цифровизации деятельности ООО «Игора», что обеспечит повышение качества обслуживания клиентов, сокращение времени на выполнение рутинных операций и создаст дополнительные конкурентные преимущества для горнолыжного курорта.

# Приложение А

**Код библиотеки:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace IgoraDemo.Services

{

internal class DiscountCalculation

{

public static int calculateDiscount(int sum)

{

if (sum < 50\_000)

{

return 0;

}

else if (sum >= 50\_000 && sum < 100\_000)

{

return 1;

}

else if (sum >= 100\_000 && sum < 300\_000)

{

return 3;

}

else

{

return 5;

}

}

}

}