**МИНОБРНАУКИ РОССИИ**

федеральное государственное автономное образовательное учреждение

высшего образования

«Санкт-Петербургский политехнический университет Петра Великого»

(ФГАОУ ВО «СПбПУ»)

**Институт среднего профессионального образования**

**ОТЧЕТ**

**по учебной практике УП.02.01 (по профилю специальности)**

по профессиональному модулю ПМ.02 «Осуществление интеграции программных модулей»

(код и наименование)

Специальность09.02.07 Информационные системы и программирование

(код и наименование специальности)

Студентка 4 курса 42919/8 группы

Шевелева Мария Александровна

(Фамилия, имя, отчество)

Место прохождения практики: УВЦ, пр. Энгельса, 23

(наименование и адрес организации)

Период прохождения практики

с «07» октября 2024 г. по «19» октября 2024 г.

Руководитель практики Хисамутдинова А.С.

(подпись) (расшифровка подписи)

Итоговая оценка по практике \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Санкт-Петербург

2024

**ЗАДАНИЕ**

**на учебную практику (по профилю специальности)**

по профессиональному модулю ПМ.02 «Осуществление интеграции программных модулей»

(код и наименование)

Специальность09.02.07 Информационные системы и программирование

(код и наименование специальности)

Студентка 4 курса 42919/8 группы

Шевелева Мария Александровна

(Фамилия, имя, отчество)

Место прохождения практики: УВЦ, пр. Энгельса, 23

(наименование и адрес организации)

Период прохождения практики

с «07» октября 2024 г. по «19» октября 2024 г.

**Виды работ, обязательные для выполнения** *(переносится из программы, соответствующего ПМ):*

* Участие в выработке требований к программному обеспечению;
* Стадии проектирования программного обеспечения. Проектирование UML-диаграмм;
* Конструирование пользовательского интерфейса. Разработка модулей программного обеспечения;
* Разработка рабочего проекта и технологической документации.

**Индивидуальное задание: ВАРИАНТ 23**

Задание выдал «07» октября 2024 г. \_\_\_\_\_\_\_\_\_\_\_\_\_ Хисамутдинова А.С.

(подпись) (Ф.И.О.)

Задание получил «07» октября 2024 г \_\_\_\_\_\_\_\_\_\_\_\_ Шевелева М.А. (подпись) (Ф.И.О.)

**МИНОБРНАУКИ РОССИИ**

федеральное государственное автономное образовательное учреждение

высшего образования

«Санкт-Петербургский политехнический университет Петра Великого»

(ФГАОУ ВО «СПбПУ»)

**Институт среднего профессионального образования**

**ДНЕВНИК**

**прохождения учебной практики УП.02.01**

**(по профилю специальности)**

по профессиональному модулю ПМ.02 «Осуществление интеграции программных модулей»

(код и наименование)

Специальность09.02.07 Информационные системы и программирование

(код и наименование специальности)

Студентка 4 курса 42919/8 группы

Шевелева Мария Александровна

(Фамилия, имя, отчество)

Место прохождения практики: УВЦ, пр. Энгельса, 23

(наименование и адрес организации)

Период прохождения практики

с «07» октября 2024 г. по «19» октября 2024 г.

Руководитель практики Хисамутдинова А.С.

(подпись) (расшифровка подписи)

Итоговая оценка по практике

Санкт-Петербург

2024

**Содержание дневника**

|  |  |  |
| --- | --- | --- |
| **Дата** | **Виды выполненных работ и заданий по программе практики** | **Подпись руководителя практики** |
| **1** | **2** | **3** |
| 07.10.2024 | Анализ предметной области. Анализ методов решения. Анализ и выбор средств. |  |
| 08.10.2024 | Анализ и выбор средств. Исследовательские работы. Разработка технического задания. |  |
| 09.10.2024 | UML. Проектирование диаграммы вариантов использования и последовательности. UML. Проектирование диаграммы вариантов активности и состояния. |  |
| 10.10.2024 | Моделирование структуры ПО. Проектирование инфологической и даталогической модели данных. |  |
| 11.10.2024 | Проектирование интерфейса пользователя. |  |
| 12.10.2024 | Разработка схемы алгоритма программного продукта. |  |
| 14.10.2024 | Разработка модели данных. |  |
| 15.10.2024 | Разработка программного продукта. Создание функциональности ПО. |  |
| 16.10.2024 | Модульное тестирование. Создание тестовых случаев. |  |
| 17.10.2024 | Создание тестовых случаев. Отладка программных модулей. |  |
| 18.10.2024 | Разработка документа «Руководство пользователя» в соответствии с ЕСПД. |  |
| 19.10.2024 | Разработка документа «Программа и методика испытаний» в соответствии с ЕСПД. Подготовка отчета. |  |

Содержание

[ВВЕДЕНИЕ 6](#_Toc180051225)

[Задание №1 «Разработка технического задания» 7](#_Toc180051226)

[Задание № 2 «Разработка алгоритмов и диаграмм» 11](#_Toc180051227)

[Задание №3 «Разработка макетов программы» 16](#_Toc180051228)

[Задание №4 «Разработка программы» 23](#_Toc180051229)

[Задание №5 «Тестирование» 30](#_Toc180051230)

[ЗАКЛЮЧЕНИЕ 35](#_Toc180051231)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 36](#_Toc180051232)

[Приложение А 37](#_Toc180051233)

[Приложение Б 43](#_Toc180051234)

ВВЕДЕНИЕ

Учебная практика, проводимая в рамках данного курса, направлена на комплексное изучение и применение различных аспектов разработки программного обеспечения. Основные задачи практики включают участие в выработке требований к программному обеспечению, проектирование UML-диаграмм, конструирование пользовательского интерфейса, разработку модулей программного обеспечения, а также создание рабочего проекта и технологической документации.

В ходе практики я получила возможность применить теоретические знания на практике, пройдя через все этапы разработки программного обеспечения. Это включало в себя определение требований, проектирование архитектуры системы с использованием UML-диаграмм, создание интуитивно понятного пользовательского интерфейса и разработку отдельных модулей программного обеспечения.

Результатом выполнения данной работы стало создание полноценного программного продукта, который сопровождался подробной технологической документацией. Это позволило мне не только закрепить полученные знания, но и приобрести ценные практические навыки, необходимые для успешной работы в области разработки программного обеспечения.

Таким образом, данная учебная практика имела для меня как теоретическое, так и практическое значение, способствуя моему всестороннему развитию и подготовке к профессиональной деятельности в сфере информационных технологий.

Задание №1 «Разработка технического задания»

Вариант 23: Химчистка

Описание предметной области

Основной целью данного модуля является автоматизация процесса управления химчисткой. Модуль позволяет сотрудникам химчистки прозрачно и эффективно управлять заказами, клиентами, услугами и отслеживать текущий статус выполнения заказов.

Разработка программного модуля для управления химчисткой - это процесс создания программного продукта, который позволяет химчисткам эффективно отслеживать и управлять заказами, клиентами и услугами. Задача разработки такого программного модуля состоит в создании удобного и эффективного инструмента для учета, отслеживания и управления процессами химчистки, что помогает минимизировать время простоя и обеспечивает бесперебойную работу химчистки.

Программный модуль позволяет химчистке повысить эффективность и точность учета заказов, сократить время обработки заказов, оптимизировать использование ресурсов и повысить удовлетворенность клиентов.

Основные функции и возможности модуля включают:

1. Управление заказами: это информация о заказах, поступающих в химчистку. Заказ может содержать данные о типе услуги, предмете (например, одежда, обувь), дате приема, дате выполнения, статусе заказа (принят, в процессе, выполнен, отменен).
2. Регистрация заказа: этот процесс включает приём и регистрацию заказа сотрудником химчистки в системе учёта. Важными аспектами регистрации являются присвоение уникального идентификатора заказу, сохранение информации о заказе.
3. Обработка заказа: процесс, включающий анализ заказа, определение его приоритетности и назначение исполнителя (сотрудника химчистки) для выполнения заказа. В процессе обработки может потребоваться дополнительная информация или уточнение деталей заказа у клиента.
4. Исполнение заказа: фактическое выполнение заказа. На этом этапе назначенный сотрудник выполняет услугу, вносит необходимые изменения в статус заказа и предмета. Важно отметить, что на этом этапе могут возникать необходимость уведомления клиента о готовности заказа или координации работ с другими специалистами.
5. Отчётность и информирование: важной составляющей учёта заказов является фиксация и отчёт о выполненной работе. После завершения заказа сотрудник должен предоставить отчёт о проделанной работе, включая информацию о затраченных ресурсах (время, материалы) и оказанной помощи.
6. Мониторинг и анализ: этот этап предполагает контроль и анализ процесса учёта заказов. Важно отслеживать и анализировать время обработки заказов, качество выполненных работ, расходы и прочие параметры, которые могут помочь в оптимизации и улучшении процесса.

Техническое задание

1. Общие сведения

1.1. Наименование проекта: Разработка программного модуля для управления химчисткой.

1.2. Заказчик: Химчистка «Чистота».

1.3. Исполнитель: Компания «ТехноЧист».

2. Функциональные требования

Роли пользователей:

* Администратор: Управляет всеми аспектами системы, включая добавление и редактирование услуг, клиентов, заказов, а также назначение ролей и прав доступа.
* Сотрудник (Химчистка): Управляет заказами, выполняет услуги, а также обновляет информацию о заказах и клиентах.
* Клиент: Может просматривать доступные услуги, делать заказы, отслеживать статус своих заказов и получать уведомления.

Возможность добавления заявок в базу данных с указанием следующих параметров:

* + - Номер заявки;
    - Дата добавления;
    - Вид одежды или обуви;
    - Описание проблемы;
    - ФИО клиента;
    - Номер телефона;
    - Статус заявки (новая заявка, в процессе обработки, завершена).

Возможность редактирования заявок:

* + - Изменение этапа выполнения (готова к выдаче, в процессе обработки, ожидание химических средств);
    - Изменение описания проблемы;
    - Изменение ответственного за выполнение работ.

Возможность отслеживания статуса заявки:

* + - Отображение списка заявок;
    - Получение уведомлений о смене статуса заявки;
    - Поиск заявки по номеру или по параметрам.

Возможность назначения ответственных за выполнение работ:

* + - Добавление специалиста по химчистке к заявке;
    - Отслеживание состояния работы и получение уведомлений о ее завершении;
    - Специалист по химчистке может добавлять комментарии на форме заявки и фиксировать информацию о заказанных химических средствах и материалах.

1. Нефункциональные требования:

Кроссплатформенность:

* + - Поддержка работы на ОС семейства Windows.

Безопасность:

* + - Логин и пароль для доступа к приложению;
    - Доступ к данным должен быть ограничен в зависимости от роли пользователя.

Удобство использования:

* + - Простой и интуитивный интерфейс;
    - Информативные уведомления и подсказки.

Производительность:

* + - Приложение должно иметь быстрый доступ к данным;
    - Минимальное время отклика на запросы пользователя.

1. Требования к реализации:

Язык программирования: C#

СУБД: Microsoft SQL Server

1. Требования к документации:

* Техническое задание на разработку программного модуля.

6. Руководство по стилю

6.1. Шрифт: Arial.

6.2. Цветовая схема: Синий фон, белый текст.

Задание № 2 «Разработка алгоритмов и диаграмм»

По составленному техническому заданию необходимо разработать алгоритмы в виде блок-схем и UML-диаграммы:

* Диаграмма вариантов использования (Рисунок 1)



Рисунок 1 – Диаграмма вариантов использования

* Диаграммы последовательности (Рисунок 2)



Рисунок 2 – Диаграмма последовательностей



Рисунок 3 – Диаграмма активностей

* ER-диаграмма для БД (Рисунок 4)



Рисунок 4 – ER-диаграмма

* Таблицы базы данных (Таблица 1 – 6)

Таблица 1 - services

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **FILED NAME** | **DATA TYPE** | **REQUIRED?** | **NOTES** |
| PK | service\_id | INT | Y | Идентификатор таблицы |
|  | name | NVARCHAR(50) | Y | Название услуги |

Таблица 2 – OrderTable1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **FILED NAME** | **DATA TYPE** | **REQUIRED?** | **NOTES** |
| PK | order\_id | INT | Y | Идентификатор таблицы |
|  | employee\_id | INT | Y | Идентификатор сотрудника |
|  | service\_id | INT | Y | Идентификатор услуги |
|  | description | NVARCHAR(200) | Y | Описание проблемы |
|  | created\_at | DATETIME | Y | Время создания (создается автоматически, если не было заполнено) |

Таблица 3 – clients

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **FILED NAME** | **DATA TYPE** | **REQUIRED?** | **NOTES** |
| PK | сlient\_id | INT | Y | Идентификатор таблицы |
|  | name | NVARCHAR(50) | Y | Имя клиента |
|  | phone | NVARCHAR(20) | Y | Телефон клиента |

Таблица 4 – employees

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **FILED NAME** | **DATA TYPE** | **REQUIRED?** | **NOTES** |
| PK | employee\_id | INT | Y | Идентификатор таблицы |
|  | name | NVARCHAR(50) | Y | Имя работника |
|  | position | NVARCHAR(50) | Y | Позиция сотрудника |

Таблица 5 – Goods

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **FILED NAME** | **DATA TYPE** | **REQUIRED?** | **NOTES** |
| PK | good\_id | INT | Y | Идентификатор таблицы |
|  | order\_id | INT | Y | Идентификатор заказа |
|  | name | NVARCHAR(50) | Y | Название товара |

Задание №3 «Разработка макетов программы»

Wireframe макеты (Рисунок 5-11).



Рисунок 5 – Wireframe макет для вкладки Авторизация

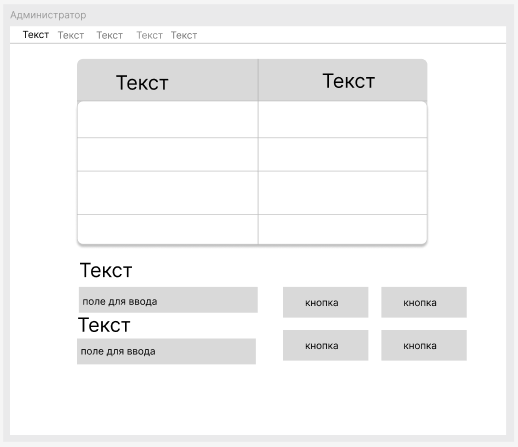


Рисунок 6 – Wireframe макет для приложения Администратора (1-я таблица)

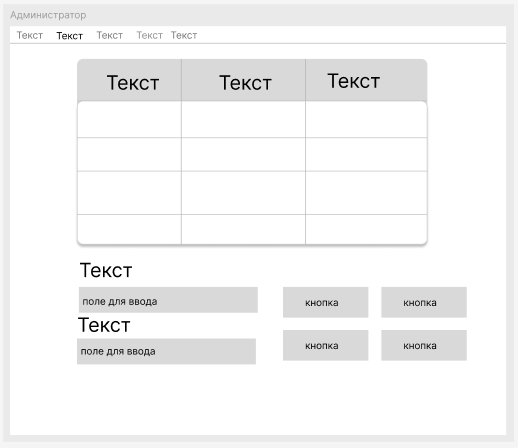


Рисунок 7 – Wireframe макет для приложения Администратора (2-я таблица)

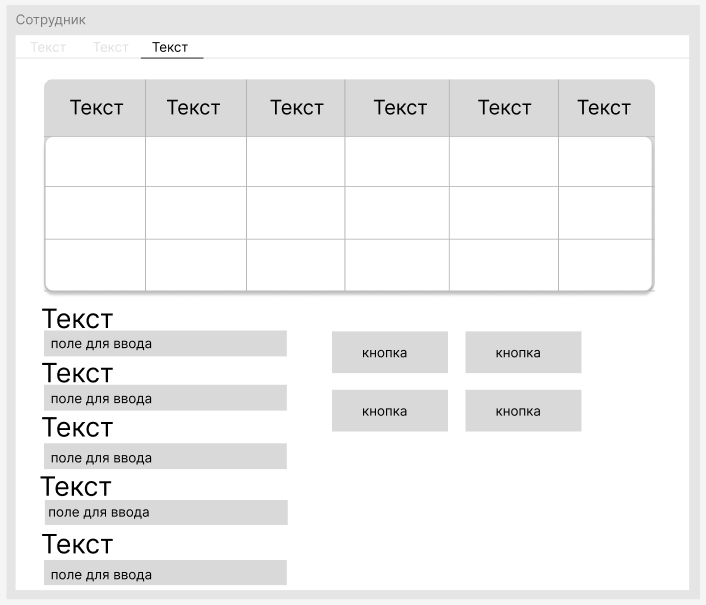


Рисунок 8 – Wireframe макет для приложения сотрудника (3-я таблица)

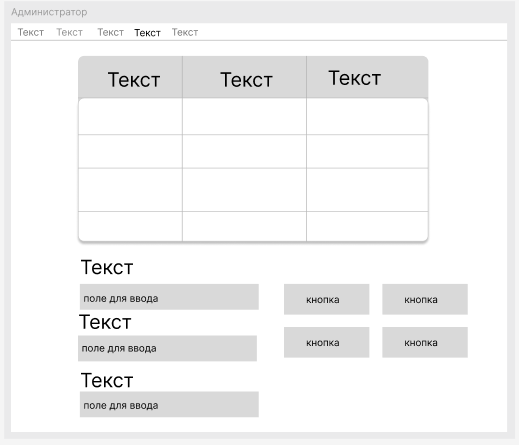


Рисунок 9 – Wireframe макет для приложения администратора (4-я таблица)

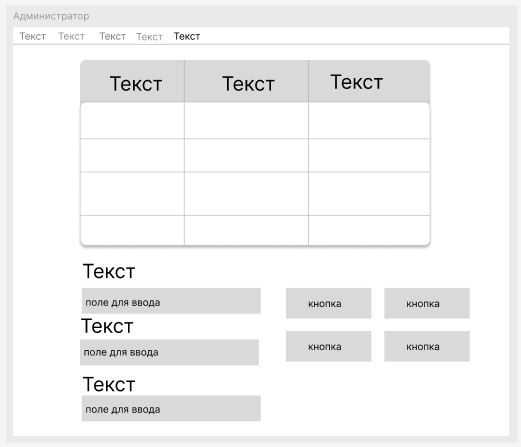


Рисунок 10 – Wireframe макет для приложения администратора (5-я таблица)

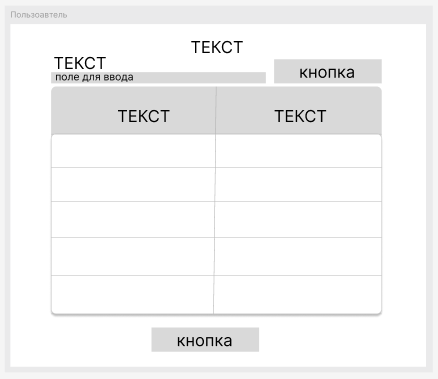


Рисунок 11– Wireframe макет для приложения пользователя

Mockup макеты (Рисунок 12-18).

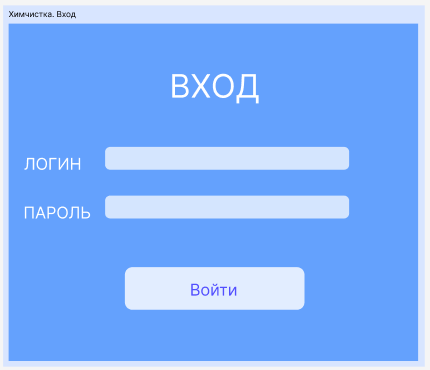


Рисунок 12– Mockup макет для Авторизации



Рисунок 13– Mockup макет для приложения Администратора (таблица Услуги)

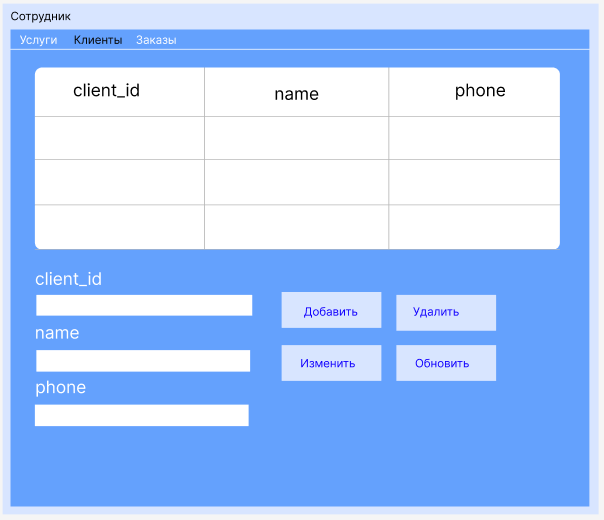


Рисунок 14– Mockup макет для приложения Администратора (таблица Клиенты)

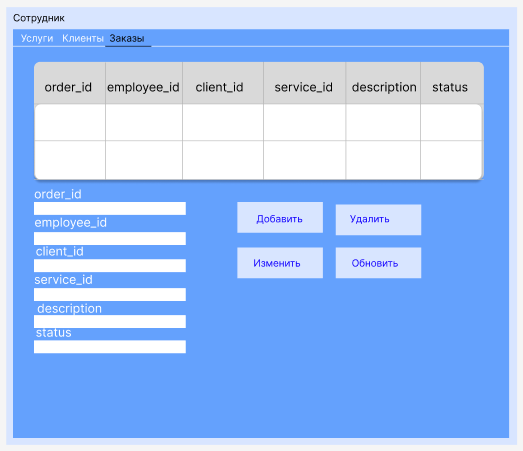


Рисунок 15– Mockup макет для приложения сотрудника (таблица Заказы)

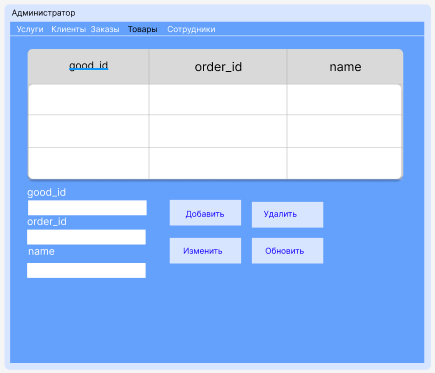


Рисунок 16– Mockup макет для приложения администратора (таблица Товары)

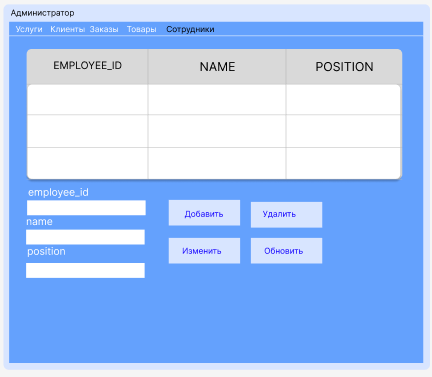


Рисунок 17– Mockup макет для приложения администратора (таблица Сотрудники)

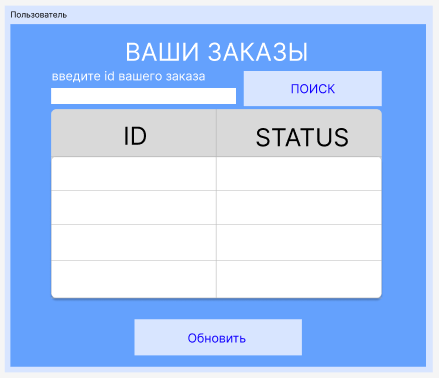


Рисунок 18– Mockup макет для приложения пользователя

Карта навигации приложения (Рисунок 19).



Рисунок 19 – Карта навигации

Задание №4 «Разработка программы»

Код ко всем формам представлен в приложении А.

При входе, открывается форма входа (Рисунок 20).

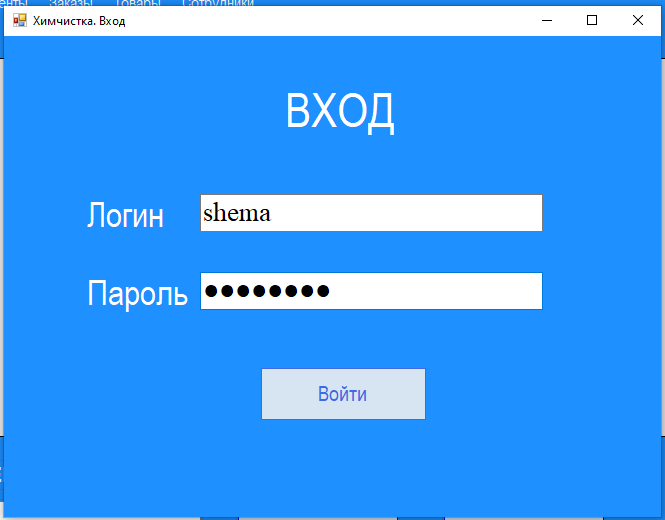


Рисунок 20 – Форма входа

При входе под любой ролью при верно введённом пароле появляется следующее окно (Рисунок 21).

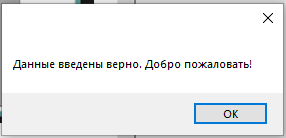


Рисунок 21 – Окно после входа

Вход под администратором отображение таблиц (Рисунок 22 -26).

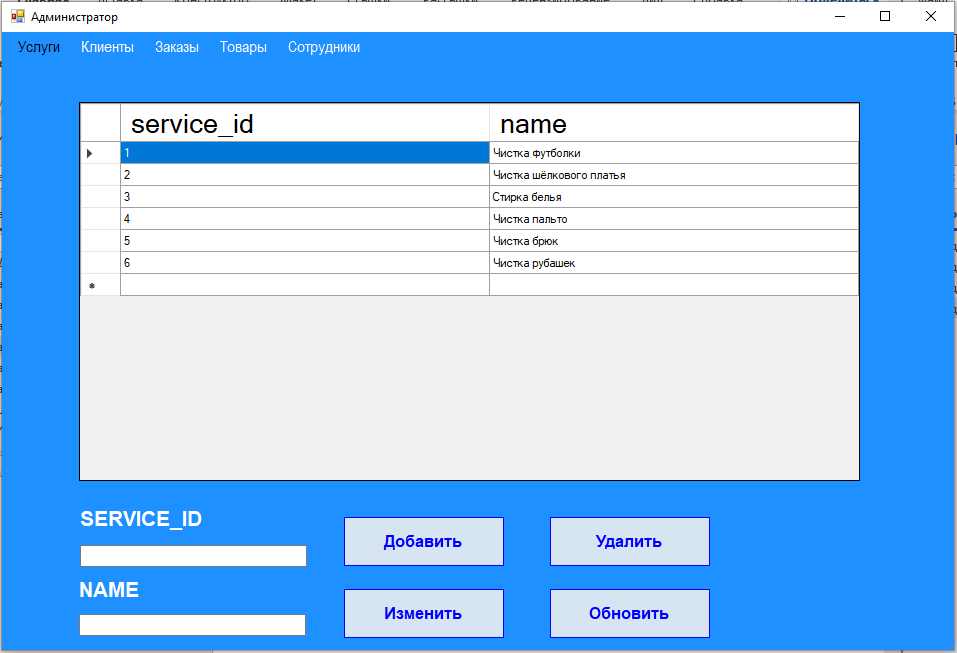


Рисунок 22 – Форма администратора (Таблица Услуги)

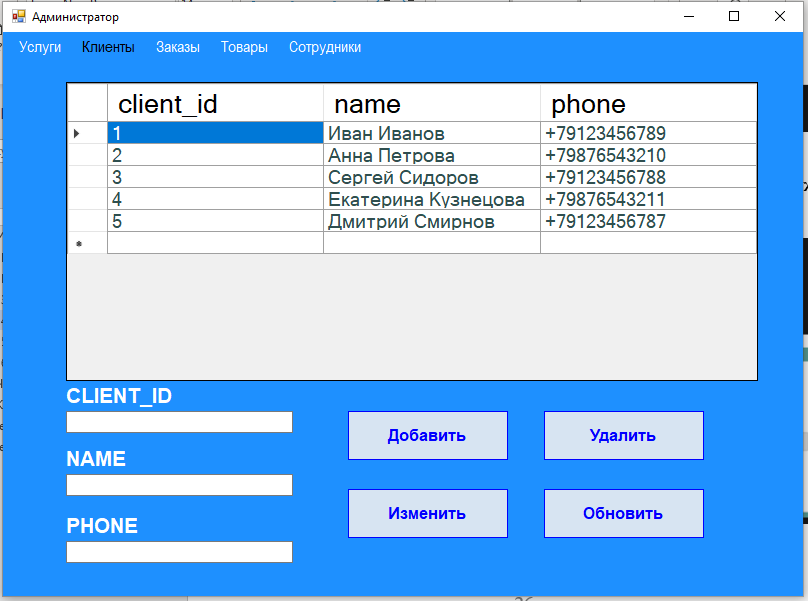


Рисунок 23 – Форма администратора (Таблица Клиенты)

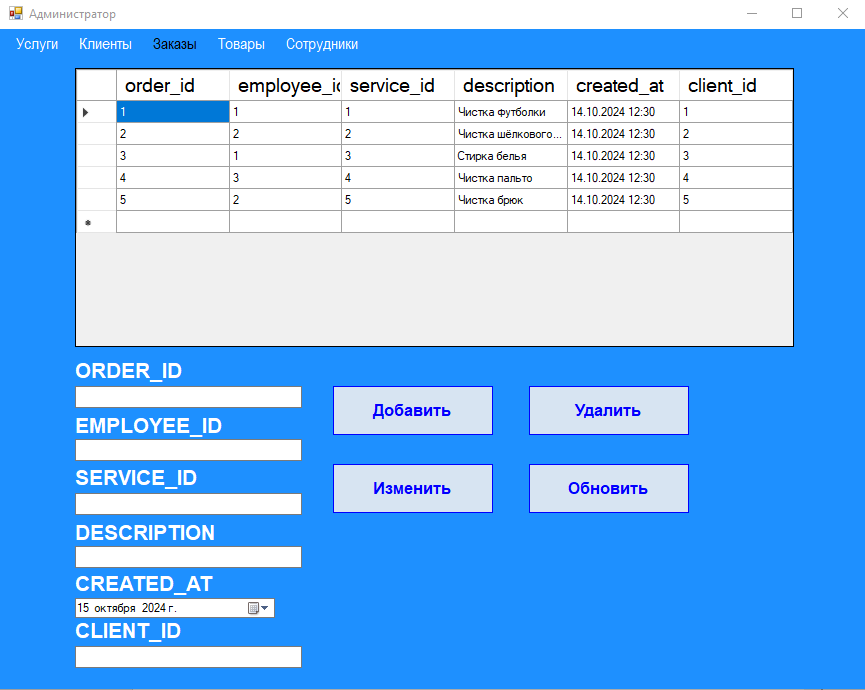


Рисунок 24 – Форма администратора (Таблица Заказы)

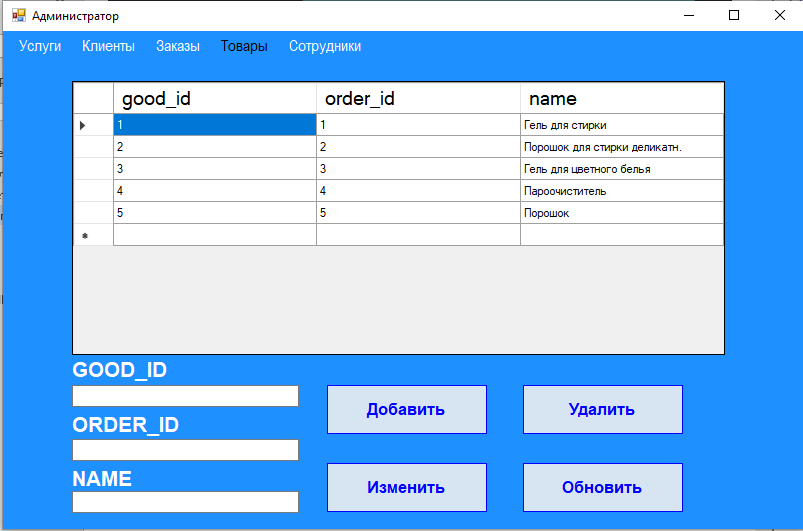


Рисунок 25 – Форма администратора (Таблица Товары)

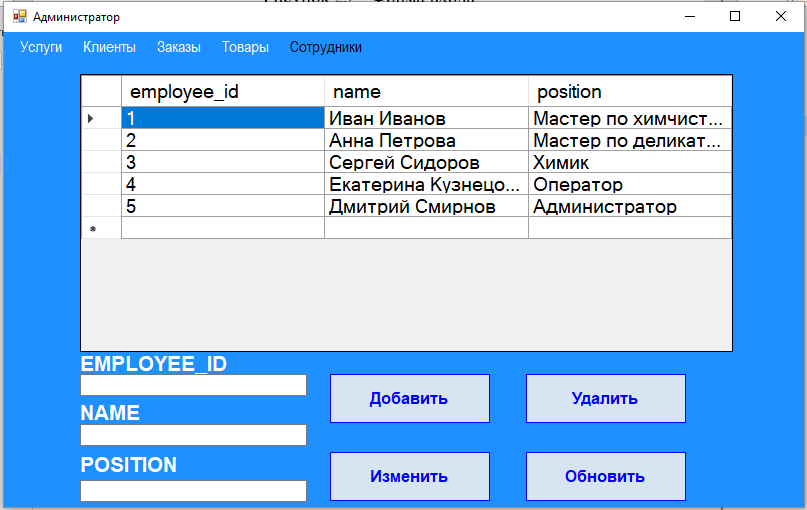


Рисунок 26 – Форма администратора (Таблица Сотрудники)

Вход под сотрудником отображение таблиц (Рисунок 27 -29).

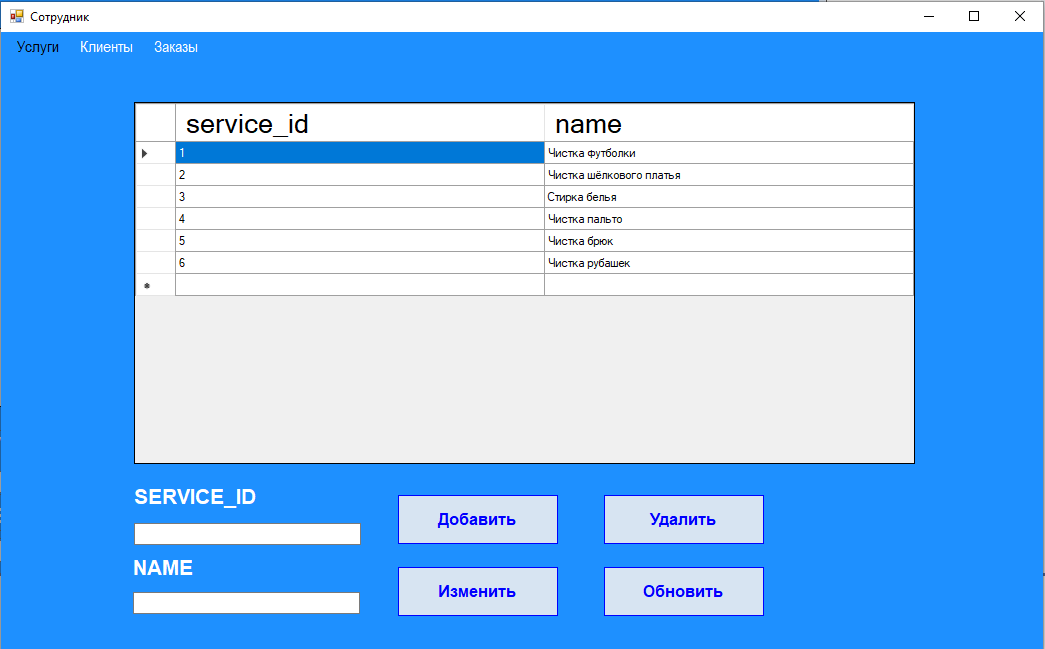


Рисунок 27 – Форма сотрудника (Таблица Услуги)

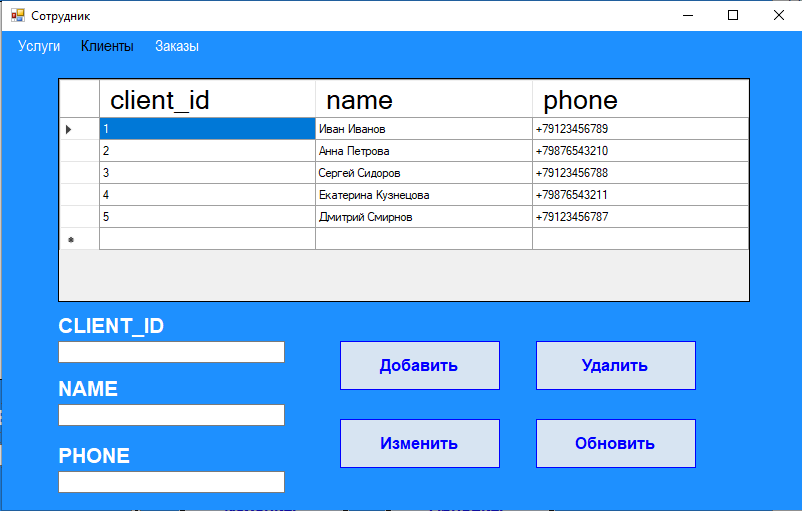


Рисунок 28 – Форма сотрудника (Таблица Клиенты)

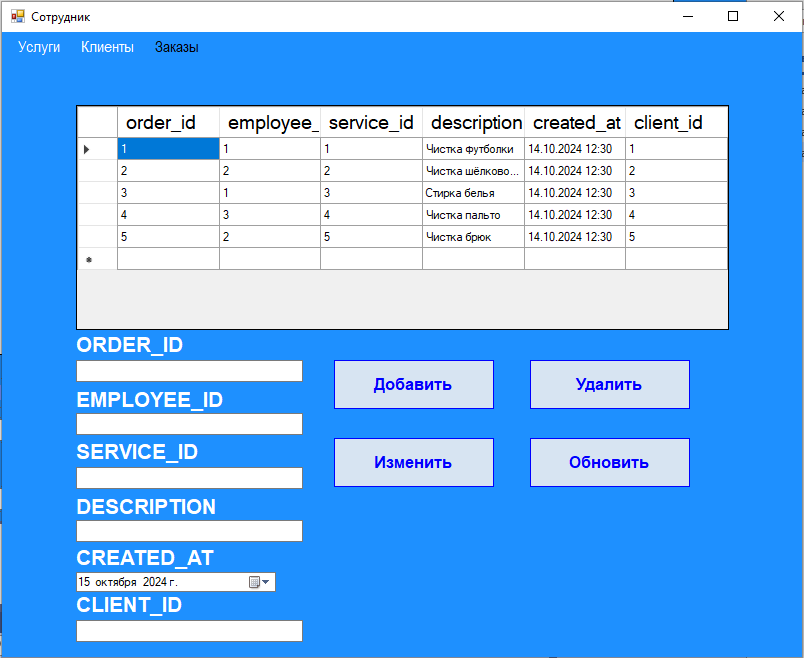


Рисунок 29 – Форма сотрудника (Таблица Заказы)

Вход под клиентом отображение таблицы Заказы и поиск заказа по ID (Рисунок 30 -32).

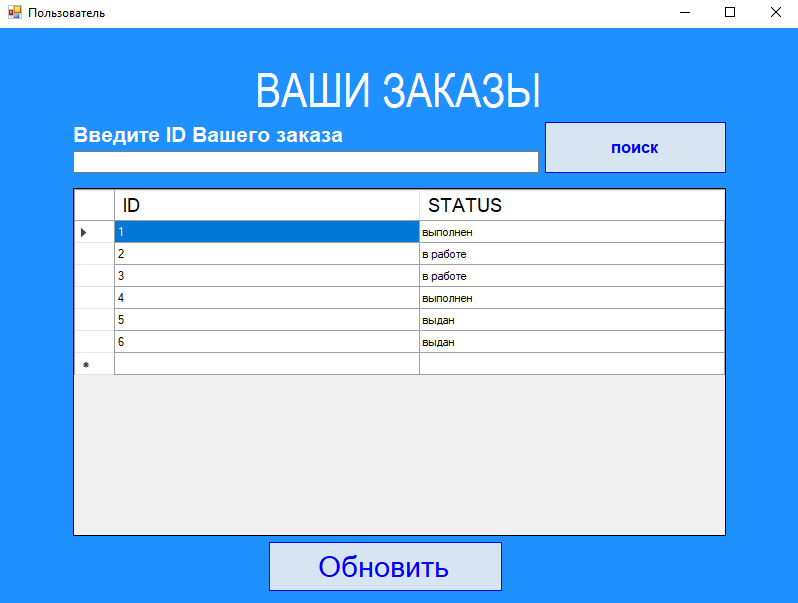


Рисунок 30 – Форма клиента

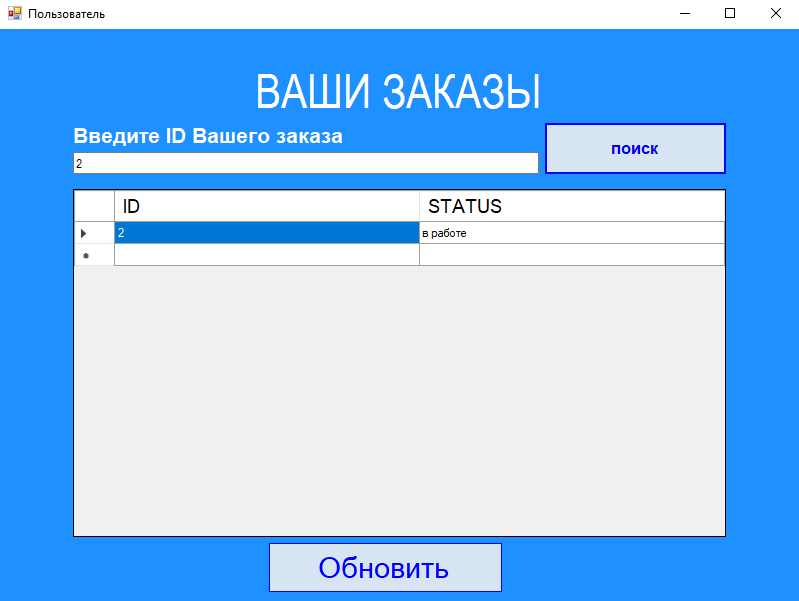


Рисунок 31 – Поиск по ID

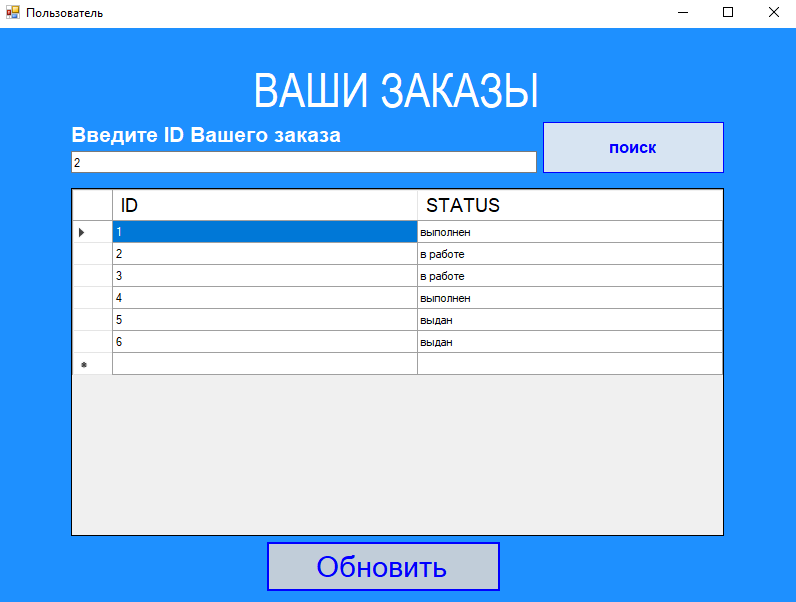


Рисунок 32 – Обновление

Задание №5 «Тестирование»

Аннотация теста (Таблица 6).

Таблица 6 – Аннотация к тестам

|  |  |
| --- | --- |
| **Название проекта** | Химчистка «Чистота» |
| **Рабочая версия** | 1.0 |
| **Имя тестирующего** | Мария |
| **Дата(ы) теста** | 16.10.2024 |

Тест-кейсы.

Разработка тестовых примеров (Таблица 7-11).

Таблица 7 – Проверка входа в систему

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_1 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Подтверждение входа сотрудника с действительным именем пользователя и паролем |
| **Краткое изложение теста** | В результате выполнения теста должна открыться форма сотрудника |
| **Этапы теста** | 1. Запустить приложение 2. Ввести логин и пароль 3. Нажать войти |
| **Тестовые данные** | 1. логин: risha 2. пароль: 87654321 |
| **Ожидаемый результат** | Открытие новой формы сотрудника и скрытие формы входа |
| **Фактический результат** | Открытие новой формы сотрудника и скрытие формы входа |
| **Статус** | Зачет |
| **Предварительное условие** | Логин и пароль должны находиться в БД |
| **Постусловие** |  |
| **Примечания/комментарии** |  |

Таблица 8 – Проверка загрузки данных у пользователя

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_2 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Загрузка данных клиента с действительным именем пользователя и паролем |
| **Краткое изложение теста** | Данные клиента должны загрузиться после успешного входа |
| **Этапы теста** | 1. Запустить приложение 2. Ввести логин и пароль 3. Нажать войти |
| **Тестовые данные** | 1. логин: klusha 2. пароль: 88005553535 |
| **Ожидаемый результат** | Открытие новой формы клиента и скрытие формы входа |
| **Фактический результат** | Данные не загрузились |
| **Статус** | Незачет |
| **Предварительное условие** | Логин и пароль должны находиться в БД |
| **Постусловие** |  |
| **Примечания/комментарии** |  |

Таблица 9 – Проверка обработки исключения с неправильным логином

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_3 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Обработка неправильного имени пользователя |
| **Краткое изложение теста** | Окно с предупреждением должно появляться при неправильном логине |
| **Этапы теста** | 1. Запустить приложение 2. Ввести неправильный логин или пароль 3. Нажать войти |
| **Тестовые данные** | 1. логин: shemaa 2. пароль: 12345678 |
| **Ожидаемый результат** | Появление окна с надписью: “Данные введены неверно” |
| **Фактический результат** | Появление окна с надписью: “Данные введены неверно” |
| **Статус** | Зачет |
| **Предварительное условие** | Логин не должен находиться в БД |
| **Постусловие** |  |
| **Примечания/комментарии** |  |

Таблица 10 – Проверка обработки исключения с неправильным паролем

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_4 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Обработка неправильного пароля |
| **Краткое изложение теста** | В результате выполнения теста должно появиться окно с предупреждением, что пароль некорректный |
| **Этапы теста** | 1. Запустить приложение 2. Ввести логин и пароль 3. Нажать войти |
| **Тестовые данные** | 1. логин: shema 2. пароль: 87654321 |
| **Ожидаемый результат** | Появление окна с надписью: “Данные введены неверно” |
| **Фактический результат** | Появление окна с надписью: “Данные введены неверно” |
| **Статус** | Зачет |
| **Предварительное условие** | Пароль не должен соответствовать с логином в БД |
| **Постусловие** |  |
| **Примечания/комментарии** |  |

Таблица 11 – Проверка ввода пустых полей

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_5 |
| **Приоритет тестирования** | Средний |
| **Заголовок/название теста** | Запуск теста с вводом пустых полей авторизации |
| **Краткое изложение теста** | В результате выполнения теста должно появиться окно с предупреждением, что не все данные были заполнены |
| **Этапы теста** | 1. Запустить приложение 2. Оставить поля логина и пароля пустыми 3. Нажать войти |
| **Тестовые данные** | 1. логин: 2. пароль: |
| **Ожидаемый результат** | Появление окна с надписью: “Некорректный ввод данных” |
| **Фактический результат** | Появление окна с надписью: “Некорректный ввод данных” |
| **Статус** | Зачет |
| **Предварительное условие** |  |
| **Постусловие** |  |
| **Примечания/комментарии** |  |

Код модульных тестов представлен в приложении Б. Результат выполнения тестов (Рисунок 33).

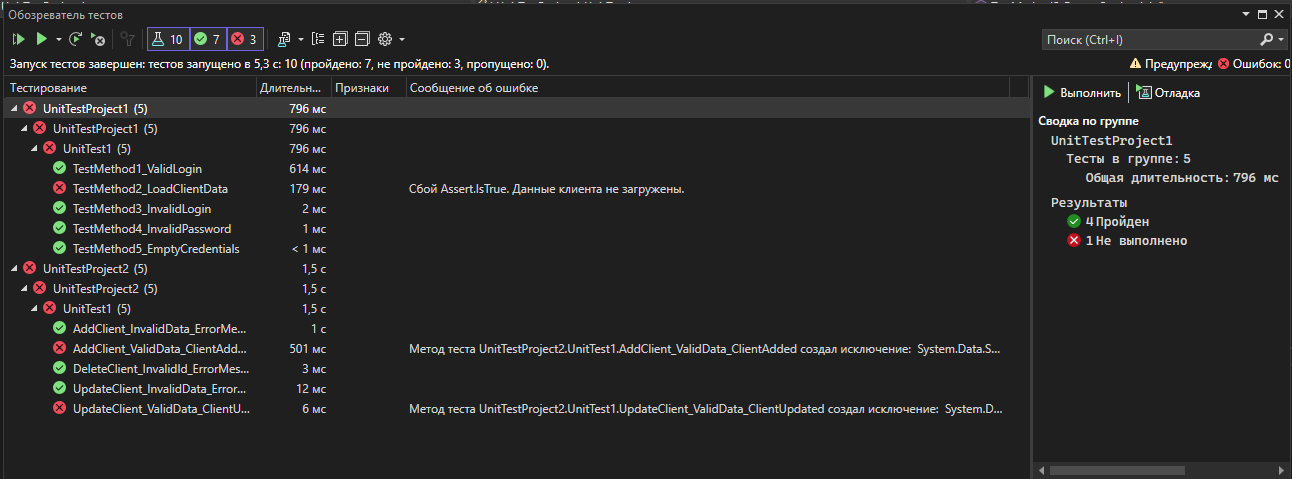


Рисунок 33 - Результат выполнения тестов

По результатам тестирования не прошли следующие тесты:

* 2 тест (UnitTestProject1), так как изначальных данных не было в БД;
* 2 тест (UnitTestProject2), так как не было настройки на ввод некорректных данных (пустой строки)
* 5 тест (UnitTestProject2), так как не было настройки на ввод некорректных данных (пустой строки), поэтому и удаление с некорректным идентификатором (пустой строкой) не сработало

Задание №6 «GitHub. Оценка проекта»

Отправление приложения с тестами и отчётом на созданный репозиторий GitHub (Рисунок 34 - ). Ссылка на созданный репозиторий - https://github.com/mariiasheveleva/up.git

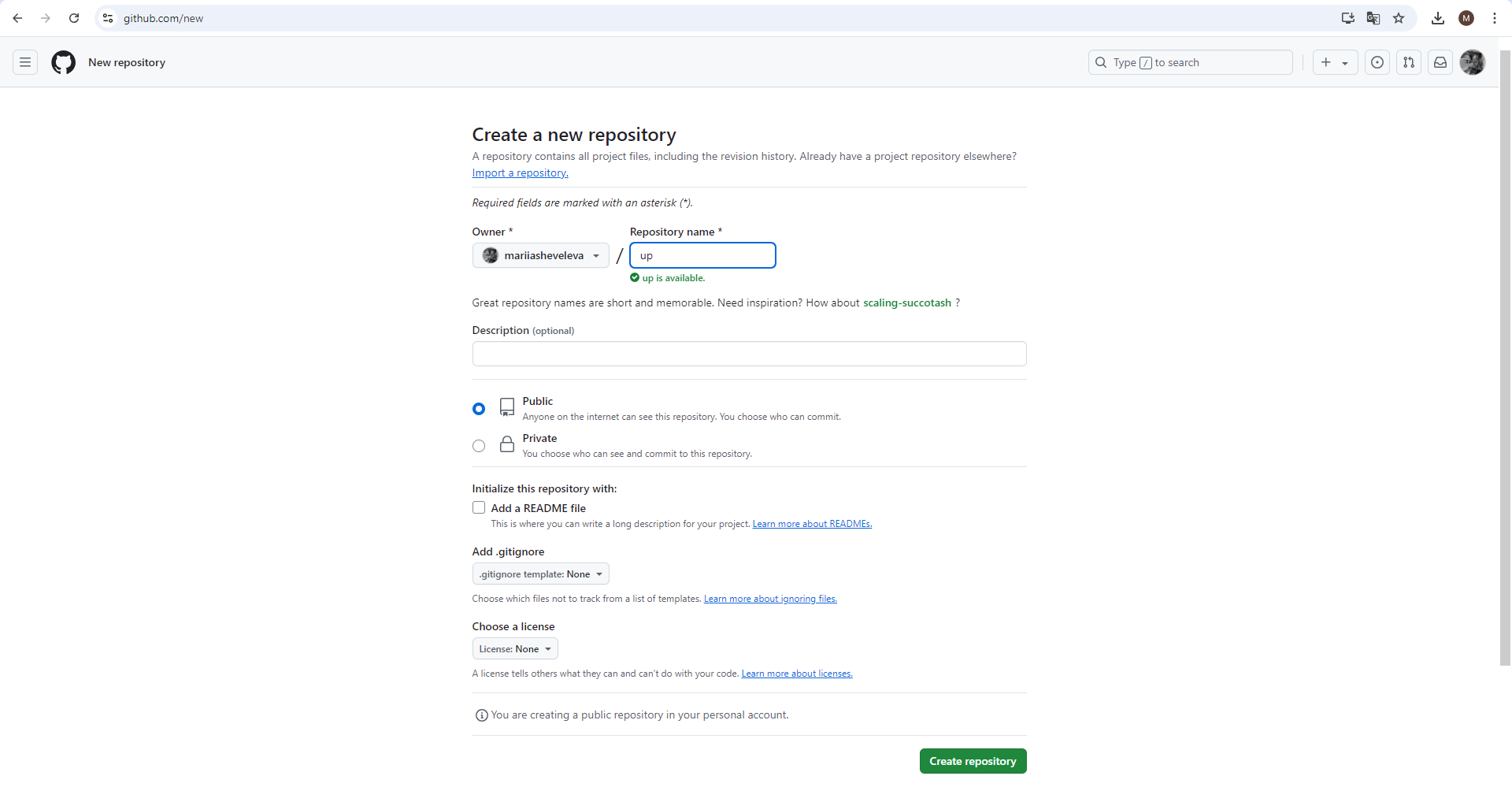


Рисунок 34 – Создание репозитория

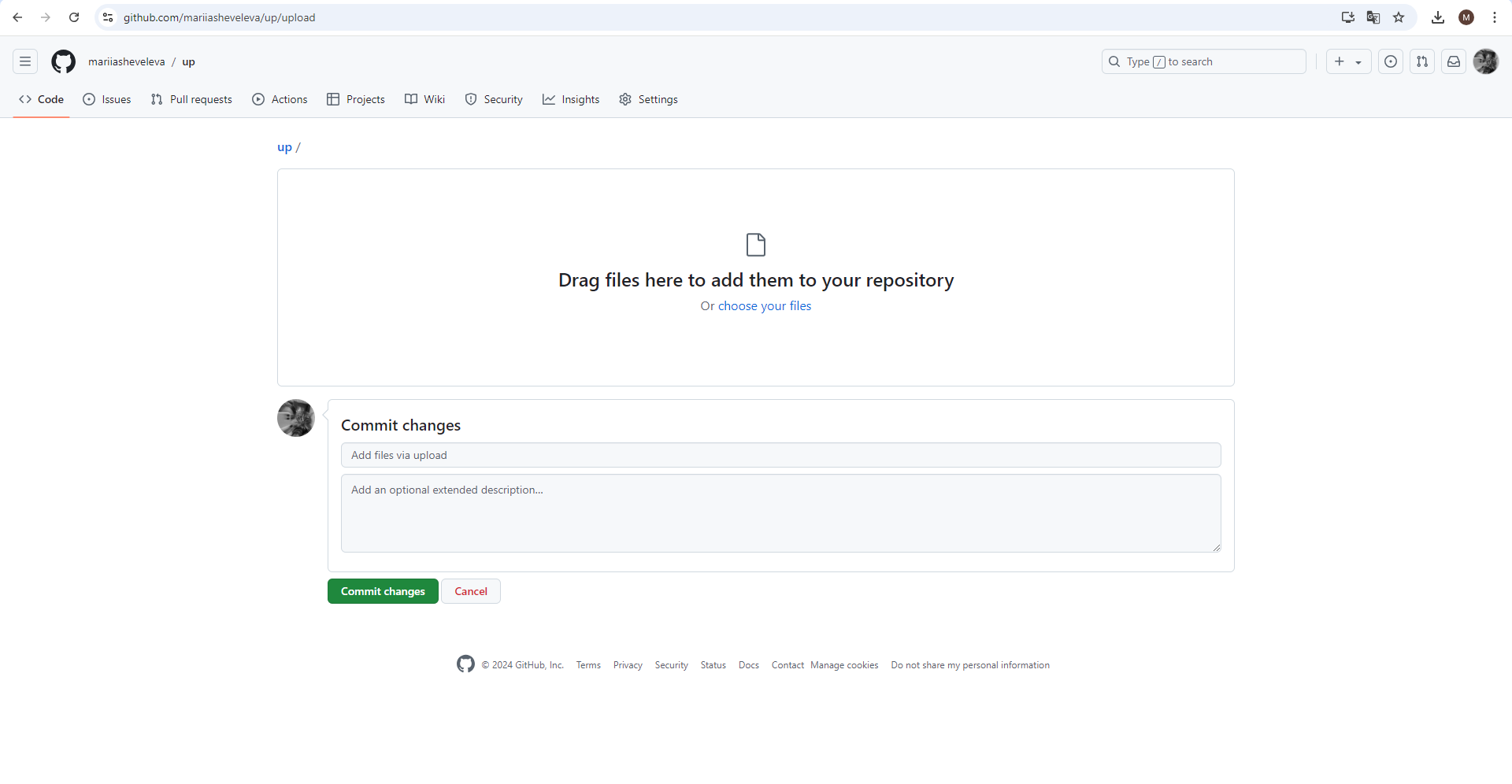


Рисунок 35 – Загрузка файлов

ЗАКЛЮЧЕНИЕ

Учебная практика, проведенная в рамках данного курса, оказалась чрезвычайно полезной и обогащающей. В ходе практики я получила уникальную возможность применить теоретические знания на практике, пройдя через все этапы разработки программного обеспечения. Это включало участие в выработке требований, проектирование архитектуры системы с использованием UML-диаграмм, создание интуитивно понятного пользовательского интерфейса и разработку отдельных модулей программного обеспечения.

Результатом моей работы стало создание полноценного программного продукта, который сопровождался подробной технологической документацией. Этот опыт позволил мне не только закрепить полученные знания, но и приобрести ценные практические навыки, необходимые для успешной работы в области разработки программного обеспечения.

Данная учебная практика имела для меня как теоретическое, так и практическое значение. Она способствовала моему всестороннему развитию и подготовке к профессиональной деятельности в сфере информационных технологий. Я уверена, что полученные знания и навыки будут полезны в моей будущей карьере и помогут мне эффективно решать задачи, связанные с разработкой программного обеспечения.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. https://learn.microsoft.com/ru-ru/visualstudio/ide/create-csharp-winform-visual-studio?view=vs-2022
2. https://metanit.com/sharp/windowsforms/1.1.php
3. «Разработка Windows приложений на C#», Н.А. Осипов 2012
4. «Программирование для Microsoft Windows на С#. В 2-х томах», Чарльз Петцольд 2002 г.
5. «Разработка на языке C# приложений с графическим интерфейсом», Мурадханов С. Э. 2019

Приложение А

Листинг кода:

//форма входа

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace курсовая\_трзбд

{

public partial class Form1 : Form

{

string connectionString = @" Data Source= ADCLG1; Initial catalog=!!!ShevelevaPr; Integrated Security=True";

public Form1()

{

InitializeComponent();

this.FormBorderStyle = FormBorderStyle.FixedSingle; // или FormBorderStyle.Fixed3D;

this.AutoScaleMode = AutoScaleMode.Font;

this.AutoSize = true;

// Устанавливаем свойство UseSystemPasswordChar для textBox2 после инициализации компонентов

textBox2.UseSystemPasswordChar = true;

}

private void button1\_Click(object sender, EventArgs e)

{

string login = textBox1.Text.Trim();

string password = textBox2.Text.Trim(); // Предположим, пароль вводится в textBox2

if (!string.IsNullOrEmpty(login))

{

string query = "SELECT Name, role FROM Paroli WHERE login = @Login AND password = @Password";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(query, connection);

command.Parameters.AddWithValue("@Login", login);

command.Parameters.AddWithValue("@Password", password); // Добавляем параметр для пароля

SqlDataReader reader = command.ExecuteReader();

if (reader.Read())

{

string userName = reader["Name"].ToString();

string userRole = reader["Role"].ToString();

if (userRole == "admin")

{

MessageBox.Show("Данные введены верно. Добро пожаловать!");

Form2 form2 = new Form2();

form2.Show();

this.Hide();

}

else if (userRole == "user")

{

MessageBox.Show("Данные введены верно. Добро пожаловать!");

Form3 form3 = new Form3();

form3.Show();

this.Hide();

}

else if (userRole == "sotr")

{

MessageBox.Show("Данные введены верно. Добро пожаловать!");

Form10 form10 = new Form10();

form10.Show();

this.Hide();

}

else

{

MessageBox.Show("У вас нет доступа к системе.");

}

}

else

{

// Проверка на неверный пароль при верном логине

MessageBox.Show("Вы ввели неверные данные");

}

connection.Close();

}

}

}

private void button2\_Click(object sender, EventArgs e)

{

string login = textBox1.Text.Trim();

string password = textBox2.Text.Trim();

string role = "user"; // Предполагая, что регистрируемый пользователь по умолчанию имеет роль "user"

if (!string.IsNullOrEmpty(login) && !string.IsNullOrEmpty(password))

{

string query = "INSERT INTO Users (login, password, role) VALUES (@Login, @Password, @Role)";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(query, connection);

command.Parameters.AddWithValue("@Login", login);

command.Parameters.AddWithValue("@Password", password);

command.Parameters.AddWithValue("@Role", role);

int rowsAffected = command.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Пользователь успешно зарегистрирован!");

}

else

{

MessageBox.Show("Ошибка при регистрации пользователя. Попробуйте еще раз.");

}

connection.Close();

}

}

else

{

MessageBox.Show("Пожалуйста, заполните все поля для регистрации.");

}

}

}

}

//форма с таблицами

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace курсовая\_трзбд

{

public partial class Form2 : Form

{

string connectionString = @"Data Source= ADCLG1; Initial catalog=!!!ShevelevaPr; Integrated Security=True";

public Form2()

{

InitializeComponent();

LoadData();

this.FormBorderStyle = FormBorderStyle.FixedSingle; // или FormBorderStyle.Fixed3D;

this.AutoScaleMode = AutoScaleMode.Font;

this.AutoSize = true;

}

private void LoadData()

{

string query = "SELECT \* FROM services ";

using (SqlConnection connection = new SqlConnection(connectionString))

{

SqlDataAdapter adapter = new SqlDataAdapter(query, connection);

DataTable table = new DataTable();

adapter.Fill(table);

dataGridView1.DataSource = table;

}

}

private void button1\_Click(object sender, EventArgs e)

{

string query = "INSERT INTO services (service\_id, name)" +

"VALUES (@service\_id, @name)";

using (SqlConnection connection = new SqlConnection(connectionString))

{

using (SqlCommand command = new SqlCommand(query, connection))

{

command.Parameters.AddWithValue("service\_id", Int32.Parse(textBox1.Text));

command.Parameters.AddWithValue("@name", textBox2.Text);

connection.Open();

command.ExecuteNonQuery();

}

}

LoadData();

}

private void button2\_Click(object sender, EventArgs e)

{

string query = "UPDATE services SET name = @name WHERE service\_id = @service\_id";

using (SqlConnection connection = new SqlConnection(connectionString))

{

using (SqlCommand command = new SqlCommand(query, connection))

{

command.Parameters.AddWithValue("service\_id", Int32.Parse(textBox1.Text));

command.Parameters.AddWithValue("@name", textBox2.Text);

connection.Open();

command.ExecuteNonQuery();

}

}

LoadData();

}

private void button3\_Click(object sender, EventArgs e)

{

string query = "DELETE FROM services WHERE service\_id = @service\_id";

using (SqlConnection connection = new SqlConnection(connectionString))

{

using (SqlCommand command = new SqlCommand(query, connection))

{

command.Parameters.AddWithValue("@service\_id", Int32.Parse(textBox1.Text));

connection.Open();

command.ExecuteNonQuery();

}

}

LoadData();

}

private void button4\_Click(object sender, EventArgs e)

{

LoadData();

}

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

{

Form4 form = new Form4();

form.Show();

this.Hide();

}

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

{

Form5 form = new Form5();

form.Show();

this.Hide();

}

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

{

Form6 form = new Form6();

form.Show();

this.Hide();

}

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

{

Form7 form = new Form7();

form.Show();

this.Hide();

}

}

}

//форма с поиском

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace курсовая\_трзбд

{

public partial class Form10 : Form

{

string connectionString = @"Data Source= ADCLG1; Initial catalog=!!!ShevelevaPr; Integrated Security=True";

public Form10()

{

InitializeComponent();

LoadData();

this.FormBorderStyle = FormBorderStyle.FixedSingle; // или FormBorderStyle.Fixed3D;

this.AutoScaleMode = AutoScaleMode.Font;

this.AutoSize = true;

}

private void LoadData()

{

string query = "SELECT \* FROM Zakaz ";

using (SqlConnection connection = new SqlConnection(connectionString))

{

SqlDataAdapter adapter = new SqlDataAdapter(query, connection);

DataTable table = new DataTable();

adapter.Fill(table);

dataGridView1.DataSource = table;

}

}

private void button1\_Click(object sender, EventArgs e)

{

LoadData();

}

private void button2\_Click(object sender, EventArgs e)

{

string query = "SELECT \* FROM Zakaz WHERE ID = @ID";

using (SqlConnection connection = new SqlConnection(connectionString))

{

using (SqlCommand command = new SqlCommand(query, connection))

{

command.Parameters.AddWithValue("@ID", Int32.Parse(textBox1.Text));

SqlDataAdapter adapter = new SqlDataAdapter(command);

DataTable table = new DataTable();

adapter.Fill(table);

dataGridView1.DataSource = table;

}

}

}

}

}

Приложение Б

using Microsoft.VisualStudio.TestTools.UnitTesting;

using System;

using System.Windows.Forms;

using курсовая\_трзбд; // Используйте имя вашего основного проекта

namespace UnitTestProject1

{

[TestClass]

public class UnitTest1

{

private Form1 form;

[TestInitialize]

public void Setup()

{

form = new Form1();

}

[TestMethod]

public void TestMethod1\_ValidLogin()

{

// Arrange

string login = "risha";

string password = "87654321";

// Act

form.Login(login, password);

// Assert

Assert.IsTrue(form.IsLoggedIn, "Форма сотрудника не открылась.");

}

[TestMethod]

public void TestMethod2\_LoadClientData()

{

// Arrange

string login = "klusha";

string password = "88005553533";

// Act

form.Login(login, password);

// Assert

Assert.IsTrue(form.IsClientDataLoaded, "Данные клиента не загружены.");

}

[TestMethod]

public void TestMethod3\_InvalidLogin()

{

// Arrange

string login = "shemaa";

string password = "12345678";

// Act

form.Login(login, password);

// Assert

Assert.IsTrue(form.IsErrorMessageShown, "Окно с предупреждением не появилось.");

}

[TestMethod]

public void TestMethod4\_InvalidPassword()

{

// Arrange

string login = "shema";

string password = "87654321";

// Act

form.Login(login, password);

// Assert

Assert.IsTrue(form.IsErrorMessageShown, "Окно с предупреждением не появилось.");

}

[TestMethod]

public void TestMethod5\_IncompleteData()

{

// Arrange

string login = "shema";

string password = "12345678";

// Act

form.Login(login, password);

// Assert

Assert.IsTrue(form.IsErrorMessageShown, "Окно с предупреждением не появилось.");

}

}

}

using Microsoft.VisualStudio.TestTools.UnitTesting;

using System;

using System.Data;

using System.Windows.Forms;

using курсовая\_трзбд;

namespace UnitTestProject2

{

[TestClass]

public class UnitTest1

{

private Form4 form;

[TestInitialize]

public void Setup()

{

form = new Form4();

}

[TestMethod]

public void AddClient\_ValidData\_ClientAdded()

{

form.AddClient("1", "John Doe", "1234567890");

Assert.IsTrue(form.IsClientAdded);

}

[TestMethod]

public void AddClient\_InvalidData\_ErrorMessageShown()

{

form.AddClient("", "John Doe", "1234567890");

Assert.IsTrue(form.IsErrorMessageShown);

}

[TestMethod]

public void UpdateClient\_ValidData\_ClientUpdated()

{

form.AddClient("2", "Jane Doe", "0987654321"); // Ensure client exists first

form.UpdateClient("2", "Jane Smith", "0987654321");

Assert.IsTrue(form.IsClientUpdated);

}

[TestMethod]

public void UpdateClient\_InvalidData\_ErrorMessageShown()

{

form.UpdateClient("", "Jane Smith", "0987654321");

Assert.IsTrue(form.IsErrorMessageShown);

}

[TestMethod]

public void DeleteClient\_InvalidId\_ErrorMessageShown()

{

form.DeleteClient("");

Assert.IsTrue(form.IsErrorMessageShown);

}

}

}