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)

Студент(ка) 4 курса 42919/3 группы

Туманов Иван Андреевич

(Фамилия, имя, отчество)

Место прохождения практики: УВЦ, пр. Энгельса, 23

(наименование и адрес организации)

Период прохождения практики

с «20» февраля 2023 г. по «18» марта 2023 г.

Руководитель практики Кузнецов К.С.

(подпись) (расшифровка подписи)

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

Санкт-Петербург

2023

**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)

Студент(ка) 4 курса 42919/3 группы

Туманов Иван Андреевич

(Фамилия, имя, отчество)

Место прохождения практики:УВЦ, пр. Энгельса, 23

(наименование и адрес организации)

Период прохождения практики

с «20» февраля 2023 г. по «18» марта 2023 г.

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

* Участие в выработке требований к программному обеспечению;
* Стадии проектирования программного обеспечения;
* Разработка модулей программного обеспечения;
* Тестирование программных модулей и их интеграции;
* Разработка программной документации и стандарты кодирования.

**Индивидуальное задание: ВАРИАНТ 19**

Задание выдал «20» февраля 2023 г. \_\_\_\_\_\_\_\_\_\_\_\_\_ Кузнецов К.С.

(подпись) (Ф.И.О.)

Задание получил «20» февраля 2023 г. \_\_\_\_\_\_\_\_\_\_\_\_ Туманов И.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)

Студент(ка) 4 курса 42919/3 группы

Туманов Иван Андреевич

(Фамилия, имя, отчество)

Место прохождения практики:УВЦ, пр. Энгельса, 23

(наименование и адрес организации)

Период прохождения практики

с «20» февраля 2023 г. по «18» марта 2023 г.

Руководитель практики Кузнецов К.С.

(подпись) (расшифровка подписи)

Итоговая оценка по практике

Санкт-Петербург

2023

**Содержание дневника**

|  |  |  |
| --- | --- | --- |
| **Дата** | **Виды выполненных работ и заданий по программе практики** | **Подпись руководителя практики** |
| **1** | **2** | **3** |
| 20.02.23 | Анализ предметной области |  |
| 21.02.23 | Выявление требований к программе |  |
| 22.02.23 | Разработка технического задания |  |
| 25.02.23 | UML. Проектирование диаграммы вариантов использования |  |
| 27.02.23 | UML. Проектирование диаграммы последовательности |  |
| 28.02.23 | UML. Проектирование диаграммы активности |  |
| 01.03.23 | Моделирование структуры ПО |  |
| 02.03.23 | Проектирование инфологической и даталогической модели данных |  |
| 03.03.23 | Проектирование интерфейса пользователя. Создание Wireframe эскизов |  |
| 04.03.23 | Разработка дизайна программы в соответствии с руководством по стилю |  |
| 06.03.23 | Разработка базы данных |  |
| 07.03.23 | Разработка словаря данных |  |
| 09.03.23 | Создание приложения. Форма авторизации |  |
| 10.03.23 | Создание приложения. Форма заказов |  |
| 11.03.23 | Создание приложения. Основные формы приложения |  |
| 13.03.23 | Разработка библиотеки классов |  |
| 14.03.23 | Подготовка отчетов и выгрузка документов для печати |  |
| 15.03.23 | Отладка программных модулей |  |
| 16.03.23 | Модульное тестирование |  |
| 16.03.23 | Создание тестовых случаев |  |
| 16.03.23 | Интеграционное тестирование |  |
| 16.03.23 | Разработка самодокументирующегося кода |  |
| 16.03.23 | Инспекция кода на соответствие стандартам кодирования |  |
| 18.03.23 | Подготовка отчета и размещение результатов в репозитории контроля версий |  |

**АТТЕСТАЦИОННЫЙ ЛИСТ**

**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)

Студент(ка) 4 курса 42919/3 группы

Туманов Иван Андреевич

(Фамилия, имя, отчество)

Место прохождения практики:УВЦ, пр. Энгельса, 23

(наименование и адрес организации)

Период прохождения практики

с «20» февраля 2023г. по «18» марта 2023 г.

**Виды и качество выполнения работ**

|  |  |  |
| --- | --- | --- |
| **Виды выполненных работ обучающимся**  **во время практики** | **Объем работ, час.** | **Качество выполнения работ (оценка по пятибалльной системе)** |
| Участие в выработке требований к программному обеспечению | 18 |  |
| Стадии проектирования программного обеспечения | 42 |  |
| Разработка модулей программного обеспечения | 48 |  |
| Тестирование программных модулей и их интеграции | 18 |  |
| Разработка программной документации и стандарты кодирования | 18 |  |

**Характеристика учебной/профессиональной деятельности обучающегося во время учебной практики (по профилю специальности):**

Общие и профессиональные компетенции, предусмотренные программой практики, освоены**/**не освоены.

(нужное подчеркнуть)

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

Руководитель практики Кузнецов К.С. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Ф.И.О.) (подпись)

Дата «18» марта 2023 г.

СОДЕРЖАНИЕ

[1 Проектирование системы 7](#_Toc129857464)

[1.1 Диаграмма вариантов использования 7](#_Toc129857465)

[1.2 Диаграмма активности 8](#_Toc129857466)

[1.3 Диаграмма последовательности 9](#_Toc129857467)

[1.4 ER-Диаграмма 9](#_Toc129857468)

[1.5 Словарь данных 10](#_Toc129857469)

[2 Создание базы данных 13](#_Toc129857470)

[3 Разработка клиентского приложения 14](#_Toc129857471)

[4 Тестирование приложения 19](#_Toc129857472)

[4.1 Разработка тест-кейсов 19](#_Toc129857473)

[Приложение А 24](#_Toc129857474)

[Приложение Б 28](#_Toc129857475)

# 1 Проектирование системы

## 1.1 Диаграмма вариантов использования

На диаграмме вариантов использования показаны акторы (директор, менеджер, бухгалтер, администратор) и действия, которые они могут делать при помощи информационной системы (рисунок 1).

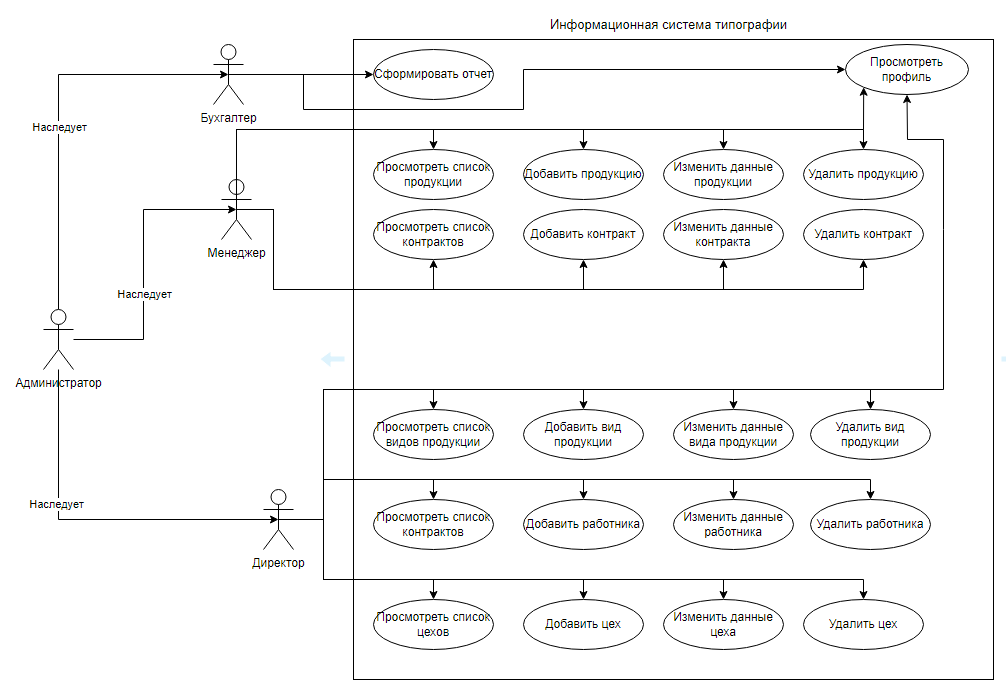


Рисунок 1 – Диаграмма вариантов использования

## 1.2 Диаграмма активности

Диаграмма активности демонстрирует взаимодействие менеджера, приложения и базы данных во время создания контракта (рисунок 2).

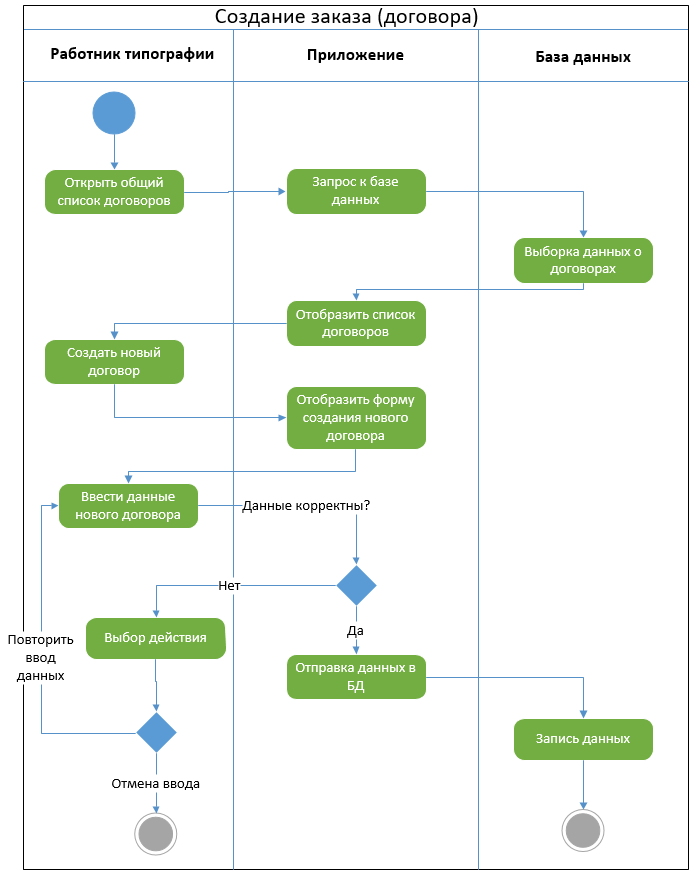


Рисунок 2 – Диаграмма активности

## 1.3 Диаграмма последовательности

На рисунке 3 представлена диаграмма последовательности, показывающая процесс авторизации пользователя в системе.

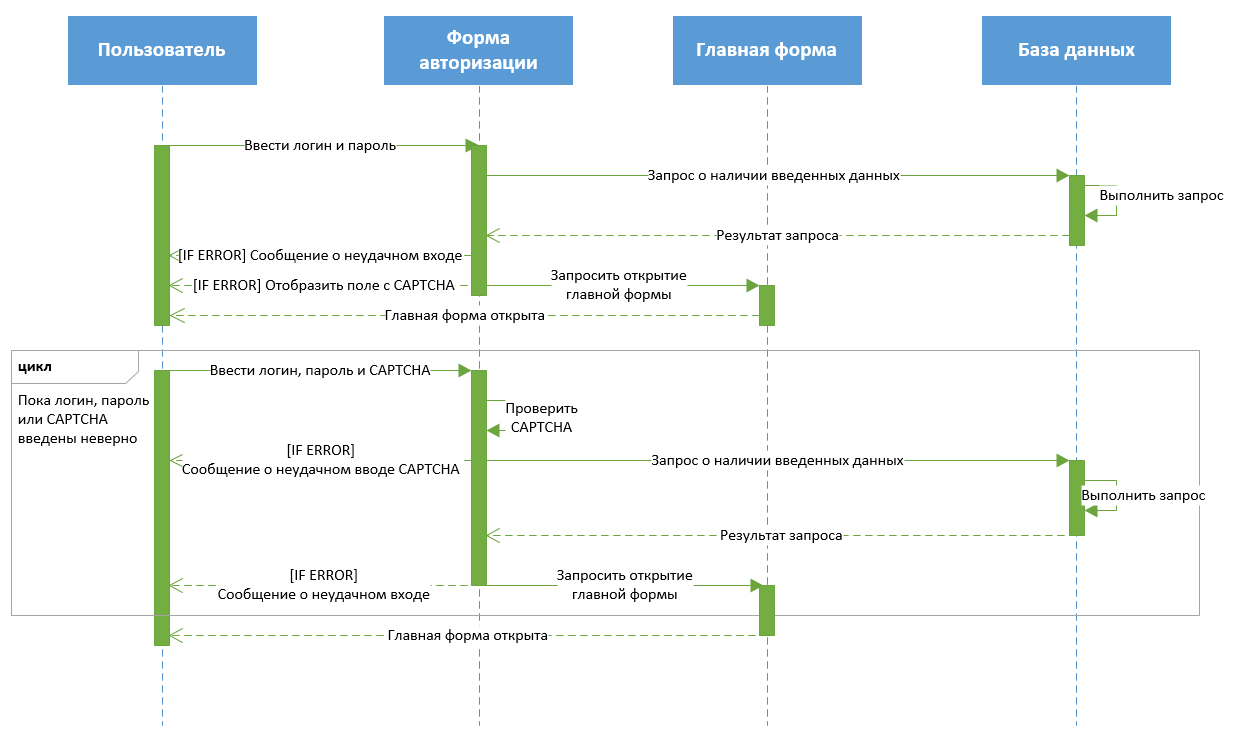


Рисунок 3 – Диаграмма последовательности

## 1.4 ER-Диаграмма

Схема базы данных представлена на рисунке 4 в виде ER-диаграммы.

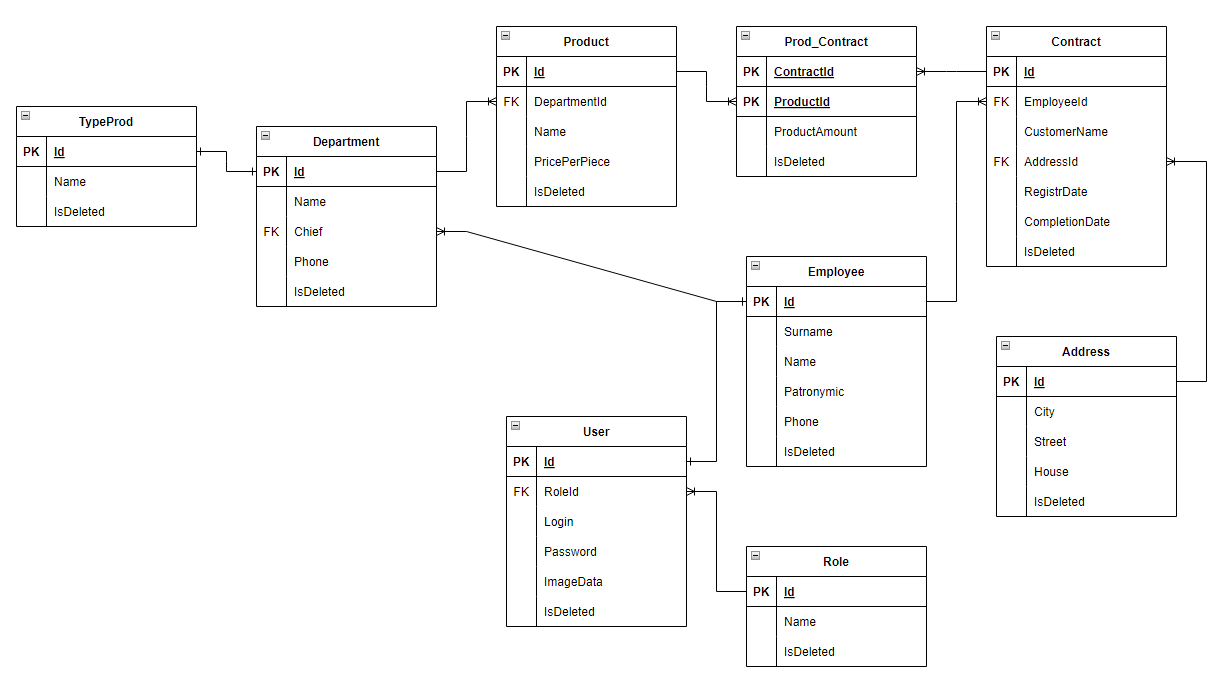


Рисунок 4 – ER-диаграмма

## 1.5 Словарь данных

В таблице 1 приведен словарь данных для каждой таблицы из схемы базы данных.

Таблица 1 – Словарь данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Department** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | Id | INTEGER | Y | NOT NULL |
|  | Name | NVARCHAR(MAX) | Y | UNIQUE, NOT NULL |
| FK | Chief | INTEGER | Y | NOT NULL |
|  | Phone | NVARCHAR(16) | Y | UNIQUE, NOT NULL  Length <=16 |
|  | IsDeleted | BIT | Y | NOT NULL |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | Id | INTEGER | Y | NOT NULL |
|  | Name | NVARCHAR(MAX) | Y | UNIQUE, NOT NULL |
| FK | Chief | INTEGER | Y | NOT NULL |
|  | Phone | NVARCHAR(16) | Y | UNIQUE, NOT NULL  Length <=16 |
|  | IsDeleted | BIT | Y | NOT NULL |
|  |  |  |  |  |
| **Product** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | Id | INTEGER | Y | Auto Increment, NOT NULL |
| FK | DepartmentId | INTEGER | Y | NOT NULL |
|  | Name | NVARCHAR(MAX) | Y | NOT NULL |
|  | PricePerPiece | Decimal(32, 2) | Y | PricePerPiece > 0.0, NOT NULL, Length <=32 |
|  | IsDeleted | BIT | Y | NOT NULL |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | Id | INTEGER | Y | Auto Increment, NOT NULL |
| FK | DepartmentId | INTEGER | Y | NOT NULL |
|  | Name | NVARCHAR(MAX) | Y | NOT NULL |
|  | PricePerPiece | Decimal(32, 2) | Y | PricePerPiece > 0.0, NOT NULL, Length <=32 |
|  | IsDeleted | BIT | Y | NOT NULL |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Order** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK, FK | ProductId | INTEGER | Y | NOT NULL |
| PK, FK | ContractId | INTEGER | Y | NOT NULL |
|  | ProductAmount | INTEGER | Y | ProductAmount > 0, NOT NULL |
|  | IsDeleted | BIT | Y | NOT NULL |
|  |  |  |  |  |
| **Employee** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | Id | INTEGER | Y | Auto Increment, NOT NULL |
|  | Surname | NVARCHAR(MAX) | Y | NOT NULL |
|  | Name | NVARCHAR(MAX) | Y | NOT NULL |
|  | Patronymic | NVARCHAR(MAX) | Y | NOT NULL |
|  | Phone | NVARCHAR(16) | Y | UNIQUE, NOT NULL Length <=16 |
|  | IsDeleted | BIT | Y | NOT NULL |
|  |  |  |  |  |
| **Contract** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | Id | INTEGER | Y | Auto Increment, NOT NULL |
| FK | EmployeeId | INTEGER | Y | NOT NULL |
|  | CustomerName | NVARCHAR(MAX) | Y | NOT NULL |
|  | AddressId | INTEGER | Y | NOT NULL |
|  | RegistrDate | DATETIME | Y | NOT NULL |
|  | CompletionDate | DATETIME | Y | NOT NULL |
|  | IsDeleted | BIT | Y | NOT NULL |
|  |  |  |  |  |
| **User** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | Id | INTEGER | Y | NOT NULL |
| FK | RoleId | INTEGER | Y | NOT NULL |
|  | Login | NVARCHAR(256) | Y | NOT NULL, Length <=256 |
|  | Password | NVARCHAR(MAX) | Y | NOT NULL |
|  | Image | VARBINARY(MAX) | Y | NOT NULL |
|  | IsDeleted | BIT | Y | NOT NULL |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Role** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | Id | INTEGER | Y | NOT NULL |
|  | Name | NVARCHAR(MAX) | Y | NOT NULL |
|  | IsDeleted | BIT | Y | NOT NULL |
|  |  |  |  |  |
| **Address** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | AddressId | INTEGER | Y | Auto Increment, NOT NULL |
|  | City | NVARCHAR(MAX) | Y | NOT NULL |
|  | Street | NVARCHAR(MAX) | Y | NOT NULL |
|  | House | NVARCHAR(MAX) | Y | NOT NULL |
|  | IsDeleted | BIT | Y | NOT NULL |
|  |  |  |  |  |
| **TypeProd** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| PK | TypeId | INTEGER | Y | NOT NULL |
|  | Name | NVARCHAR(MAX) | Y | NOT NULL |
|  | IsDeleted | BIT | Y | NOT NULL |

# 2 Создание базы данных

База данных была создана при помощи Microsoft SQL Server 2022 и Microsoft SQL Server Management Studio 2018. Код создания таблиц, представлений, заполнения таблиц данными приведён в приложении А. Диаграмма базы данных, созданная и хранящаяся на сервере, приведена на рисунке 5.

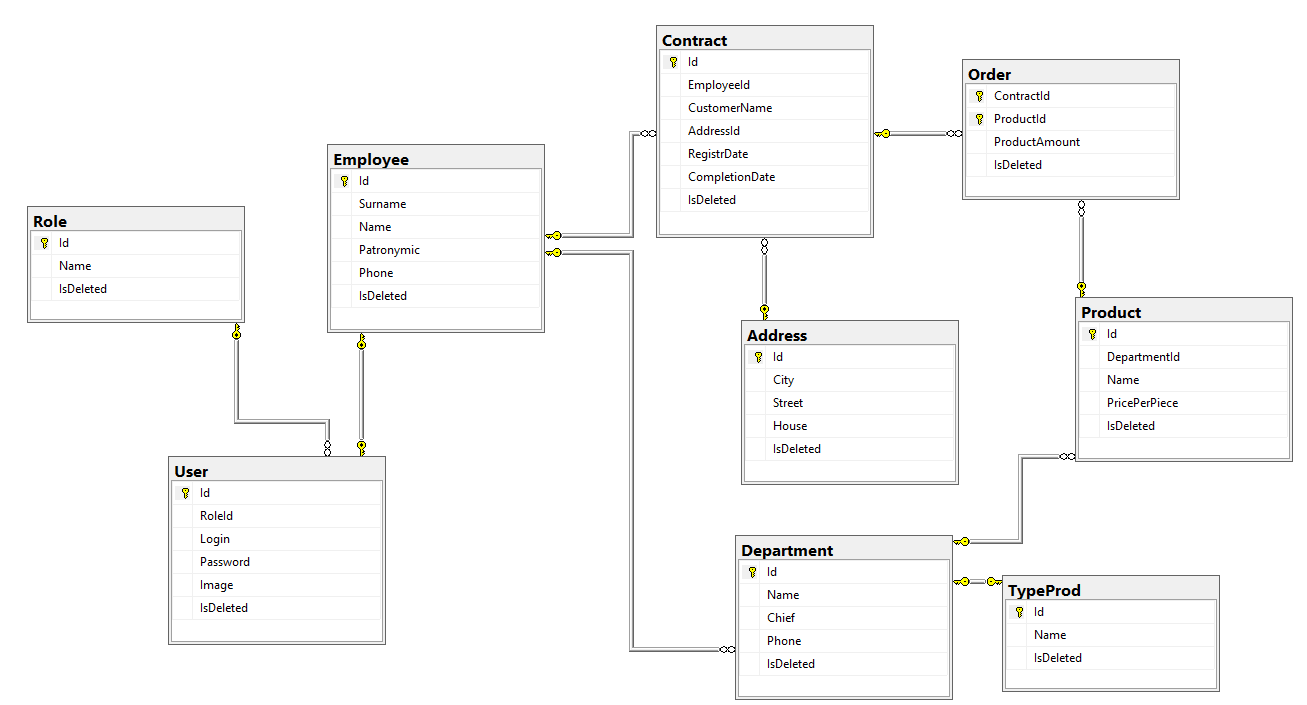


Рисунок 5 – Диаграмма базы данных

# 3 Разработка клиентского приложения

Исходный код клиентского приложения находится в приложении Б, а также в репозитории GitHub (ссылка: https://github.com/Fazer777/PrintingHouseApplication).

Снимки основных форм приведены на рисунках 6-15.

Начальная форма авторизации представлена на рисунке 6.

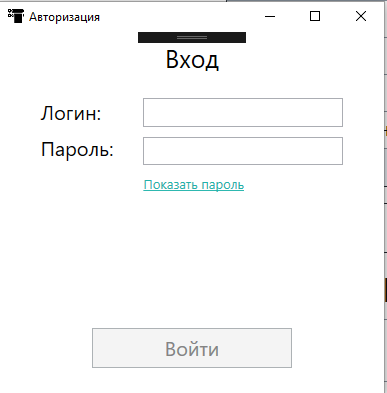


Рисунок 6 – Форма авторизации (без captcha)

На рисунке 7 представлена форма авторизации с капчей.

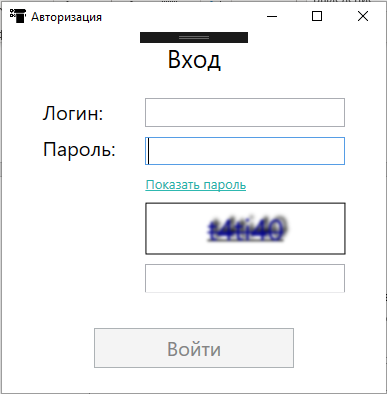


Рисунок 7 – Форма авторизации (с captcha)

На рисунке 8 представлен модуль директора с доступом к таблицам: сотрудники, цеха типографии, виды продукции.

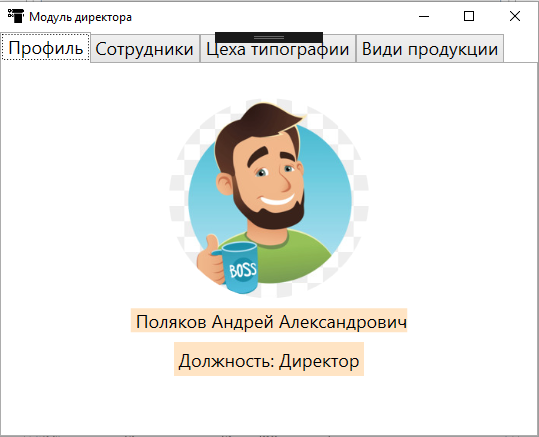


Рисунок 8 – Экранная форма директора

На рисунке 9 представлена форма директора с открытой таблицей «Цеха типографии».

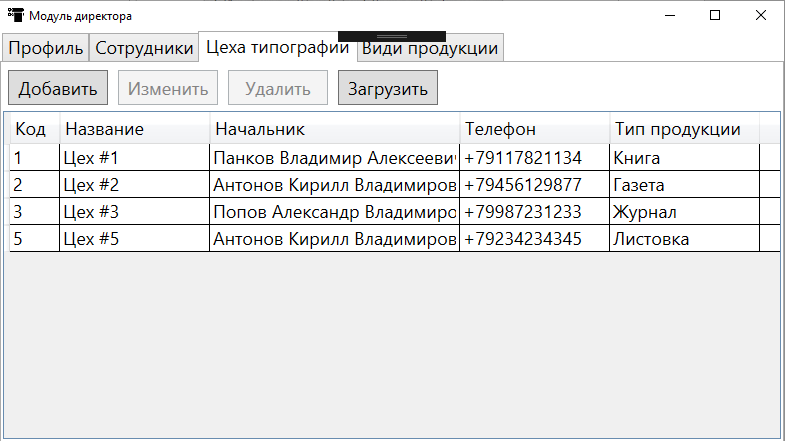


Рисунок 9 – Экранная форма таблицы цехов производства

На рисунке 10 представлена форма менеджера с доступом к таблицам: продукция, контракты.

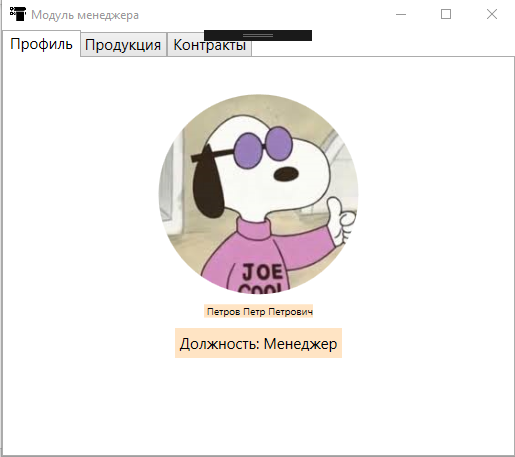


Рисунок 10 – Экранная форма менеджера

На рисунке 11 представлена форма менеджера с открытой таблицей «Контракты».

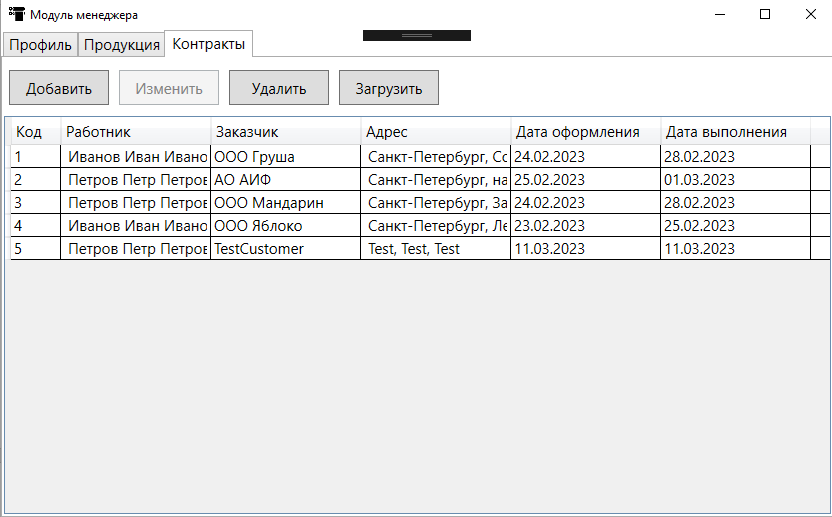


Рисунок 11 – Экранная форма таблицы контракты

На рисунке 12 представлена форма менеджера с открытой таблицей «Продукция».

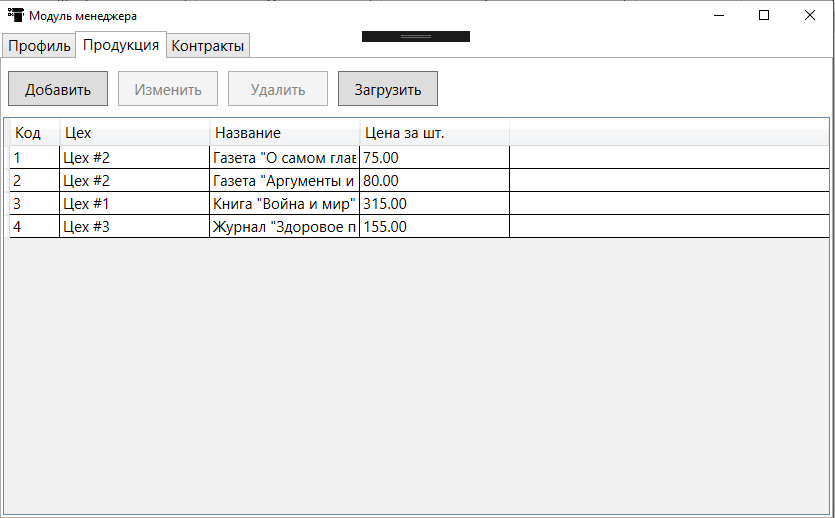


Рисунок 12 – Экранная форма таблицы продукция

На рисунке 13 представлена форма работы с данными для добавления нового контракта.

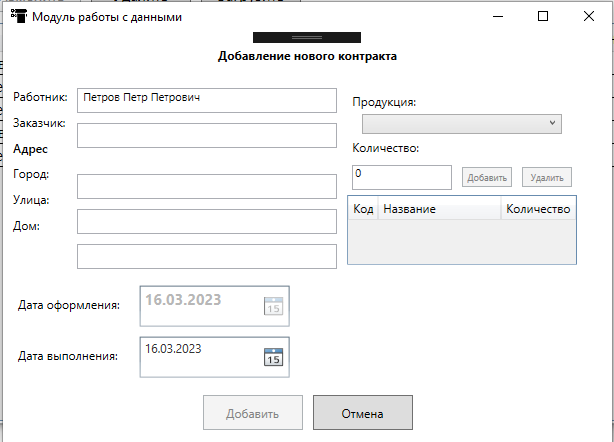


Рисунок 13 – Экранная форма добавления нового контракта

На рисунке 14 представлена форма администратора с доступом ко всем таблицам.

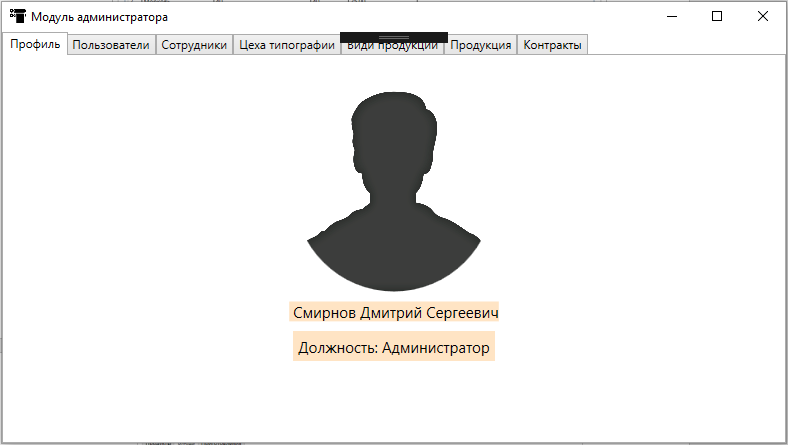


Рисунок 14 – Экранная форма администратора системы

На рисунке 15 представлена форма администратора с открытой таблицей «Пользователи».

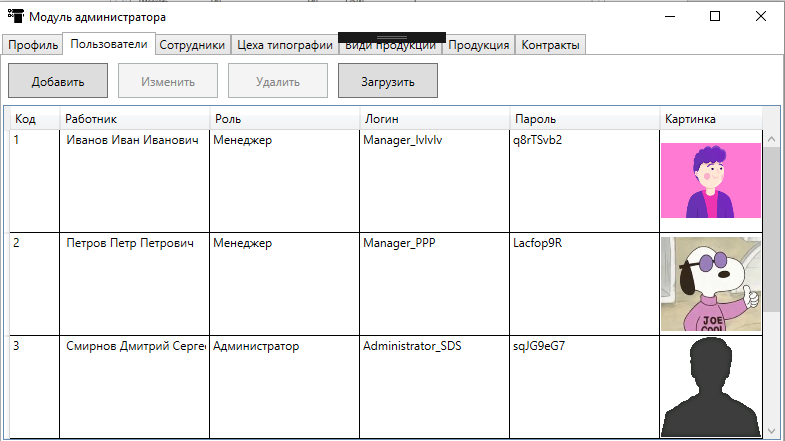


Рисунок 15 – Экранная форма таблицы пользователи

# 4 Тестирование приложения

## 4.1 Разработка тест-кейсов

Тест-кейсы описывают совокупность шагов, конкретных условий и параметров, необходимых для проверки реализации тестируемых функций или их частей.

Разработанные тест-кейсы приведены в таблицах 2-6.

Таблица 2 – Тест-кейс TC\_Add\_Type\_Product\_Positive1

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_Add\_Type\_Product\_Positive1 |
| **Приоритет тестирования** | Средний |
| **Заголовок/название теста** | Добавление нового вида продукции при корректном вводе данных |
| **Краткое изложение теста** | Проверить добавление нового вида продукции при корректном вводе данных |
| **Этапы теста** | 1. Войти в систему как «Директор» 2. Открыть вкладку «Виды продукции» 3. Нажать кнопку «Добавить» 4. Заполнить поля формы 5. Нажать кнопку «Добавить» |
| **Тестовые данные** | Номер: 5  Название: Листовка |
| **Ожидаемый результат** | Вывод сообщения об успешном добалении объекта. |
| **Фактический результат** | Вывод сообщения об успешном добалении объекта. |
| **Статус** | Зачет |
| **Предварительное условие** | Перед добавление нового вида продукции необходимо добавить цех производства.  Номер нового вида продукции должен совпадать с номером цеха производства |
| **Постусловие** |  |
| **Примечания/комментарии** | Для отображение добавленных данных нажать «Загрузить» |

Таблица 3 – Тест-кейс TC\_Add\_Type\_Product\_Negative1

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_Add\_Type\_Product\_Negative1 |
| **Приоритет тестирования** | Средний |
| **Заголовок/название теста** | Добавление нового вида продукции при некорректном вводе номера вида продукции |
| **Краткое изложение теста** | Проверить добавление нового вида продукции при некорректном вводе номера вида продукции |
| **Этапы теста** | 1. Войти в систему как «Директор» 2. Открыть вкладку «Виды продукции» 3. Нажать кнопку «Добавить» 4. Заполнить поля формы 5. Нажать кнопку «Добавить» |
| **Тестовые данные** | Номер: -1  Название: Буклет |
| **Ожидаемый результат** | Вывод сообщения об ошибке при добавлении объекта |
| **Фактический результат** | Вывод сообщения об ошибке при добавлении объекта |
| **Статус** | Зачет |
| **Предварительное условие** | Перед добавление нового вида продукции необходимо добавить цех производства.  Номер нового вида продукции должен совпадать с номером цеха производства |
| **Постусловие** |  |
| **Примечания/комментарии** | Для отображение добавленных данных нажать «Загрузить» |

Таблица 4 – Тест-кейс TC\_Add\_Type\_Product\_Negative2

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_Add\_Type\_Product\_Negative2 |
| **Приоритет тестирования** | Средний |
| **Заголовок/название теста** | Добавление нового вида продукции при некорректном вводе названия вида продукции. |
| **Краткое изложение теста** | Проверить добавление нового вида продукции при некорректном вводе названия вида продукции. |
| **Этапы теста** | 1. Войти в систему как «Директор» 2. Открыть вкладку «Виды продукции» 3. Нажать кнопку «Добавить» 4. Заполнить поля формы 5. Нажать кнопку «Добавить» |
| **Тестовые данные** | Номер: 6  Название: \*Пустая строка\* |
| **Ожидаемый результат** | Вывод сообщения об ошибке при добавлении объекта |
| **Фактический результат** | Вывод сообщения об ошибке при добавлении объекта |
| **Статус** | Зачет |
| **Предварительное условие** | Перед добавление нового вида продукции необходимо добавить цех производства.  Номер нового вида продукции должен совпадать с номером цеха производства |
| **Постусловие** |  |
| **Примечания/комментарии** | Для отображение добавленных данных нажать «Загрузить» |

Таблица 5 – Тест-кейс TC\_Add\_Employee\_Positive1

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_Add\_Employee\_Positive1 |
| **Приоритет тестирования** | Средний |
| **Заголовок/название теста** | Добавление нового сотрудника при корректном вводе данных формы |
| **Краткое изложение теста** | Проверить добавление нового сотрудника при корректном вводе данных формы |
| **Этапы теста** | 1. Войти в систему как «Директор» 2. Открыть вкладку «Сотрудники» 3. Нажать кнопку «Добавить» 4. Заполнить поля формы 5. Нажать кнопку «Добавить» |
| **Тестовые данные** | Фамилия: Александров  Имя: Александр  Отчетство: Александрович  Телефон: +79343457834 |
| **Ожидаемый результат** | Вывод собщения об успешном добавлении объекта |
| **Фактический результат** | Вывод собщения об успешном добавлении объекта |
| **Статус** | Зачет |
| **Предварительное условие** |  |
| **Постусловие** |  |
| **Примечания/комментарии** | Для отображение добавленных данных нажать «Загрузить» |

Таблица 6 – Тест-кейс TC\_Add\_Employee\_Negative1

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_Add\_Employee\_Negative1 |
| **Приоритет тестирования** | Средний |
| **Заголовок/название теста** | Добавление нового сотрудника при некорректном вводе имени сотрудника в форме. |
| **Краткое изложение теста** | Проверить добавление нового сотрудника при некорректном вводе имени сотрудника в форме. |
| **Этапы теста** | 1. Войти в систему как «Директор» 2. Открыть вкладку «Сотрудники» 3. Нажать кнопку «Добавить» 4. Заполнить поля формы 5. Нажать кнопку «Добавить» |
| **Тестовые данные** | Фамилия: \*Пустая строка\*  Имя: \*Пустая строка\*  Отчетство: \*Пустая строка\*  Телефон: +79343457834 |
| **Ожидаемый результат** | Вывод сообщения об ошибке при добавлении объекта |
| **Фактический результат** | Вывод сообщения об ошибке при добавлении объекта |
| **Статус** | Зачет |
| **Предварительное условие** |  |
| **Постусловие** |  |
| **Примечания/комментарии** | Для отображение добавленных данных нажать «Загрузить» |

# Приложение А

Код SQL

**A.1 Запросы на создание таблиц**

Create table Employee

(

Id INT PRIMARY KEY IDENTITY(1,1) NOT NULL,

Surname NVARCHAR(MAX) NOT NULL,

[Name] NVARCHAR(MAX) NOT NULL,

Patronymic NVARCHAR(MAX) NOT NULL,

Phone NVARCHAR(16) NOT NULL UNIQUE,

IsDeleted BIT NOT NULL DEFAULT 0

);

Create table Department

(

Id INT PRIMARY KEY NOT NULL,

[Name] NVARCHAR(256) NOT NULL UNIQUE,

Chief INT NOT NULL FOREIGN KEY REFERENCES Employee(Id) ON DELETE NO ACTION ON UPDATE NO ACTION,

Phone NVARCHAR(16) NOT NULL UNIQUE,

IsDeleted BIT NOT NULL DEFAULT 0

);

Create table Product

(

Id INT PRIMARY KEY IDENTITY(1,1) NOT NULL,

DepartmentId INT NOT NULL FOREIGN KEY REFERENCES Department (Id) ON DELETE NO ACTION ON UPDATE NO ACTION,

[Name] NVARCHAR(MAX) NOT NULL,

PricePerPiece DECIMAL(32,2) NOT NULL CONSTRAINT CK\_Product\_PricePerPiece CHECK(PricePerPiece > 0.0),

IsDeleted BIT NOT NULL DEFAULT 0

);

Create table [Address]

(

Id INT PRIMARY KEY IDENTITY(1,1) NOT NULL,

City NVARCHAR(MAX) NOT NULL,

Street NVARCHAR(MAX) NOT NULL,

House NVARCHAR(MAX) NOT NULL,

IsDeleted BIT NOT NULL DEFAULT 0

);

Create table [Contract]

(

Id INT PRIMARY KEY IDENTITY(1,1) NOT NULL,

EmployeeId INT NOT NULL FOREIGN KEY REFERENCES Employee (Id) ON DELETE NO ACTION ON UPDATE NO ACTION,

CustomerName NVARCHAR(MAX) NOT NULL,

AddressId INT NOT NULL FOREIGN KEY REFERENCES [Address] (Id) ON DELETE NO ACTION ON UPDATE NO ACTION,

RegistrDate DATETIME NOT NULL,

CompletionDate DATETIME NOT NULL,

IsDeleted BIT NOT NULL DEFAULT 0

);

Create table [Order]

(

ContractId INT NOT NULL,

ProductId INT NOT NULL,

ProductAmount INT NOT NULL

CONSTRAINT CK\_Order\_ProductAmount CHECK(ProductAmount > 0),

IsDeleted BIT NOT NULL DEFAULT 0,

PRIMARY KEY (ContractId, ProductId),

FOREIGN KEY (ContractId) REFERENCES [Contract] (Id) ON DELETE NO ACTION ON UPDATE NO ACTION,

FOREIGN KEY (ProductId) REFERENCES [Product] (Id) ON DELETE NO ACTION ON UPDATE NO ACTION

);

Create table [Role]

(

Id INT PRIMARY KEY NOT NULL,

[Name] NVARCHAR(MAX) NOT NULL,

IsDeleted BIT NOT NULL DEFAULT 0

);

Create table [User]

(

Id INT PRIMARY KEY NOT NULL,

RoleId INT NOT NULL,

[Login] NVARCHAR(256) UNIQUE NOT NULL,

[Password] NVARCHAR(MAX) NOT NULL,

[Image] VARBINARY(MAX) NOT NULL,

IsDeleted BIT NOT NULL DEFAULT 0,

FOREIGN KEY (Id) REFERENCES Employee (Id) ON DELETE NO ACTION ON UPDATE NO ACTION,

FOREIGN KEY (RoleId) REFERENCES [Role] (Id) ON DELETE NO ACTION ON UPDATE NO ACTION

);

Create table TypeProd

(

Id INT PRIMARY KEY NOT NULL,

[Name] NVARCHAR(MAX) NOT NULL,

IsDeleted BIT NOT NULL DEFAULT 0,

FOREIGN KEY (Id) REFERENCES Department (Id) ON DELETE NO ACTION ON UPDATE NO ACTION

);

**А.2 Запросы на заполнение таблиц данными**

INSERT INTO Employee (Surname, [Name], Patronymic, Phone)

VALUES

('Иванов','Иван','Иванович','+79067845632'),

('Петров','Петр','Петрович','+79113452344'),

('Смирнов','Дмитрий','Сергеевич','+79064534523'),

('Комаров','Валентин','Андреевич','+79112895721'),

('Поляков','Андрей','Александрович','+79062854671'),

('Панков','Владимир','Алексеевич','+79114217036'),

('Антонов','Кирилл','Владимирович','+79061234567'),

('Попов','Александр','Владимирович','+79062731425');

INSERT INTO Department (Id, [Name], Chief, Phone)

VALUES

(1, 'Цех #1', 6, '+79117821134'),

(2, 'Цех #2', 7, '+79456129877'),

(3, 'Цех #3', 8, '+79987231233');

INSERT INTO TypeProd (Id,[Name])

VALUES

(1, 'Книга'),

(2, 'Газета'),

(3, 'Журнал');

INSERT INTO Product (DepartmentId,[Name], PricePerPiece)

VALUES

(2, 'Газета "О самом главном"', 75.00),

(2, 'Газета "Аргументы и факты"', 80.00),

(1, 'Книга "Война и мир"', 315.00),

(3, 'Журнал "Здоровое питание"', 155.00);

INSERT INTO [Address] (City, Street, House)

VALUES

('Санкт-Петербург','Софийская','59'),

('Санкт-Петербург','наб. канала Грибоедова','69'),

('Санкт-Петербург','Загородный проспект','22'),

('Санкт-Петербург', 'Лермонтовский проспект', '30');

INSERT INTO [Contract] (EmployeeId, CustomerName, AddressId, RegistrDate, CompletionDate)

VALUES

(1, 'ООО Груша', 1, '20230224', '20230228'),

(2, 'АО АИФ', 2, '20230225', '20230301'),

(2, 'ООО Мандарин', 3, '20230224', '20230228'),

(1, 'ООО Яблоко', 4, '20230223', '20230225');

INSERT INTO [Order] (ContractId, ProductId, ProductAmount)

VALUES

(1, 1, 100000),

(2, 2, 150000),

(3, 3, 3000),

(4, 4, 50000);

INSERT INTO [Role] (Id, [Name])

VALUES

(1, 'Директор'),

(2, 'Администратор'),

(3, 'Менеждер'),

(4, 'Бухгалтер');

# Приложение Б

Исходный код приложения

**Б.1 App.Config**

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

<configuration>

<startup>

<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.6.1" />

</startup>

<connectionStrings>

<add name="DataContext" connectionString="metadata=res://\*/Model.csdl|res://\*/Model.ssdl|res://\*/Model.msl;provider=System.Data.SqlClient;provider connection string=&quot;data source=ADCLG1;initial catalog=NewPrintingHouse;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework&quot;" providerName="System.Data.EntityClient" />

</connectionStrings>

</configuration>

**Б.2 Command.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Input;

namespace PrintingHouseApplication.MyCommand

{

class Command : ICommand

{

private Action<object> \_execute;

private Func<object, bool> \_canExecute;

public event EventHandler CanExecuteChanged

{

add { CommandManager.RequerySuggested += value; }

remove { CommandManager.RequerySuggested -= value; }

}

public Command(Action<object> execute, Func<object, bool> canExecute)

{

\_execute = execute;

\_canExecute = canExecute;

}

public bool CanExecute(object parameter) => \_canExecute == null || \_canExecute(parameter);

public void Execute(object parameter)

{

\_execute?.Invoke(parameter);

}

}

}

**Б.3 BaseViewModel.cs**

using System.ComponentModel;

using System.Runtime.CompilerServices;

namespace PrintingHouseApplication.ViewModels

{

class BaseViewModel : INotifyPropertyChanged

{

public event PropertyChangedEventHandler PropertyChanged;

public void OnPropertyChanged([CallerMemberName] string propName = null)

{

PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propName));

}

}

}

**Б.4 LoginViewModel.cs**

using System;

using System.Windows;

using System.Linq;

using System.Windows.Threading;

using PrintingHouseApplication.Models;

using PrintingHouseApplication.MyCommand;

using PrintingHouseApplication.Views;

namespace PrintingHouseApplication.ViewModels

{

class LoginViewModel : BaseViewModel

{

private string \_login = string.Empty;

private string \_password = string.Empty;

private bool \_isAccess = false;

private LoginModel \_loginModel = new LoginModel();

private bool \_isError = false;

private string \_captchaInput = string.Empty;

private Captcha \_captcha;

private DispatcherTimer dt = new DispatcherTimer();

private int \_timerValue = 0;

private int \_timeLock = 10;

private bool \_isLock = false;

public bool ShowMaskPass { get; set; } = true;

public bool ShowUnmaskPass { get; set; } = false;

public string CaptchaInput

{

get => \_captchaInput;

set

{

\_captchaInput = value;

OnPropertyChanged(nameof(CaptchaInput));

}

}

public string Captcha { get; set; }

public bool IsError

{

get => \_isError;

set

{

\_isError = value;

OnPropertyChanged(nameof(IsError));

}

}

public string Login

{

get => \_login;

set

{

\_login = value;

OnPropertyChanged(nameof(Login));

}

}

public string Password

{

private get => \_password;

set

{

\_password = value;

OnPropertyChanged(nameof(Password));

}

}

public Command SignIn { get; private set; }

public int TimerValue

{

get => \_timerValue;

set

{

\_timerValue = value;

OnPropertyChanged(nameof(TimerValue));

}

}

public LoginViewModel()

{

dt.Interval = TimeSpan.FromSeconds(1);

dt.Tick += Dt\_Tick;

SignIn = new Command((\_) =>

{

if (IsError)

{

if (!\_loginModel.CheckCaptcha(Captcha, CaptchaInput))

{

MessageBox.Show("Captcha введена неверно\n Вход в систему заблокирован на 10 секунд", "Ошибка входа", MessageBoxButton.OK, icon: MessageBoxImage.Error);

CaptchaInput = string.Empty;

IsError = true;

\_isLock = true;

dt.Start();

return;

}

}

\_isAccess = \_loginModel.GetUser(Login, Password);

GetAccess();

}, (\_) => (Login != string.Empty && Password != string.Empty && !(\_isLock)));

}

private void Dt\_Tick(object sender, EventArgs e)

{

TimerValue++;

if (TimerValue > \_timeLock)

{

dt.Stop();

TimerValue = 0;

\_isLock = false;

MessageBox.Show("Вход в систему доступен", "Уведомление", MessageBoxButton.OK, icon: MessageBoxImage.Information);

return;

}

}

private void GetAccess()

{

if (\_isAccess)

{

IsError = false;

MessageBox.Show($"Добро пожаловать!", "Успешный вход", button: MessageBoxButton.OK, icon: MessageBoxImage.Information);

Application.Current.Windows.OfType<LoginView>().SingleOrDefault(x => x.IsVisible).Hide();

\_loginModel.OpenUserView();

}

else

{

MessageBox.Show("Логин или пароль введены неверно", "Ошибка входа", button: MessageBoxButton.OK, icon: MessageBoxImage.Error);

IsError = true;

\_captcha = new Captcha();

Captcha = \_captcha.Generate(6);

OnPropertyChanged(nameof(Captcha));

}

}

public void VisiblePasswordOn()

{

ChangeVisiblePassword(true);

}

public void VisiblePasswordOff()

{

ChangeVisiblePassword(false);

}

private void ChangeVisiblePassword(bool state)

{

ShowMaskPass = !state;

OnPropertyChanged(nameof(ShowMaskPass));

ShowUnmaskPass = state;

OnPropertyChanged(nameof(ShowUnmaskPass));

}

public void SetPassword(string password) => Password = password;

}

}

**Б.5 UserViewModel.cs**

using System.Collections.Generic;

using System.Collections.ObjectModel;

using PrintingHouseApplication.MyCommand;

using PrintingHouseApplication.Models;

using PrintingHouseApplication.Views;

using System.Windows;

namespace PrintingHouseApplication.ViewModels

{

internal class UserViewModel : BaseViewModel

{

private ObservableCollection<User> \_users = new ObservableCollection<User>();

private User \_selectedUser;

private ActionView \_actionView;

public ReadOnlyObservableCollection<User> Users { get; private set; }

public Command LoadList { get; private set; }

public Command Add { get; private set; }

public Command Change { get; private set; }

public Command Delete { get; private set; }

public User SelectedUser

{

get => \_selectedUser;

set

{

\_selectedUser = value;

OnPropertyChanged(nameof(SelectedUser));

UserModel.SelectedUser = SelectedUser;

}

}

public UserViewModel()

{

Users = new ReadOnlyObservableCollection<User>(\_users);

LoadList = new Command((\_) =>

{

\_users.Clear();

List<User> users = UserModel.GetUsers();

users.ForEach(x => \_users.Add(x));

}, null);

Add = new Command((\_) =>

{

\_actionView = new ActionView(new AddUserViewModel());

\_actionView.ShowDialog();

},null);

Change = new Command((\_) =>

{

\_actionView = new ActionView(new UpdUserViewModel());

\_actionView.ShowDialog();

SelectedUser = null;

}, (\_)=> SelectedUser !=null);

Delete = new Command((\_) => {

MessageBoxResult result = MessageBox.Show($"Удалить пользователя с номером #{SelectedUser.Id}?",

"Удаление",

MessageBoxButton.OKCancel,

MessageBoxImage.Question);

bool resDel = result == MessageBoxResult.OK ? UserModel.DeleteUser(SelectedUser.Id) : false;

string title = "Удаление";

if (resDel)

{

MessageBox.Show("Удаление успешно выполнено", title, MessageBoxButton.OK, MessageBoxImage.Information);

}

else

{

MessageBox.Show("Удаление пользователя отменено", title, MessageBoxButton.OK, MessageBoxImage.Error);

}

SelectedUser = null;

}, (\_) => SelectedUser != null);

}

}

}

**Б.6 ProfileViewModel.cs**

using System.Windows.Media;

using PrintingHouseApplication.Models;

namespace PrintingHouseApplication.ViewModels

{

class ProfileViewModel : BaseViewModel

{

private string \_employeeRole;

private Employee \_currentEmployee;

private ImageSource \_imgSource;

public ImageSource ImgSource

{

get => \_imgSource;

set

{

\_imgSource = value;

OnPropertyChanged(nameof(ImgSource));

}

}

public Employee CurrentEmployee

{

get => \_currentEmployee;

set

{

\_currentEmployee = value;

OnPropertyChanged(nameof(CurrentEmployee));

}

}

public string EmployeeRole

{

get => "Должность: " + \_employeeRole;

private set

{

\_employeeRole = value;

OnPropertyChanged(nameof(EmployeeRole));

}

}

public ProfileViewModel()

{

CurrentEmployee = UserHelper.GetEmployee();

EmployeeRole = UserHelper.GetUserRole();

ImgSource = ProfileModel.GetImage();

}

}

}

**Б.7 ProductViewModel.cs**

using System;

using System.Windows;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using PrintingHouseApplication.Models;

using PrintingHouseApplication.MyCommand;

using PrintingHouseApplication.Views;

namespace PrintingHouseApplication.ViewModels

{

class ProductViewModel : BaseViewModel

{

private ObservableCollection<Product> \_products = new ObservableCollection<Product>();

public ReadOnlyObservableCollection<Product> Products { get; private set; }

private Product \_selectedProduct;

private ActionView \_actionView = null;

public Command LoadList { get; private set; }

public Command Add { get; private set; }

public Command Change { get; private set; }

public Command Delete { get; private set; }

public Product SelectedProduct

{

get => \_selectedProduct;

set

{

\_selectedProduct = value;

OnPropertyChanged(nameof(SelectedProduct));

ProductModel.SelectedProduct = SelectedProduct;

}

}

public ProductViewModel()

{

Products = new ReadOnlyObservableCollection<Product>(\_products);

LoadList = new Command((\_)=> {

\_products.Clear();

List<Product> prods = ProductModel.GetProducts();

prods.ForEach(x=> \_products.Add(x));

}, null);

Add = new Command((\_)=> {

\_actionView = new ActionView(new AddProductViewModel());

\_actionView.ShowDialog();

}, null);

Change = new Command((\_) =>

{

\_actionView = new ActionView(new UpdProductViewModel());

\_actionView.ShowDialog();

SelectedProduct = null;

}, (\_) => SelectedProduct != null);

Delete = new Command((\_) => {

MessageBoxResult result = MessageBox.Show($"Удалить продукцию с номером #{SelectedProduct.Id}?",

"Удаление",

MessageBoxButton.OKCancel,

MessageBoxImage.Question);

bool resDel = result == MessageBoxResult.OK ? ProductModel.DeleteProduct(SelectedProduct.Id) : false;

string title = "Удаление";

if (resDel)

{

MessageBox.Show("Удаление успешно выполнено", title, MessageBoxButton.OK, MessageBoxImage.Information);

}

else

{

MessageBox.Show("Удаление цеха отменено", title, MessageBoxButton.OK, MessageBoxImage.Error);

}

SelectedProduct = null;

}, (\_)=> SelectedProduct!=null );

}

}

}

**Б.8 EmployeeViewModel.cs**

using System;

using System.Windows;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using PrintingHouseApplication.Models;

using PrintingHouseApplication.MyCommand;

using PrintingHouseApplication.Views;

namespace PrintingHouseApplication.ViewModels

{

class EmployeeViewModel : BaseViewModel

{

private ObservableCollection<Employee> \_employees = new ObservableCollection<Employee>();

public ReadOnlyObservableCollection<Employee> Employees { get; private set; }

private EmployeeModel \_empModel = new EmployeeModel();

private ActionView \_actionView = null;

private Employee \_selectedEmployee = null;

public Command Add { get; private set; }

public Command Change { get; private set; }

public Command Delete { get; private set; }

public Command LoadList { get; private set; }

public Employee SelectedEmployee

{

get => \_selectedEmployee;

set

{

\_selectedEmployee = value;

OnPropertyChanged(nameof(SelectedEmployee));

EmployeeModel.SelectedEmployee = SelectedEmployee;

}

}

public EmployeeViewModel()

{

Employees = new ReadOnlyObservableCollection<Employee>(\_employees);

LoadList = new Command((\_) => {

\_employees.Clear();

List<Employee> temp = \_empModel.GetEmployees();

temp.ForEach(x => \_employees.Add(x));

}, null);

Add = new Command((\_) =>

{

\_actionView = new ActionView(new AddEmployeeViewModel());

\_actionView.ShowDialog();

}, null);

Change = new Command((\_) =>

{

\_actionView = new ActionView(new UpdEmployeeViewModel());

\_actionView.ShowDialog();

SelectedEmployee = null;

}, (\_) => SelectedEmployee != null);

Delete = new Command((\_) =>

{

MessageBoxResult result = MessageBox.Show($"Удалить работника с номером #{SelectedEmployee.Id}?",

"Удаление",

MessageBoxButton.OKCancel,

MessageBoxImage.Question);

bool resDel = result == MessageBoxResult.OK ? \_empModel.DeleteEmployee(SelectedEmployee.Id) : false;

string title = "Удаление";

if (resDel)

{

MessageBox.Show("Удаление успешно выполнено", title, MessageBoxButton.OK, MessageBoxImage.Information);

}

else

{

MessageBox.Show("Удаление работника отменено", title, MessageBoxButton.OK, MessageBoxImage.Warning);

}

SelectedEmployee = null;

}, (\_) => SelectedEmployee != null);

}

}

}

**Б.9 DepartmentViewModel.cs**

using System;

using System.Windows;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using PrintingHouseApplication.Models;

using PrintingHouseApplication.Views;

using PrintingHouseApplication.MyCommand;

namespace PrintingHouseApplication.ViewModels

{

class DepartmentViewModel : BaseViewModel

{

private ObservableCollection<Department> \_departments = new ObservableCollection<Department>();

public ReadOnlyObservableCollection<Department> Departments { get; private set; }

private DepartmentModel \_departmentModel = new DepartmentModel();

private ActionView \_actionView = null;

private Department \_selectedDepartment;

public Command Add { get; private set; }

public Command Change { get; private set; }

public Command Delete { get; private set; }

public Command LoadList { get; private set; }

public Department SelectedDepartment

{

get => \_selectedDepartment;

set

{

\_selectedDepartment = value;

OnPropertyChanged(nameof(SelectedDepartment));

DepartmentModel.SelectedDepartment = SelectedDepartment;

}

}

public DepartmentViewModel()

{

Departments = new ReadOnlyObservableCollection<Department>(\_departments);

LoadList = new Command((\_) => {

\_departments.Clear();

List<Department> temp = \_departmentModel.GetDepartments();

temp.ForEach(x => \_departments.Add(x));

}, null);

Add = new Command((\_) =>

{

\_actionView = new ActionView(new AddDepartmentViewModel());

\_actionView.ShowDialog();

}, null);

Change = new Command(\_ =>

{

\_actionView = new ActionView(new UpdDepartmentViewModel());

\_actionView.ShowDialog();

SelectedDepartment = null;

}, \_ => SelectedDepartment != null);

Delete = new Command((\_) =>

{

MessageBoxResult result = MessageBox.Show($"Удалить цех с номером #{SelectedDepartment.Id}?",

"Удаление",

MessageBoxButton.OKCancel,

MessageBoxImage.Question);

bool resDel = result == MessageBoxResult.OK ? \_departmentModel.DeleteDepartment(SelectedDepartment.Id) : false;

string title = "Удаление";

if (resDel)

{

MessageBox.Show("Удаление успешно выполнено", title, MessageBoxButton.OK, MessageBoxImage.Information);

}

else

{

MessageBox.Show("Удаление цеха отменено", title, MessageBoxButton.OK, MessageBoxImage.Error);

}

SelectedDepartment = null;

}, (\_) => SelectedDepartment != null);

}

}

}

**Б.10 ContractViewModel.cs**

using System;

using System.Windows;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using PrintingHouseApplication.Views;

using PrintingHouseApplication.MyCommand;

using PrintingHouseApplication.Models;

namespace PrintingHouseApplication.ViewModels

{

internal class ContractViewModel: BaseViewModel

{

private ObservableCollection<Contract> \_contracts = new ObservableCollection<Contract>();

private Contract \_selectedContract;

private ActionView \_actionView;

public ReadOnlyObservableCollection<Contract> Contracts { get; private set; }

public Command Add { get; private set; }

public Command Change { get; private set; }

public Command Delete { get; private set; }

public Command LoadList { get; private set; }

public Contract SelectedContract

{

get => \_selectedContract;

set

{

\_selectedContract = value;

OnPropertyChanged(nameof(SelectedContract));

ContractModel.SelectedContract = SelectedContract;

}

}

public ContractViewModel()

{

Contracts = new ReadOnlyObservableCollection<Contract>(\_contracts);

LoadList = new Command((\_) =>

{

\_contracts.Clear();

List<Contract> contracts = ContractModel.GetContracts();

contracts.ForEach(x => \_contracts.Add(x));

}, null);

Add = new Command((\_) =>

{

\_actionView = new ActionView(new AddContractViewModel());

\_actionView.ShowDialog();

}, null);

Change = new Command((\_) => {

\_actionView = new ActionView(new UpdContractViewModel());

\_actionView.ShowDialog();

SelectedContract = null;

}, (\_)=> SelectedContract!=null);

Delete = new Command((\_) => {

MessageBoxResult result = MessageBox.Show($"Удалить договор с номером #{SelectedContract.Id}?",

"Удаление",

MessageBoxButton.OKCancel,

MessageBoxImage.Question);

bool resDel = result == MessageBoxResult.OK ? ContractModel.DeleteContract(SelectedContract.Id): false;

string title = "Удаление";

if (resDel)

{

MessageBox.Show("Удаление успешно выполнено", title, MessageBoxButton.OK, MessageBoxImage.Information);

}

else

{

MessageBox.Show("Удаление договора отменено", title, MessageBoxButton.OK, MessageBoxImage.Error);

}

SelectedContract = null;

}, null);

}

}

}

**Б.11 TypeProdViewModel.cs**

using System;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using System.Windows;

using PrintingHouseApplication.Models;

using PrintingHouseApplication.MyCommand;

using PrintingHouseApplication.Views;

namespace PrintingHouseApplication.ViewModels

{

internal class TypeProdViewModel: BaseViewModel

{

private ObservableCollection<TypeProd> \_typesProd = new ObservableCollection<TypeProd>();

private TypeProd \_selectedType;

private ActionView \_actionView;

public ReadOnlyObservableCollection<TypeProd> TypesProd { get; private set; }

public Command LoadList { get; private set; }

public Command Add { get; private set; }

public Command Change { get; private set; }

public Command Delete { get; private set; }

public TypeProd SelectedType

{

get => \_selectedType;

set

{

\_selectedType = value;

OnPropertyChanged(nameof(SelectedType));

TypesProdModel.SelectedTypeProd = SelectedType;

}

}

public TypeProdViewModel()

{

TypesProd = new ReadOnlyObservableCollection<TypeProd>(\_typesProd);

LoadList = new Command((\_) =>

{

\_typesProd.Clear();

List<TypeProd> types = TypesProdModel.GetTypeProds();

types.ForEach(x => \_typesProd.Add(x));

}, null);

Add = new Command((\_) =>

{

\_actionView = new ActionView(new AddTypeProdViewModel());

\_actionView.ShowDialog();

}, null);

Change = new Command((\_) =>

{

\_actionView = new ActionView(new UpdTypeProdViewModel());

\_actionView.ShowDialog();

SelectedType = null;

}, (\_) => SelectedType != null);

Delete = new Command((\_) =>

{

MessageBoxResult result = MessageBox.Show($"Удалить вид продукции с номером #{SelectedType.Id}?",

"Удаление",

MessageBoxButton.OKCancel,

MessageBoxImage.Question);

bool resDel = result == MessageBoxResult.OK ? TypesProdModel.DeleteTypeProd(SelectedType.Id) : false;

string title = "Удаление";

if (resDel)

{

MessageBox.Show("Удаление успешно выполнено", title, MessageBoxButton.OK, MessageBoxImage.Information);

}

else

{ }

SelectedType = null;

}, (\_) => SelectedType != null);

}

}

}

**Б.12 UserHelper.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PrintingHouseApplication.Models

{

class UserHelper

{

private static User \_user;

public static bool SetUser(User user)

{

\_user = user;

return \_user != null;

}

public static Employee GetEmployee()

{

Employee emp = null;

using (DataContext db = new DataContext())

{

emp = db.Employees.SingleOrDefault(e => e.Id == \_user.Id && !e.IsDeleted);

}

return emp;

}

public static string GetUserRole()

{

return \_user != null ? \_user.Role.Name : string.Empty;

}

public static byte[] GetImageData()

{

return \_user != null ? \_user.Image : null;

}

}

}

**Б.13 Captcha.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PrintingHouseApplication.Models

{

class Captcha

{

private string \_strCaptcha;

private string \_allowChar;

private string \_allowDigitalChar;

private Random \_random = new Random();

public Captcha()

{

\_allowChar = "A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z";

\_allowDigitalChar = "1,2,3,4,5,6,7,8,9,0";

}

public string Generate(int N)

{

\_strCaptcha = string.Empty;

string temp = \_allowChar.ToLower() + "," + \_allowDigitalChar;

string[] separChar = temp.Split(',');

for (int i = 0; i < N; ++i)

{

temp = separChar[(\_random.Next(0, separChar.Length))];

\_strCaptcha += temp;

}

return \_strCaptcha;

}

}

}

**Б.14 ManagerView.xaml**

<Window x:Class="PrintingHouseApplication.Views.ManagerView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:PrintingHouseApplication.Views"

xmlns:vm="clr-namespace:PrintingHouseApplication.ViewModels"

mc:Ignorable="d"

Title="Модуль менеджера" Height="500" Width="900"

FontSize="15"

Closed="Window\_Closed">

<Window.DataContext>

<vm:ViewModels/>

</Window.DataContext>

<Window.Resources>

<Style TargetType="{x:Type Button}">

<Setter Property="Margin" Value="5"/>

<Setter Property="Width" Value="100"/>

<Setter Property="Height" Value="35"/>

</Style>

<Style TargetType="{x:Type TextBlock}">

<Setter Property="FontSize" Value="10"/>

</Style>

</Window.Resources>

<Grid>

<TabControl>

<!--Профиль-->

<TabItem Header="Профиль"

DataContext="{Binding ProfileVM }">

<Grid ShowGridLines="False">

<Grid.RowDefinitions>

<RowDefinition Height="0.5\*"/>

<RowDefinition Height="6\*"/>

</Grid.RowDefinitions>

<StackPanel Grid.Row="1">

<Ellipse Width="200" Height="200" Margin="5">

<Ellipse.Fill>

<ImageBrush ImageSource="{Binding ImgSource}"

Stretch="UniformToFill"/>

</Ellipse.Fill>

</Ellipse>

<TextBlock

HorizontalAlignment="Center"

Margin="5"

Background="Bisque">

<TextBlock.Text>

<MultiBinding StringFormat="{} {0} {1} {2}">

<Binding Path="CurrentEmployee.Surname"/>

<Binding Path="CurrentEmployee.Name"/>

<Binding Path="CurrentEmployee.Patronymic"/>

</MultiBinding>

</TextBlock.Text>

</TextBlock>

<Label Content="{Binding EmployeeRole}"

HorizontalContentAlignment="Center"

HorizontalAlignment="Center"

Margin="5"

Background="Bisque"/>

</StackPanel>

</Grid>

</TabItem>

<!--Продукция-->

<TabItem Header="Продукция"

DataContext="{Binding ProductVM}">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="7\*"/>

</Grid.RowDefinitions>

<StackPanel Orientation="Horizontal"

Grid.Row="0">

<Button Content="Добавить"

Command="{Binding Add}"/>

<Button Content="Изменить"

Command="{Binding Change}"/>

<Button Content="Удалить"

Command="{Binding Delete}"/>

<Button Content="Загрузить"

Command="{Binding LoadList}"/>

</StackPanel>

<DataGrid HorizontalContentAlignment="Center"

IsReadOnly="True"

Grid.Row="1"

x:Name="dg\_production"

ItemsSource="{Binding Products}"

AutoGenerateColumns="False"

CanUserAddRows="False"

CanUserDeleteRows="False"

CanUserReorderColumns="False"

SelectedItem="{Binding SelectedProduct}">

<DataGrid.Columns>

<DataGridTextColumn Header="Код"

Width="50"

Binding="{Binding Id}"/>

<DataGridTextColumn Header="Цех"

Width="150"

Binding="{Binding Department.Name}"/>

<DataGridTextColumn Header="Название"

Width="150"

Binding="{Binding Name}"/>

<DataGridTextColumn Header="Цена за шт."

Width="150"

Binding="{Binding PricePerPiece}"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</TabItem>

<!--Контракты-->

<TabItem Header="Контракты"

DataContext="{Binding ContractVM}">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="7\*"/>

</Grid.RowDefinitions>

<StackPanel Orientation="Horizontal"

Grid.Row="0">

<Button Content="Добавить"

Command="{Binding Add}"/>

<Button Content="Изменить"

Command="{Binding Change}"/>

<Button Content="Удалить"

Command="{Binding Delete}"/>

<Button Content="Загрузить"

Command="{Binding LoadList}"/>

</StackPanel>

<DataGrid HorizontalContentAlignment="Center"

IsReadOnly="True"

Grid.Row="1"

ItemsSource="{Binding Contracts}"

AutoGenerateColumns="False"

CanUserAddRows="False"

CanUserDeleteRows="False"

CanUserReorderColumns="False"

SelectedItem="{Binding SelectedContract}">

<DataGrid.Columns>

<DataGridTextColumn Header="Код"

Width="50"

Binding="{Binding Id}"/>

<DataGridTextColumn Header="Работник"

Width="150">

<DataGridTextColumn.Binding>

<MultiBinding StringFormat="{} {0} {1} {2}">

<Binding Path="Employee.Surname"/>

<Binding Path="Employee.Name"/>

<Binding Path="Employee.Patronymic"/>

</MultiBinding>

</DataGridTextColumn.Binding>

</DataGridTextColumn>

<DataGridTextColumn Header="Заказчик"

Width="150"

Binding="{Binding CustomerName}"/>

<DataGridTextColumn Header="Адрес"

Width="150">

<DataGridTextColumn.Binding>

<MultiBinding StringFormat="{} {0}, {1}, {2}">

<Binding Path="Address.City"/>

<Binding Path="Address.Street"/>

<Binding Path="Address.House"/>

</MultiBinding>

</DataGridTextColumn.Binding>

</DataGridTextColumn>

<DataGridTextColumn Header="Дата оформления"

Width="150"

Binding="{Binding RegistrDate, StringFormat='{}{0:dd.MM.yyyy}'}"/>

<DataGridTextColumn Header="Дата выполнения"

Width="150"

Binding="{Binding CompletionDate, StringFormat='{}{0:dd.MM.yyyy}'}"/>

</DataGrid.Columns>

<DataGrid.RowDetailsTemplate>

<DataTemplate>

<StackPanel>

<StackPanel Orientation="Horizontal" Margin="5">

<TextBlock Text="Работник:" FontWeight="SemiBold"/>

<TextBlock>

<TextBlock.Text>

<MultiBinding StringFormat="{} {0} {1} {2}">

<Binding Path="Employee.Surname"/>

<Binding Path="Employee.Name"/>

<Binding Path="Employee.Patronymic"/>

</MultiBinding>

</TextBlock.Text>

</TextBlock>

</StackPanel>

<StackPanel Orientation="Horizontal" Margin="5">

<TextBlock Text="Заказчик: " FontWeight="SemiBold"/>

<TextBlock Text="{Binding CustomerName}"/>

</StackPanel>

<StackPanel Orientation="Horizontal" Margin="5">

<TextBlock Text="Адрес: " FontWeight="SemiBold"/>

<TextBlock>

<TextBlock.Text>

<MultiBinding StringFormat="{} {0}, {1}, {2}">

<Binding Path="Address.City"/>

<Binding Path="Address.Street"/>

<Binding Path="Address.House"/>

</MultiBinding>

</TextBlock.Text>

</TextBlock>

</StackPanel>

<StackPanel Orientation="Horizontal" Margin="5">

<TextBlock Text="Дата оформеления: " FontWeight="SemiBold"/>

<TextBlock Text="{Binding RegistrDate, StringFormat='{}{0:dd.MM.yyyy}'}"/>

</StackPanel>

<StackPanel Orientation="Horizontal" Margin="5">

<TextBlock Text="Дата выполнения: " FontWeight="SemiBold"/>

<TextBlock Text="{Binding CompletionDate, StringFormat='{}{0:dd.MM.yyyy}'}"/>

</StackPanel>

<Label Content="Продукция в контракте:"/>

<DataGrid AutoGenerateColumns="False"

IsReadOnly="True"

ItemsSource="{Binding Orders}"

ScrollViewer.CanContentScroll="True">

<DataGrid.Columns>

<DataGridTextColumn Header="Код" Binding="{Binding Product.Id}"/>

<DataGridTextColumn Header="Название" Binding="{Binding Product.Name}"/>

<DataGridTextColumn Header="Количество" Binding="{Binding ProductAmount}"/>

</DataGrid.Columns>

</DataGrid>

</StackPanel>

</DataTemplate>

</DataGrid.RowDetailsTemplate>

</DataGrid>

</Grid>

</TabItem>

</TabControl>

</Grid>

</Window>

**Б.15 DirectorView.xaml**

<Window x:Class="PrintingHouseApplication.Views.DirectorView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:PrintingHouseApplication.Views"

xmlns:vm="clr-namespace:PrintingHouseApplication.ViewModels"

mc:Ignorable="d"

Title="Модуль директора" Height="450" Width="800"

FontSize="18"

Closed="Window\_Closed">

<Window.DataContext>

<vm:ViewModels/>

</Window.DataContext>

<Window.Resources>

<Style TargetType="{x:Type Button}">

<Setter Property="Margin" Value="5"/>

<Setter Property="Width" Value="100"/>

<Setter Property="Height" Value="35"/>

</Style>

</Window.Resources>

<Grid>

<TabControl>

<!--Профиль-->

<TabItem Header="Профиль"

DataContext="{Binding ProfileVM }">

<Grid ShowGridLines="False">

<Grid.RowDefinitions>

<RowDefinition Height="0.5\*"/>

<RowDefinition Height="6\*"/>

</Grid.RowDefinitions>

<StackPanel Grid.Row="1">

<Ellipse Width="200" Height="200" Margin="5">

<Ellipse.Fill>

<ImageBrush ImageSource="{Binding ImgSource}"

Stretch="UniformToFill"/>

</Ellipse.Fill>

</Ellipse>

<TextBlock

HorizontalAlignment="Center"

Margin="5"

Background="Bisque">

<TextBlock.Text>

<MultiBinding StringFormat="{} {0} {1} {2}">

<Binding Path="CurrentEmployee.Surname"/>

<Binding Path="CurrentEmployee.Name"/>

<Binding Path="CurrentEmployee.Patronymic"/>

</MultiBinding>

</TextBlock.Text>

</TextBlock>

<Label Content="{Binding EmployeeRole}"

HorizontalContentAlignment="Center"

HorizontalAlignment="Center"

Margin="5"

Background="Bisque"/>

</StackPanel>

</Grid>

</TabItem>

<!--Сотрудники-->

<TabItem Header="Сотрудники"

DataContext="{Binding EmployeeVM}">

<Grid>

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition Height="7\*"/>

</Grid.RowDefinitions>

<StackPanel Orientation="Horizontal"

Grid.Row="0">

<Button Content="Добавить"

Command="{Binding Add}"/>

<Button Content="Изменить"

Command="{Binding Change}"/>

<Button Content="Удалить"

Command="{Binding Delete}"/>

<Button Content="Загрузить"

Command="{Binding LoadList}"/>

</StackPanel>

<DataGrid HorizontalContentAlignment="Center"

Grid.Row="1"

IsReadOnly="True"

x:Name="dg\_employees"

ItemsSource="{Binding Employees}"

AutoGenerateColumns="False"

CanUserAddRows="False"

CanUserDeleteRows="False"

CanUserReorderColumns="False"

SelectedItem="{Binding SelectedEmployee}">

<DataGrid.Columns>

<DataGridTextColumn Header="Код"

Width="50"

Binding="{Binding Id}"/>

<DataGridTextColumn Header="Фамилия"

Width="150"

Binding="{Binding Surname}"/>

<DataGridTextColumn Header="Имя"

Width="150"

Binding="{Binding Name}"/>

<DataGridTextColumn Header="Отчество"

Width="150"

Binding="{Binding Patronymic}"/>

<DataGridTextColumn Header="Телефон"

Width="150"

Binding="{Binding Phone}"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</TabItem>

<!--Цеха типографии-->

<TabItem Header="Цеха типографии"

DataContext="{Binding DepartmentVM}">

<Grid>

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition Height="7\*"/>

</Grid.RowDefinitions>

<StackPanel Orientation="Horizontal"

Grid.Row="0">

<Button Content="Добавить"

Command="{Binding Add}"/>

<Button Content="Изменить"

Command="{Binding Change}"/>

<Button Content="Удалить"

Command="{Binding Delete}"/>

<Button Content="Загрузить"

Command="{Binding LoadList}"/>

</StackPanel>

<DataGrid HorizontalContentAlignment="Center"

IsReadOnly="True"

Grid.Row="1"

x:Name="dg\_departments"

ItemsSource="{Binding Departments}"

AutoGenerateColumns="False"

CanUserAddRows="False"

CanUserDeleteRows="False"

CanUserReorderColumns="False"

SelectedItem="{Binding SelectedDepartment}">

<DataGrid.Columns>

<DataGridTextColumn Header="Код"

Width="50"

Binding="{Binding Id}"/>

<DataGridTextColumn Header="Название"

Width="150"

Binding="{Binding Name}"/>

<DataGridTextColumn Header="Начальник"

Width="250">

<DataGridTextColumn.Binding>

<MultiBinding StringFormat="{}{0} {1} {2}">

<Binding Path="Employee.Surname"/>

<Binding Path="Employee.Name"/>

<Binding Path="Employee.Patronymic"/>

</MultiBinding>

</DataGridTextColumn.Binding>

</DataGridTextColumn>

<DataGridTextColumn Header="Телефон"

Width="150"

Binding="{Binding Phone}"/>

<DataGridTextColumn Header="Тип продукции"

Width="150"

Binding="{Binding TypeProd.Name}"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</TabItem>

<!--Виды продукции-->

<TabItem Header="Види продукции"

DataContext="{Binding TypeProdVM}">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="7\*"/>

</Grid.RowDefinitions>

<StackPanel Orientation="Horizontal"

Grid.Row="0">

<Button Content="Добавить"

Command="{Binding Add}"/>

<Button Content="Изменить"

Command="{Binding Change}"/>

<Button Content="Удалить"

Command="{Binding Delete}"/>

<Button Content="Загрузить"

Command="{Binding LoadList}"/>

</StackPanel>

<DataGrid HorizontalContentAlignment="Center"

IsReadOnly="True"

Grid.Row="1"

x:Name="dg\_typesProd"

ItemsSource="{Binding TypesProd}"

AutoGenerateColumns="False"

CanUserAddRows="False"

CanUserDeleteRows="False"

CanUserReorderColumns="False"

SelectedItem="{Binding SelectedType}">

<DataGrid.Columns>

<DataGridTextColumn Header="Код"

Width="50"

Binding="{Binding Id}"/>

<DataGridTextColumn Header="Название"

Width="150"

Binding="{Binding Name}"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</TabItem>

</TabControl>

</Grid>

</Window>

**Б.16 LoginView.xaml**

<Window x:Class="PrintingHouseApplication.Views.LoginView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:PrintingHouseApplication.Views"

xmlns:vm="clr-namespace:PrintingHouseApplication.ViewModels"

mc:Ignorable="d"

Title="Авторизация" Height="400" Width="400"

FontSize="20">

<Window.DataContext>

<vm:LoginViewModel/>

</Window.DataContext>

<Window.Resources>

<BooleanToVisibilityConverter x:Key="BoolToVisConverter" />

</Window.Resources>

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="45\*"/>

<RowDefinition Height="193\*"/>

<RowDefinition Height="80\*"/>

</Grid.RowDefinitions>

<StackPanel

Orientation="Vertical"

HorizontalAlignment="Center"

VerticalAlignment="Center">

<Label Content="Вход" FontSize="25"/>

</StackPanel>

<StackPanel

Orientation="Horizontal"

HorizontalAlignment="Center"

Grid.Row="1">

<StackPanel Orientation="Vertical"

Margin="10"

HorizontalAlignment="Center"

>

<Label Content="Логин:"/>

<Label Content="Пароль:" />

</StackPanel>

<StackPanel Orientation="Vertical"

Margin="10"

HorizontalAlignment="Center">

<TextBox Width="200" Margin="5" Text="{Binding Login, UpdateSourceTrigger=PropertyChanged}"/>

<PasswordBox Width="200"

x:Name="PassMask"

Margin="5"

PasswordChar="•"

Visibility="{Binding ShowMaskPass, Converter={StaticResource BoolToVisConverter}}"

PasswordChanged="PassMask\_PasswordChanged" />

<TextBox Width="200"

x:Name="PassUnMask"

Margin="5"

Text="{Binding Password, UpdateSourceTrigger=PropertyChanged}"

Visibility="{Binding ShowUnmaskPass, Converter={StaticResource BoolToVisConverter}}"

>

</TextBox

<TextBlock VerticalAlignment="Center"

Margin="5"

Text="Показать пароль"

FontSize="13"

TextDecorations="Underline"

Foreground="LightSeaGreen"

PreviewMouseDown="TextBlock\_PreviewMouseDown"

PreviewMouseUp="TextBlock\_PreviewMouseUp"

/

<Border BorderThickness="1"

BorderBrush="Black"

Margin="5"

Visibility="{Binding IsError, Converter={StaticResource BoolToVisConverter}}">

<TextBlock Margin="5"

FontSize="30"

TextAlignment="Center"

TextDecorations="Baseline"

Visibility="{Binding IsError, Converter={StaticResource BoolToVisConverter}}"

Width="100">

<TextBlock.Effect >

<BlurEffect Radius="4"/>

</TextBlock.Effect>

<TextBlock Foreground="DarkBlue"

Text="{Binding Captcha}"

TextDecorations="Strikethrough">

<TextBlock.Effect>

<DropShadowEffect

Direction="35"

ShadowDepth="5"

BlurRadius="5"/>

</TextBlock.Effect>

</TextBlock>

</TextBlock>

</Border>

<TextBox Margin="5"

Text="{Binding CaptchaInput, UpdateSourceTrigger=PropertyChanged, Mode=TwoWay}"

Visibility="{Binding IsError, Converter={StaticResource BoolToVisConverter}}"/>

</StackPanel>

</StackPanel>

<StackPanel Grid.Row="2"

HorizontalAlignment="Center"

VerticalAlignment="Center">

<Button Content="Войти"

Height="40"

Width="200"

Command="{Binding SignIn}"

/>

</StackPanel>

</Grid>

</Window>

**Б.17 AdminView.xaml**

<Window x:Class="PrintingHouseApplication.Views.AdminView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:PrintingHouseApplication.Views"

xmlns:vm="clr-namespace:PrintingHouseApplication.ViewModels"

mc:Ignorable="d"

Title="Модуль администратора" Height="450" Width="800"

Closed="Window\_Closed" >

<Window.DataContext>

<vm:ViewModels/>

</Window.DataContext>

<Window.Resources>

<Style TargetType="{x:Type Button}">

<Setter Property="Margin" Value="5"/>

<Setter Property="Width" Value="100"/>

<Setter Property="Height" Value="35"/>

</Style>

<Style TargetType="{x:Type TextBlock}">

<Setter Property="FontSize" Value="10"/>

</Style>

<vm:ImageConverter x:Key="ImageConverter"/>

</Window.Resources>

<Grid>

<TabControl>

<TabItem Header="Профиль"

DataContext="{Binding ProfileVM }">

<Grid ShowGridLines="False">

<Grid.RowDefinitions>

<RowDefinition Height="0.5\*"/>

<RowDefinition Height="6\*"/>

</Grid.RowDefinitions>

<StackPanel Grid.Row="1">

<Ellipse Width="200" Height="200" Margin="5">

<Ellipse.Fill>

<ImageBrush ImageSource="{Binding ImgSource}"

Stretch="UniformToFill"/>

</Ellipse.Fill>

</Ellipse>

<TextBlock

HorizontalAlignment="Center"

Margin="5"

Background="Bisque"

FontSize="15">

<TextBlock.Text>

<MultiBinding StringFormat="{} {0} {1} {2}">

<Binding Path="CurrentEmployee.Surname"/>

<Binding Path="CurrentEmployee.Name"/>

<Binding Path="CurrentEmployee.Patronymic"/>

</MultiBinding>

</TextBlock.Text>

</TextBlock>

<Label Content="{Binding EmployeeRole}"

HorizontalContentAlignment="Center"

HorizontalAlignment="Center"

Margin="5"

Background="Bisque"

FontSize="15"/>

</StackPanel>

</Grid>

</TabItem>

<TabItem Header="Пользователи"

DataContext="{Binding UserVM}">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="7\*"/>

</Grid.RowDefinitions>

<StackPanel Orientation="Horizontal"

Grid.Row="0">

<Button Content="Добавить"

Command="{Binding Add}"/>

<Button Content="Изменить"

Command="{Binding Change}"/>

<Button Content="Удалить"

Command="{Binding Delete}"/>

<Button Content="Загрузить"

Command="{Binding LoadList}"/>

</StackPanel>

<DataGrid HorizontalContentAlignment="Center"

IsReadOnly="True"

Grid.Row="1"

x:Name="dg\_users"

ItemsSource="{Binding Users}"

AutoGenerateColumns="False"

CanUserAddRows="False"

CanUserDeleteRows="False"

CanUserReorderColumns="False"

SelectedItem="{Binding SelectedUser}">

<DataGrid.Columns>

<DataGridTextColumn Header="Код"

Width="50"

Binding="{Binding Id}"/>

<DataGridTextColumn Header="Работник"

Width="150">

<DataGridTextColumn.Binding>

<MultiBinding StringFormat="{} {0} {1} {2}">

<Binding Path="Employee.Surname"/>

<Binding Path="Employee.Name"/>

<Binding Path="Employee.Patronymic"/>

</MultiBinding>

</DataGridTextColumn.Binding>

</DataGridTextColumn>

<DataGridTextColumn Header="Роль"

Width="150"

Binding="{Binding Role.Name}"/>

<DataGridTextColumn Header="Логин"

Width="150"

Binding="{Binding Login}"/>

<DataGridTextColumn Header="Пароль"

Width="150"

Binding="{Binding Password}"/>

<DataGridTemplateColumn Header="Картинка">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Image Source="{Binding Image}"

Width="100"

Height="100"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

</Grid>

</TabItem>

<TabItem Header="Сотрудники"

DataContext="{Binding EmployeeVM}">

<Grid>

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition Height="7\*"/>

</Grid.RowDefinitions>

<StackPanel Orientation="Horizontal"

Grid.Row="0">

<Button Content="Добавить"

Command="{Binding Add}"/>

<Button Content="Изменить"

Command="{Binding Change}"/>

<Button Content="Удалить"

Command="{Binding Delete}"/>

<Button Content="Загрузить"

Command="{Binding LoadList}"/>

</StackPanel>

<DataGrid HorizontalContentAlignment="Center"

Grid.Row="1"

IsReadOnly="True"

x:Name="dg\_employees"

ItemsSource="{Binding Employees}"

AutoGenerateColumns="False"

CanUserAddRows="False"

CanUserDeleteRows="False"

CanUserReorderColumns="False"

SelectedItem="{Binding SelectedEmployee}">

<DataGrid.Columns>

<DataGridTextColumn Header="Код"

Width="50"

Binding="{Binding Id}"/>

<DataGridTextColumn Header="Фамилия"

Width="150"

Binding="{Binding Surname}"/>

<DataGridTextColumn Header="Имя"

Width="150"

Binding="{Binding Name}"/>

<DataGridTextColumn Header="Отчество"

Width="150"

Binding="{Binding Patronymic}"/>

<DataGridTextColumn Header="Телефон"

Width="150"

Binding="{Binding Phone}"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</TabItem>

<TabItem Header="Цеха типографии"

DataContext="{Binding DepartmentVM}">

<Grid>

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition Height="7\*"/>

</Grid.RowDefinitions>

<StackPanel Orientation="Horizontal"

Grid.Row="0">

<Button Content="Добавить"

Command="{Binding Add}"/>

<Button Content="Изменить"

Command="{Binding Change}"/>

<Button Content="Удалить"

Command="{Binding Delete}"/>

<Button Content="Загрузить"

Command="{Binding LoadList}"/>

</StackPanel>

<DataGrid HorizontalContentAlignment="Center"

IsReadOnly="True"

Grid.Row="1"

x:Name="dg\_departments"

ItemsSource="{Binding Departments}"

AutoGenerateColumns="False"

CanUserAddRows="False"

CanUserDeleteRows="False"

CanUserReorderColumns="False"

SelectedItem="{Binding SelectedDepartment}">

<DataGrid.Columns>

<DataGridTextColumn Header="Код"

Width="50"

Binding="{Binding Id}"/>

<DataGridTextColumn Header="Название"

Width="150"

Binding="{Binding Name}"/>

<DataGridTextColumn Header="Начальник"

Width="250">

<DataGridTextColumn.Binding>

<MultiBinding StringFormat="{}{0} {1} {2}">

<Binding Path="Employee.Surname"/>

<Binding Path="Employee.Name"/>

<Binding Path="Employee.Patronymic"/>

</MultiBinding>

</DataGridTextColumn.Binding>

</DataGridTextColumn>

<DataGridTextColumn Header="Телефон"

Width="150"

Binding="{Binding Phone}"/>

<DataGridTextColumn Header="Тип продукции"

Width="150"

Binding="{Binding TypeProd.Name}"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</TabItem>

<TabItem Header="Види продукции"

DataContext="{Binding TypeProdVM}">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="7\*"/>

</Grid.RowDefinitions>

<StackPanel Orientation="Horizontal"

Grid.Row="0">

<Button Content="Добавить"

Command="{Binding Add}"/>

<Button Content="Изменить"

Command="{Binding Change}"/>

<Button Content="Удалить"

Command="{Binding Delete}"/>

<Button Content="Загрузить"

Command="{Binding LoadList}"/>

</StackPanel>

<DataGrid HorizontalContentAlignment="Center"

IsReadOnly="True"

Grid.Row="1"

x:Name="dg\_typesProd"

ItemsSource="{Binding TypesProd}"

AutoGenerateColumns="False"

CanUserAddRows="False"

CanUserDeleteRows="False"

CanUserReorderColumns="False"

SelectedItem="{Binding SelectedType}">

<DataGrid.Columns>

<DataGridTextColumn Header="Код"

Width="50"

Binding="{Binding Id}"/>

<DataGridTextColumn Header="Название"

Width="150"

Binding="{Binding Name}"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</TabItem>

<TabItem Header="Продукция"

DataContext="{Binding ProductVM}">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="7\*"/>

</Grid.RowDefinitions>

<StackPanel Orientation="Horizontal"

Grid.Row="0">

<Button Content="Добавить"

Command="{Binding Add}"/>

<Button Content="Изменить"

Command="{Binding Change}"/>

<Button Content="Удалить"

Command="{Binding Delete}"/>

<Button Content="Загрузить"

Command="{Binding LoadList}"/>

</StackPanel>

<DataGrid HorizontalContentAlignment="Center"

IsReadOnly="True"

Grid.Row="1"

x:Name="dg\_production"

ItemsSource="{Binding Products}"

AutoGenerateColumns="False"

CanUserAddRows="False"

CanUserDeleteRows="False"

CanUserReorderColumns="False"

SelectedItem="{Binding SelectedProduct}">

<DataGrid.Columns>

<DataGridTextColumn Header="Код"

Width="50"

Binding="{Binding Id}"/>

<DataGridTextColumn Header="Цех"

Width="150"

Binding="{Binding Department.Name}"/>

<DataGridTextColumn Header="Название"

Width="150"

Binding="{Binding Name}"/>

<DataGridTextColumn Header="Цена за шт."

Width="150"

Binding="{Binding PricePerPiece}"/>

</DataGrid.Columns>

</DataGrid>

</Grid>

</TabItem>

<TabItem Header="Контракты"

DataContext="{Binding ContractVM}">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="7\*"/>

</Grid.RowDefinitions>

<StackPanel Orientation="Horizontal"

Grid.Row="0">

<Button Content="Добавить"

Command="{Binding Add}"/>

<Button Content="Изменить"

Command="{Binding Change}"/>

<Button Content="Удалить"

Command="{Binding Delete}"/>

<Button Content="Загрузить"

Command="{Binding LoadList}"/>

</StackPanel>

<DataGrid HorizontalContentAlignment="Center"

IsReadOnly="True"

Grid.Row="1"

ItemsSource="{Binding Contracts}"

AutoGenerateColumns="False"

CanUserAddRows="False"

CanUserDeleteRows="False"

CanUserReorderColumns="False"

SelectedItem="{Binding SelectedContract}">

<DataGrid.Columns>

<DataGridTextColumn Header="Код"

Width="50"

Binding="{Binding Id}"/>

<DataGridTextColumn Header="Работник"

Width="150">

<DataGridTextColumn.Binding>

<MultiBinding StringFormat="{} {0} {1} {2}">

<Binding Path="Employee.Surname"/>

<Binding Path="Employee.Name"/>

<Binding Path="Employee.Patronymic"/>

</MultiBinding>

</DataGridTextColumn.Binding>

</DataGridTextColumn>

<DataGridTextColumn Header="Заказчик"

Width="150"

Binding="{Binding CustomerName}"/>

<DataGridTextColumn Header="Адрес"

Width="150">

<DataGridTextColumn.Binding>

<MultiBinding StringFormat="{} {0}, {1}, {2}">

<Binding Path="Address.City"/>

<Binding Path="Address.Street"/>

<Binding Path="Address.House"/>

</MultiBinding>

</DataGridTextColumn.Binding>

</DataGridTextColumn>

<DataGridTextColumn Header="Дата оформления"

Width="150"

Binding="{Binding RegistrDate, StringFormat='{}{0:dd.MM.yyyy}'}"/>

<DataGridTextColumn Header="Дата выполнения"

Width="150"

Binding="{Binding CompletionDate, StringFormat='{}{0:dd.MM.yyyy}'}"/>

</DataGrid.Columns>

<DataGrid.RowDetailsTemplate>

<DataTemplate>

<StackPanel>

<StackPanel Orientation="Horizontal" Margin="5">

<TextBlock Text="Работник:" FontWeight="SemiBold"/>

<TextBlock>

<TextBlock.Text>

<MultiBinding StringFormat="{} {0} {1} {2}">

<Binding Path="Employee.Surname"/>

<Binding Path="Employee.Name"/>

<Binding Path="Employee.Patronymic"/>

</MultiBinding>

</TextBlock.Text>

</TextBlock>

</StackPanel>

<StackPanel Orientation="Horizontal" Margin="5">

<TextBlock Text="Заказчик: " FontWeight="SemiBold"/>

<TextBlock Text="{Binding CustomerName}"/>

</StackPanel>

<StackPanel Orientation="Horizontal" Margin="5">

<TextBlock Text="Адрес: " FontWeight="SemiBold"/>

<TextBlock>

<TextBlock.Text>

<MultiBinding StringFormat="{} {0}, {1}, {2}">

<Binding Path="Address.City"/>

<Binding Path="Address.Street"/>

<Binding Path="Address.House"/>

</MultiBinding>

</TextBlock.Text>

</TextBlock>

</StackPanel>

<StackPanel Orientation="Horizontal" Margin="5">

<TextBlock Text="Дата оформеления: " FontWeight="SemiBold"/>

<TextBlock Text="{Binding RegistrDate, StringFormat='{}{0:dd.MM.yyyy}'}"/>

</StackPanel>

<StackPanel Orientation="Horizontal" Margin="5">

<TextBlock Text="Дата выполнения: " FontWeight="SemiBold"/>

<TextBlock Text="{Binding CompletionDate, StringFormat='{}{0:dd.MM.yyyy}'}"/>

</StackPanel>

<Label Content="Продукция в контракте:"/>

<DataGrid AutoGenerateColumns="False"

IsReadOnly="True"

ItemsSource="{Binding Orders}"

ScrollViewer.CanContentScroll="True">

<DataGrid.Columns>

<DataGridTextColumn Header="Код" Binding="{Binding Product.Id}"/>

<DataGridTextColumn Header="Название" Binding="{Binding Product.Name}"/>

<DataGridTextColumn Header="Количество" Binding="{Binding ProductAmount}"/>

</DataGrid.Columns>

</DataGrid>

</StackPanel>

</DataTemplate>

</DataGrid.RowDetailsTemplate>

</DataGrid>

</Grid>

</TabItem>

</TabControl>

</Grid>

</Window>

**Б.18 ActionView.xaml**

<Window x:Class="PrintingHouseApplication.Views.ActionView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:PrintingHouseApplication.Views"

xmlns:vm="clr-namespace:PrintingHouseApplication.ViewModels"

xmlns:ctrl="clr-namespace:PrintingHouseApplication.Views.Controls"

mc:Ignorable="d"

Title="Модуль работы с данными" Height="450" Width="625">

<Window.DataContext>

<vm:ActionViewModel/>

</Window.DataContext>

<Window.Resources>

<DataTemplate DataType="{x:Type vm:AddEmployeeViewModel}">

<ctrl:AddEmployeeControl/>

</DataTemplate>

<DataTemplate DataType="{x:Type vm:UpdEmployeeViewModel}">

<ctrl:UpdEmployeeControl/>

</DataTemplate>

<DataTemplate DataType="{x:Type vm:AddDepartmentViewModel}">

<ctrl:AddDepartmentControl/>

</DataTemplate>

<DataTemplate DataType="{x:Type vm:UpdDepartmentViewModel}">

<ctrl:UpdDepartmentControl/>

</DataTemplate>

<DataTemplate DataType="{x:Type vm:AddTypeProdViewModel}">

<ctrl:AddTypeProdControl/>

</DataTemplate>

<DataTemplate DataType="{x:Type vm:UpdTypeProdViewModel}">

<ctrl:UpdTypeProdControl/>

</DataTemplate>

<DataTemplate DataType="{x:Type vm:AddProductViewModel}">

<ctrl:AddProductControl/>

</DataTemplate>

<DataTemplate DataType="{x:Type vm:UpdProductViewModel}">

<ctrl:UpdProductControl/>

</DataTemplate>

<DataTemplate DataType="{x:Type vm:AddContractViewModel}">

<ctrl:AddContractControl/>

</DataTemplate>

<DataTemplate DataType="{x:Type vm:UpdContractViewModel}">

<ctrl:UpdContractControl/>

</DataTemplate>

<DataTemplate DataType="{x:Type vm:AddUserViewModel}">

<ctrl:AddUserControl/>

</DataTemplate>

<DataTemplate DataType="{x:Type vm:UpdUserViewModel}">

<ctrl:UpdUserControl/>

</DataTemplate>

</Window.Resources>

<Grid>

<ContentControl Content="{Binding SelectedViewModel}"/>

</Grid>

</Window>