МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА № 42

КУРСОВАЯ РАБОТА (ПРОЕКТ)   
ЗАЩИЩЕНА С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| доцент, к. т. н., доцент |  |  |  | А.В. Бржезовский |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОЙ РАБОТЕ  АВТОМАТИЗИРОВАННАЯ ИНФОРМАЦИОННАЯ СИСТЕМА ОБЕСПЕЧЕНИЯ РЕДАКЦИОННОЙ ДЕЯТЕЛЬНОСТИ СТУДЕНЧЕСКОЙ ГАЗЕТЫ |
|  |
| по дисциплине: МЕТОДЫ И СРЕДСТВА ИНФОРМАЦИОННЫХ СИСТЕМ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4926 |  |  |  | В.Н.Пировских |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2022

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc6438240)

[1 Описание предметной области 4](#_Toc6438241)

[1.1 Назначение 4](#_Toc6438242)

[1.2 Возможности бизнеса 4](#_Toc6438243)

[1.3 Цели создания системы 5](#_Toc6438244)

[2 Требования к системе 5](#_Toc6438245)

[2.1 Пользовательские требования 5](#_Toc6438246)

2.2 Функциональные требования 12

[3 Концептуальная модель БД 14](#_Toc6438247)

[4 Физическая модель БД 15](#_Toc6438247)

[4.1 Хранимые процедуры 16](#_Toc6438250)

[4.2 Триггеры 19](#_Toc6438251)

[4.3 Индексы 19](#_Toc6438252)

[4.4 Ограничения целостности](#_Toc6438252) 19

[5 Интерфейс пользователя 20](#_Toc6438253)

[ЗАКЛЮЧЕНИЕ 29](#_Toc6438254)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 30](#_Toc6438255)

[ПРИЛОЖЕНИЕ А 31](#_Toc6438256)

[ПРИЛОЖЕНИЕ В 36](#_Toc6438257)

ВВЕДЕНИЕ

Целью данной курсовой работы является разработка автоматизированной информационной системы обеспечения редакционной деятельности студенческой газеты.

Основной задачей разработки данной системы является обеспечение быстрого взаимодействия между участниками, а также между участниками и материалами, с которыми они должны работать.

Перед разработкой системы необходимо составить перечень бизнес-правил, пользовательских и функциональных требований, составить UseCase-диаграмму, показывающую, каким образом и какие участники будут взаимодействовать с разрабатываемой системой, описать алгоритмы поведения пользователей в системе. Все это оформляется с учетом требований пользователей и необходимыми атрибутами разработки.

Для разработки интерфейса данной информационной системы была выбрана среда Microsoft Visual Studio 2019, база данных спроектирована в SQL Server Management Studio 2018, диаграммы разрабатывались в среде разработки PowerDesigner.

1. Описание предметной области

1.1 Назначение

Назначение системы заключается в упрощении и повышении скорости редакционной деятельности студенческой газеты. В нашем ВУЗе существует творческая студия «Пресс-центр ГУАП», которая каждый месяц выпускает студенческую газету «Спутник студента». Но чтобы читателей становилось больше, газета должна выпускаться чаще. Например, ИТМО периодически выпускает свою газету и два раза в месяц. Уже сейчас есть проблема с тем, что все необходимые ресурсы для очередного номера (тексты статей, иллюстрации к ним, комментарии и правки) разбросаны по различным ресурсам (социальные сети, облачные хранилища). Сейчас это не так сильно влияет на работу и на скорость выпуска. Но если частота выпуска номеров возрастет, эта проблема может повлиять на скорость и качество работы (какой-то ресурс может просто потеряться среди сообщений – такое иногда происходит и сейчас). Также такая неструктурированная система может усложнить работу новичкам. А информационная система может помочь им быстрее влиться в работу. Далее разрабатываемая информационная система (ИС) будет называться «Система» (наименование Системы – RIS PRIS), а участник студии, который будет ею пользоваться, – «Пользователь».

1.2 Возможности бизнеса

Для качественной работы редакции газеты можно осуществить автоматизацию процесса выпуска газеты. Так как в процессе редакционной деятельности происходит много правок и изменений, вероятно появление большого количества ошибок. Также сказывается человеческий фактор, поэтому его следует снизить за счет автоматизации процесса. В редакционно-издательской системе (РИС) задается цепочка прохождения материала по разным стадиям подготовки – от написания текста до верстки и утверждения макета. Отдельным объектом управления будет материал. Для каждого объекта на каждом этапе работ может быть назначен ответственный участник студии, работающий над материалом на этом этапе, могут оставляться комментарии. Особенностью сложных комплексных РИС является то, что заданная технологическая цепочка стадий (конвейер работ) может охватывать не только этапы журналистско-корректорской работы над текстом, но и этапы подготовки иллюстраций и верстки издания до его отправки в типографию. Важными свойствами информационной системы при описании стадий прохождения материала являются возможность задания сроков окончания работ этого этапа и возможность автоматического оповещения студийца о задержке или передаче материала на следующую стадию.

1.3 Цели создания системы

**BR-1** Исключить использование социальных сетей для обмена всеми файлами и правками;

**BR-2** Достигнуть запланированного режима выпуска номера (два номера в месяц, а не один) за счет напоминания о дедлайнах с помощью уведомлений системы, а также за счет ускорения процесса выполнения работы в принципе;

**BR-3** Новые участники студии могут брать работу уже после первого месяца пребывания в студии;

**BR-4** За счет автоматизации процесса выполнения работы освободить администратора студии от обязанности следить за дедлайнами каждого участника студии.

1. Требования к системе

Требования к системе разработаны на основе анализа других систем со схожими функиями.

2.1 Пользовательские требования

Пользовательские требования будут разработаны для группы пользователей Системы (Журналист и Верстальщик), эти требования представляют собой список задач, выполнение которых Пользователь видит полезным и необходимым для себя в конечном приложении.

**UR**-**1 Работа с учетной записью пользователя** (пользователь должен иметь возможность авторизации, регистрации и выхода из учетной записи);

**UR**-**1.1** **Вход в систему** (вход в систему должен осуществляться через ввод пароля и логина)**;**

**UR**-**1.2 Регистрация в системе** (пользователь должен иметь доступ к окну регистрации, где он сможет ввести все свои персональные данные);

**UR**-**1.3** **Выход из системы** (пользователю должна быть доступна функция выхода из его учетной записи).

**UR**-**2 Работа с материалом в БД** (пользователь должен иметь возможность работать с материалами из БД в соответствии со своими правами доступа);

**UR-2.1 Добавление нового материала в БД** (пользователь должен иметь возможность добавлять в систему материалы газеты, а также назначать ответственных за выполнение работ людей);

**UR-2.2 Редактирование материала в БД** (пользователь должен иметь возможность изменять в системе материалы газеты);

**UR-2.3 Удаление материала в БД** (пользователь должен иметь возможность удалять в системе материалы газеты);

**UR-2.4 Переход на веб-страницу с ресурсами** (пользователь должен иметь возможность переходить по ссылке, которая ведет на веб-страницу с ресурсами, необходимыми для работы с материалом);

**UR-2.5 Отправка материала на следующий уровень** (пользователь должен иметь возможность отправить материал на следующий этап работы, при этом пользователь должен иметь возможность оставлять комментарии к работам);

**UR-2.6 Просмотр списка материалов** (пользователь должен иметь возможность выбрать доступный ему материал из списка для последующих манипуляций с ним).

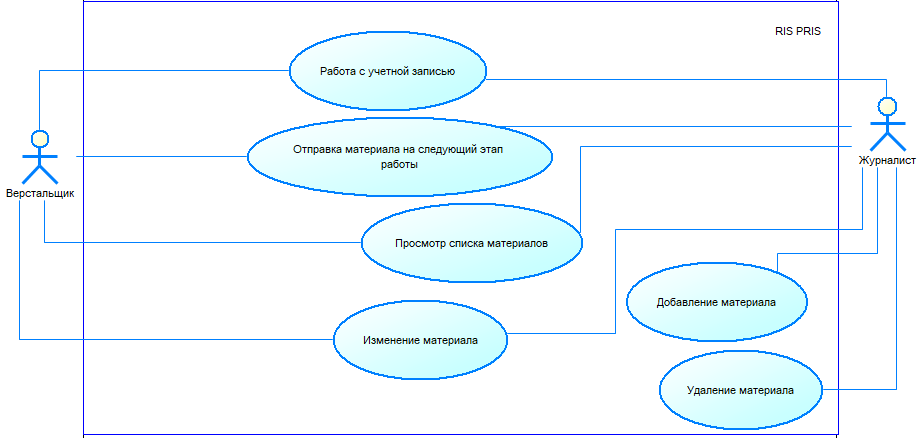


Рисунок 1 – UseCase-диаграмма

Описание варианта использования «Вход в систему»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор и название | UR-1.1 Вход в систему | | |
| Основное действующее лицо | Журналист, Верстальщик | **Дополнительное действующее лицо** | БД |
| Описание | Пользователь авторизуется в системе | | |
| Триггер | Необходимо войти в систему для начала работы | | |
| Предварительные условия | PRE 1 – Пользователь запустил систему | | |
| Выходные условия | POST 1 – Открывается форма «ЛК» | | |
| Нормальное направление | 1. Пользователь вводит логин 2. Пользователь вводит пароль 3. Пользователь нажимает кнопку «Войти» 4. Проверка на незаполненные поля 5. Система передает полученные данные на сервер 6. Проверка валидности полученных данных 7. Отправка положительного ответа 8. Открывается следующая форма (ЛК) 9. Вывод приветственной надписи с именем пользователя | | |
| Альтернативное направление | 1. Пользователь вводит логин 2. Пользователь вводит пароль 3. Пользователь нажимает кнопку «Войти» 4. Проверка на незаполненные поля 5. Система передает полученные данные на сервер 6. Проверка валидности полученных данных 7. Отправка отрицательного ответа 8. Вывод сообщения о необходимости проверить корректность логина либо с предложением зарегистрироваться 9. Поля ввода очищаются | | |
| Исключения | 1. Произошла ошибка передачи данных в БД | | |

Описание варианта использования «Регистрация в системе»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор и название | UR-1.2 Регистрация в системе | | |
| Основное действующее лицо | Журналист, Верстальщик | **Дополнительное действующее лицо** | БД |
| Описание | Пользователь регистрируется в системе | | |
| Триггер | Сообщение системы о необходимости зарегистрироваться | | |
| Предварительные условия | PRE 1 – Пользователь перешел к форме «Регистрация» | | |
| Выходные условия | POST 1 – На сервер отправляются данные о новом пользователе | | |
| Нормальное направление | 1. Пользователь заполняет все поля 2. Пользователь нажимает кнопку «Зарегистрироваться» 3. Проверка на незаполненные поля 4. Проверка совпадения паролей 5. Система передает полученные данные на сервер 6. Проверка на уникальность логина 7. Отправка положительного ответа 8. Открывается форма авторизации | | |
| Альтернативное направление (1) | 1. Пользователь заполняет все поля 2. Пользователь нажимает кнопку «Зарегистрироваться» 3. Проверка на незаполненные поля 4. Проверка совпадения паролей 5. Вывод сообщения о несовпадении паролей 6. Очищение полей Пароль и Повторите пароль | | |
| Альтернативное направление (2) | 1. Пользователь заполняет все поля 2. Пользователь нажимает кнопку «Зарегистрироваться» 3. Проверка на незаполненные поля 4. Проверка совпадения паролей 5. Система передает полученные данные на сервер 6. Проверка на уникальность логина 7. Отправка отрицательного ответа 8. Вывод сообщения о неуникальности логина 9. Очищение полей Логин, Пароль и Повторите пароль | | |
| Исключения | 1. Произошла ошибка передачи данных в БД | | |

Описание варианта использования «Выход из системы»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор и название | UR-1.3 Выход из системы | | |
| Основное действующее лицо | Журналист, Верстальщик | **Дополнительное действующее лицо** | БД |
| Описание | Пользователь осуществляет выход из учетной записи системы | | |
| Триггер | Необходимость покинуть ЛК | | |
| Предварительные условия | PRE 1 – Пользователь нажал на кнопку «Выйти» | | |
| Выходные условия | POST 1 – Закрытие формы ЛК и открытие формы «Авторизация» | | |
| Нормальное направление | 1. Пользователь нажимает на кнопку «Выйти» 2. Закрытие формы ЛК 3. Открытие формы Авторизация | | |
| Альтернативное направление | нет | | |
| Исключения | нет | | |

Описание варианта использования «Добавление нового материала в БД»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор и название | UR-2.1 Добавление нового материала в БД | | |
| Основное действующее лицо | Журналист | **Дополнительное действующее лицо** | БД |
| Описание | Пользователь добавляет новый материал в БД системы | | |
| Триггер | Планирование написания нового материала | | |
| Предварительные условия | PRE 1 – Пользователь нажал на кнопку «Создать материал» | | |
| Выходные условия | POST 1 – Добавление в БД нового материала | | |
| Нормальное направление | 1. Пользователь нажимает на кнопку «Создать материал» 2. Открывается форма «Окно редактирования материала» 3. Пользователь заполняет все поля 4. Пользователь нажимает кнопку «Далее» 5. Проверка на незаполненные поля 6. Пользователь выбирает из списка ответственных за выполнение создаваемой работы 7. Проверка на незаполненные поля 8. Пользователь нажимает кнопку «Сохранить» 9. Система передает полученные данные на сервер 10. Вывод сообщения об успешном создании материала 11. Закрытие формы «Окно редактирования материала» | | |
| Альтернативное направление | 1. Пользователь нажимает на кнопку «Создать материал» 2. Открывается форма «Окно редактирования материала» 3. Пользователь заполняет все поля 4. Пользователь нажимает кнопку «Далее» 5. Проверка на незаполненные поля 6. Вывод сообщения о незаполнении каких-либо полей | | |
| Исключения | 1. Произошла ошибка передачи данных в БД | | |

Описание варианта использования «Редактирование материала в БД»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор и название | UR-2.2 Редактирование материала в БД | | |
| Основное действующее лицо | Журналист, Верстальщик | **Дополнительное действующее лицо** | БД |
| Описание | Пользователь изменяет материал в БД системы | | |
| Триггер | Для журналиста: необходимость скорректировать ранее созданный материал;  Для верстальщика: необходимость добавить новую ссылку, относящуюся к материалу | | |
| Предварительные условия | PRE 1 – Пользователь нажал на кнопку «Редактировать материал» | | |
| Выходные условия | POST 1 – Обновление в БД материала | | |
| Нормальное направление | 1. Пользователь выбирает из списка нужный материал 2. Пользователь нажимает на кнопку «Редактировать материал» 3. Открывается форма «Окно редактирования материала» 4. Система передает полученные данные на сервер 5. Система возвращает необходимые данные в поля 6. Пользователь изменяет нужные данные 7. Пользователь нажимает кнопку «Далее» 8. Проверка на незаполненные поля 9. Пользователь нажимает кнопку «Сохранить» 10. Система передает полученные данные на сервер 11. Вывод сообщения об успешном редактировании материала 12. Закрытие формы «Окно редактирования материала» | | |
| Альтернативное направление | 1. Пользователь выбирает из списка нужный материал 2. Пользователь нажимает на кнопку «Редактировать материал» 3. Открывается форма «Окно редактирования материала» 4. Система передает полученные данные на сервер 5. Сервер возвращает необходимые данные в поля 6. Пользователь изменяет нужные данные 7. Пользователь нажимает кнопку «Далее» 8. Проверка на незаполненные поля 9. Вывод сообщения о незаполнении каких-либо полей | | |
| Исключения | 1. Произошла ошибка передачи данных в БД | | |

Описание варианта использования «Удаление материала в БД»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор и название | UR-2.3 Удаление материала в БД | | |
| Основное действующее лицо | Журналист | **Дополнительное действующее лицо** | БД |
| Описание | Пользователь удаляет материал в БД системы | | |
| Триггер | Было принято решение не писать определенный материал | | |
| Предварительные условия | PRE 1 – Пользователь нажал на кнопку «Удалить материал» | | |
| Выходные условия | POST 1 – Удаление в БД материала | | |
| Нормальное направление | 1. Пользователь выбирает из списка нужный материал 2. Пользователь нажимает на кнопку «Удалить материал» 3. Вывод сообщения для подтверждения действия 4. Пользователь нажимает «Ок» 5. Система передает полученные данные на сервер 6. Вывод сообщения об успешном удалении материала из БД 7. Обновление списка материалов | | |
| Альтернативное направление | 1. Пользователь выбирает из списка нужный материал 2. Пользователь нажимает на кнопку «Удалить материал» 3. Вывод сообщения для подтверждения действия 4. Пользователь нажимает «Отмена» 5. Возврат к окну ЛК | | |
| Исключения | 1. Произошла ошибка передачи данных в БД | | |

Описание варианта использования «Переход на веб-страницу с ресурсами»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор и название | UR-2.4 Переход на веб-страницу с ресурсами | | |
| Основное действующее лицо | Верстальщик | **Дополнительное действующее лицо** | БД |
| Описание | Пользователь переходит по ссылке, которая ведет на веб-страницу с ресурсами, необходимыми для работы с материалом | | |
| Триггер | Пользователю необходимо получить необходимые ресурсы для работы с материалом | | |
| Предварительные условия | PRE 1 – Пользователь нажал на кнопку «Перейти к ресурсам» | | |
| Выходные условия | POST 1 – Пользователь попадает на веб-страницу с ресурсами | | |
| Нормальное направление | 1. Пользователь выбирает из списка нужный материал 2. Пользователь нажимает кнопку «Перейти к ресурсам» 3. Система передает полученные данные на сервер 4. Сервер возвращает необходимые данные в переменную «Ссылка» 5. Система производит переход на веб-страницу по полученной ссылке | | |
| Альтернативное направление | 1. Пользователь выбирает из списка нужный материал 2. Пользователь нажимает кнопку «Перейти к ресурсам» 3. Система передает полученные данные на сервер 4. Сервер не возвращает необходимые данные в переменную «Ссылка» 5. Вывод сообщения о невозможности перейти к ресурсам | | |
| Исключения | 1. Произошла ошибка передачи данных в БД | | |

Описание варианта использования «Отправка материала на следующий уровень»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор и название | UR-2.5 Отправка материала на следующий уровень | | |
| Основное действующее лицо | Журналист, Верстальщик | **Дополнительное действующее лицо** | БД |
| Описание | Пользователь хочет отправить материал на следующий этап работы, при этом пользователь должен иметь возможность оставлять комментарии к работам | | |
| Триггер | Пользователю выполнил свою часть работы и хочет передать материал дальше | | |
| Предварительные условия | PRE 1 – Пользователь нажимает на кнопку «Передать» | | |
| Выходные условия | POST 1 – Статус материала меняется | | |
| Нормальное направление | 1. Пользователь выполняет сценарий «Создание нового материала» или «Редактирование материала» 2. Пользователь выбирает из списка нужный материал 3. По желанию Пользователь может написать комментарии к материалу 4. Пользователь нажимает кнопку «Передать» 5. Система передает полученные данные на сервер 6. Вывод сообщения об успешной передаче материала на следующий уровень 7. Обновление списка материалов | | |
| Альтернативное направление | 1. Пользователь выполняет сценарий «Создание нового материала» или «Редактирование материала» 2. По желанию Пользователь может написать комментарии к материалу 3. Пользователь нажимает кнопку «Передать» 4. Вывод сообщения о необходимости выбрать материал из списка | | |
| Исключения | 1. Произошла ошибка передачи данных в БД | | |

*Прим. По кнопке «Вернуть назад» производятся те же действия, но в обратном порядке.*

Описание варианта использования «Просмотр списка материалов»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор и название | UR-2.6 Просмотр списка материалов | | |
| Основное действующее лицо | Журналист, Верстальщик | **Дополнительное действующее лицо** | БД |
| Описание | Пользователь хочет выбрать доступный ему материал из списка для последующих манипуляций с ним | | |
| Триггер | Пользователь хочет изменить, передать, удалить материал, посмотреть комментарии к нему, перейти по ссылке, узнать свой дедлайн | | |
| Предварительные условия | PRE 1 – Пользователь кликом мыши выбирает из списка материал | | |
| Выходные условия | POST 1 – Сервер передает данные о выбранном материале в нужные поля | | |
| Нормальное направление | 1. Пользователь выбирает из списка нужный материал 2. Система передает полученные данные на сервер 3. Сервер возвращает нужные данные о материале 4. Запись в поля нужных данных 5. Вызов функции на изменение цвета даты дедлайна для материала | | |
| Альтернативное направление | нет | | |
| Исключения | 1. Произошла ошибка передачи данных в БД | | |

2.2 Функциональные требования

Функциональные требования определяют, каким должно быть поведение Системы в тех или иных условиях. Они определяют, какими функциями должна обладать Система с точки зрения разработки, чтобы Пользователь смог выполнить свои задачи (пользовательские требования) в рамках бизнес-требований.

**UR**-**1.1** **Вход в систему** (вход в систему должен осуществляться через ввод пароля и логина)

**FR-1.1.1: Проверка логина**

**FR-1.1.2: Проверка пароля**

**UR**-**1.2 Регистрация в системе** (пользователь должен иметь доступ к окну регистрации, где он сможет ввести все свои персональные данные)

**FR-1.2.1 Проверка на корректность заполнения**

**FR-1.2.2 Проверка на совпадение паролей**

**FR-1.2.3 Проверка на существование в БД логина**

**FR-1.2.4 Сохранение данных пользователя в БД**

**UR**-**1.3** **Выход из системы** (пользователю должна быть доступна функция выхода из его учетной записи)

**FR-1.3.1 Выход из учетной записи в интерфейсе**

**UR-2.1 Добавление нового материала в БД** (пользователь должен иметь возможность добавлять в систему материалы газеты, а также назначать ответственных за выполнение работ людей)

**FR-2.1.1 Открытие формы создания материала**

**FR-2.1.2 Проверка на корректность заполнения**

**FR-2.1.3 Выбор из списка участника студии**

**FR-2.1.4 Сохранение нового материала в БД**

**UR-2.2 Редактирование материала в БД** (пользователь должен иметь возможность изменять в системе материалы газеты)

**FR-2.2.1 Открытие формы редактирования материала**

**FR-2.2.2 Редактирование полей**

**FR-2.2.3 Сохранение изменений в материале в БД**

**UR-2.3 Удаление материала в БД** (пользователь должен иметь возможность удалять в системе материалы газеты)

**FR-2.3.1 Удаление материла из БД**

**FR-2.3.2 Вывод уведомления о подтверждении удаления материала**

**UR-2.4 Переход на веб-страницу с ресурсами** (пользователь должен иметь возможность переходить по ссылке, которая ведет на веб-страницу с ресурсами, необходимыми для работы с материалом)

**FR-2.4.1 Переход по ссылке на веб-страницу с ресурсами (Google Диск/Яндекс Диск) по нажатию на кнопку**

**UR-2.5 Отправка материала на следующий уровень** (пользователь должен иметь возможность отправить материал на следующий этап работы, при этом пользователь должен иметь возможность оставлять комментарии к работам)

**FR-2.5.1 Ввод текста комментария в соответствующее поле**

**FR-2.5.2 Изменение статуса материала и сохранение комментария в БД за этим материалом при нажатии на кнопку**

**UR-2.6 Просмотр списка материалов** (пользователь должен иметь возможность выбрать доступный ему материал из списка для последующих манипуляций с ним)

**FR-2.6.1 Выбор материала в списке**

**FR-2.6.2 Вывод обозначенного соответствующим цветом дедлайна выполнения материала в соответствии с этапом, на котором он сейчас находится**

**FR-2.6.3 Вывод комментария с прошлого уровня**

1. Концептуальная модель БД

Разработанная база данных содержит в себе 8 таблиц:

* Material – таблица, содержащая информацию о материале (статье, полосе и т.д.);
* Member – таблица, содержащая информацию об участниках студии, т.н. Пользователях;
* Position – таблица, содержащая наименования и номера позиций, которые могут занимать участники студии, т.н. должности (например, администратор, бильд-редактор и т.д.);
* Specialization – таблица, содержащая наименования и номера специализаций, которыми обладают участники студии, специализация показывает, каким родом деятельности занимается определенный Пользователь (например, верстальщик);
* Status – таблица, содержащая на именования и номера статусов, которыми могут обладать материалы. Отображает то, на каком уровне выполнения материал находится в данный момент (например, «пишется текст» или «проходит коррекцию»);
* Resourse – таблица, содержащая ссылки на ресурсы, необходимые для выполнения работы над материалом;
* Comment – таблица, содержащая комментарии, оставляемые Пользователями друг другу;
* Release – таблица, в которой хранятся данные об определенном выпуске газеты (например, выпуск №5, июль 2022 года), к ней привязываются материалы, окончательно готовые к печати.

У всех таблиц отношения поставлены как 1:М, за исключением связи Creates – она отображает связь М:М между таблицами Member и Material, так как один Пользователь может работать над несколькими материалами, а один материал может изготавливаться несколькими Пользователями.

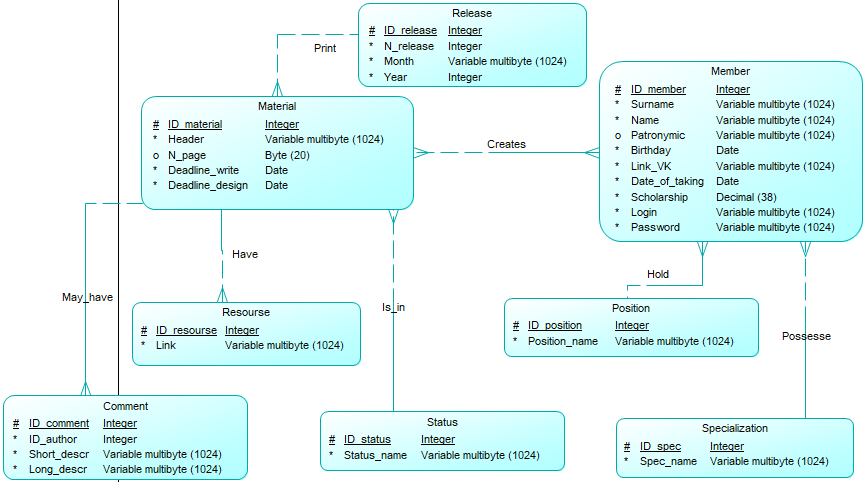


Рисунок 2 – Концептуальная модель БД

1. Физическая модель БД

Код запроса SQL для создания БД приведен в Приложении А. На рисунке 2 представлена схема базы данных, для которой спроектирован интерфейс.

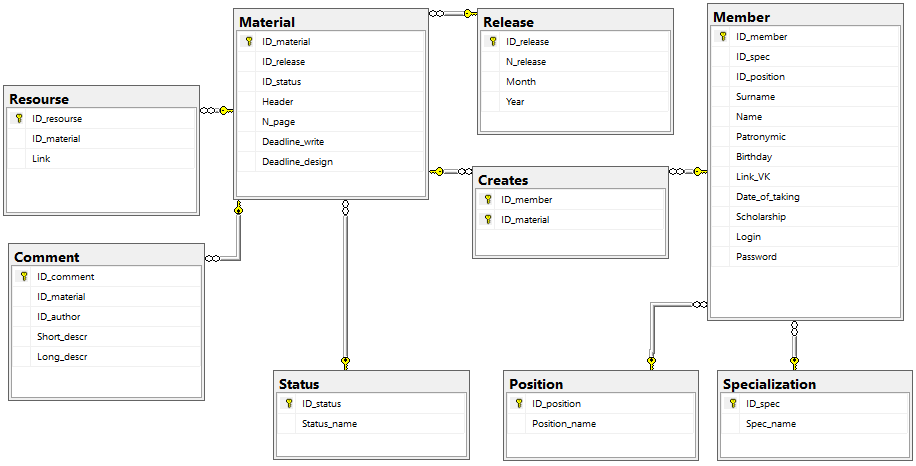


Рисунок 3 – Физическая модель БД

* 1. Хранимые процедуры

Созданные в БД хранимые процедуры (ХП) можно разбить на 2 группы:

* ХП для окна ЛК;
* ХП для окон редактирования материала.

Первая ХП: SelectHeaders (@ID\_member int), эта процедура по своей сути отвечает за правильное отображение списка материалов в ЛК, которые доступны человеку, который авторизовался в системе. Она проверяет права доступа, специализацию Пользователя (чтобы верстальщик не имел доступа к материалу, который даже еще не написан), статус, в котором находится материал. Принимает на вход один параметр – ID Пользователя, который авторизовался в системе, дальше процедура находит в БД, какими правами обладает Пользователь, какие материалы ему нужно выводить. И при помощи условий, основанных на сравнении ID специализаций и ID позиций, она определяет, какие материалы (то есть их заголовки и ID) нужно выводить этому пользователю. Срабатывает при загрузке формы «ЛК».

Вторая ХП: SelectDeadline (@ID\_material int, @ID\_member int), эта процедура возвращает в интерфейс дедлайн, до которого Пользователю необходимо выполнить свою часть работы относительно выбранного материала. И так как первая процедура уже проверила права доступа, выводиться будет именно тот дедлайн, который непосредственно относится к Пользователю, который на данный момент в системе. Срабатывает при выборе материала из списка. Принимает на вход ID материала и ID Пользователя.

Третья ХП: ChangeStatus (@ID\_material int), эта процедура отвечает за передачу материала на следующий уровень разработки, то есть увеличивает его статус на 1, причем задано определенное число – «потолок», дальше которого статус повышать уже нельзя, и он отправляется в печать. Срабатывает по нажатию кнопки «Передать». Принимает на вход 1 параметр – ID материала. Вместе с этим в интерфейсе происходит запись введенных Пользователем комментариев под этим материалом, они будут видны Пользователям выше.

Четвертая ХП: ChangeStatusBack (@ID\_material int), эта процедура совершает действие, противоположное ХП ChangeStatus, – возвращает материал назад. Нужна на случай, если материал по ошибке был передан на следующий уровень, либо Пользователь выше делает «правки» по работе Пользователя ниже.

Пятая ХП: SelectLink (@ID\_material int), эта процедура нужна для верстальщика, чтобы ему проще было переходить к ресурсам материала. Она просто возвращает ссылку, которая относится к выбранному материалу. Срабатывает при выборе материала из списка. Сам переход по ссылке можно осуществить нажатием кнопки «Перейти к ресурсам». Получает на вход 1 параметр – ID материала.

Шестая ХП: SelectComments (@ID\_material int, @ID\_member int), эта процедура возвращает комментарии, которые относятся к выбранному из списка материалу. Процесс происходит следующим образом: у каждого комментария при сохранении его в БД сохраняется и ID Пользователя, создавшего его. Процедура определяет позицию Пользователя, оставившего комментарий и Пользователя, которому этот комментарий должен выводиться. Если позиция оставившего комментарий больше, чем позиция того, кому он выводится, то комментарий отображается в интерфейсе. Принимает на вход 2 параметра: ID материала и ID Пользователя.

Седьмая ХП: EditFormAdding (@ID\_material int), эта процедура уже относится к форме редактирования материала. Она заполняет поля в окне редактирования материала, выбранного из списка. Срабатывает при нажатии кнопки «Редактировать материал». Принимает на вход 1 параметр – ID выбранного из списка материала.

Восьмая ХП: UpdateMaterial (@ID\_material int, @Link nvarchar(1024), @Header nvarchar(1024), @N\_page int, @Deadline\_write date, @Deadline\_design date), эта процедура просто осуществляет обновление материала в БД. Принимает на вход все атрибуты, которые относятся к таблице Material. Срабатывает при нажатии кнопки «Далее» в первом окне редактирования в режиме полного редактирования (этот режим доступен только журналисту, верстальщик при редактировании материала может обновлять только ссылку данного материала).

Девятая ХП: AddMaterial (@Header nvarchar(1024), @N\_page int, @Deadline\_write date, @Deadline\_design date), эта процедура просто осуществляет сохранение нового материала в БД. Принимает на вход все атрибуты, которые относятся к таблице Material. Срабатывает при нажатии кнопки «Далее» в первом окне редактирования в режиме создания материала.

Десятая ХП: LastMaterial, эта процедура не принимает параметров, только возвращает максимальный ID материала в БД. Нужна для того, чтобы при переходе ко второму окну редактирования/создания материала (где происходит выбор участников, ответственных за выполнение работы), Система понимала, под каким ID был записан последний материал, чтобы взять этот ID и записать под него ID участников в таблицу Creates.

Одиннадцатая ХП: MembersDoMaterial (@ID\_material int, @ID\_spec int), эта процедура выводит Пользователей, ответственных за выполнение выбранного материала. Получает на вход параметры ID материала и ID специальности – в зависимости от того, в какую из двух таблиц ( в 1й таблице отображается журналист, во 2й – верстальщик) будут записываться данные, будут задаваться разные ID специальностей – 1 (журналист) или 2 (верстальщик).

Скрипты всех ХП находятся в ПРИЛОЖЕНИИ А.

* 1. Триггеры.

Наложено ограничение на вставку данных в таблицу Member, по столбцу Login идет проверка на совпадение логинов с целью избежать создания одинаковых аккаунтов. Для этого создан триггер EqualLogins, который не позволяет добавить новый аккаунт, если был введен Login, который уже существует в БД.

Также, так как таблица Material связана с таблицами Comment и Creates, а таблица Member – с таблицей Creates, необходимо было создать триггер на удаление материала и участника, чтобы обеспечить сохранность данных и предотвратить появление «висячих» и лишних строк.

Скрипты триггеров находятся в ПРИЛОЖЕНИИ А.

* 1. Индексы

Большинство применяемых и достаточных индексов создались автоматически при генерации физической модели БД, на основе взаимосвязей между таблицами. Потребностям запросов они удовлетворяют. Также был создан индекс MemberSpec. Скрипты индексов находятся в ПРИЛОЖЕНИИ А.

* 1. Ограничения целостности

В таблице Material идет проверка на даты дедлайнов, столбцы Deadline\_write и Deadline\_design (чтобы участники не могли поставить дату дедлайна меньше сегодняшнего дня). Также в самом интерфейсе доступная дата для выбора – сегодняшний день с добавлением еще двух дней.

В таблице Member идет проверка на дату прихода в студию, столбец Date\_Of\_Taking (чтобы не была меньше даты официального основания газеты).

Эти ограничения целостности были заданы, для большего удобства, с помощью команд, они находятся в ПРИЛОЖЕНИИ А.

1. Интерфейс пользователя

Система разрабатывалась с помощью языка C#, интерфейс представляет собой 6 форм, которые были реализованы при помощи технологий, предоставляемых библиотекой ADO.NET.

Первая форма «Авторизация» (рисунок 4) представляет собой окно авторизации пользователя. После ввода логина и пароля отправляется запрос на сервер, поиск ведется по введенному логину. Если нужный логин найден, а пароль правильный – производится переход к форме с личным кабинетом (далее – ЛК). Если нужный логин найден, но пароль неправильный, появляется сообщение, говорящее об этом (рисунок 5). Если логин не нашелся, система оповещает пользователя о том, что ему следует проверить введенный логин или зарегистрироваться (рисунок 6).

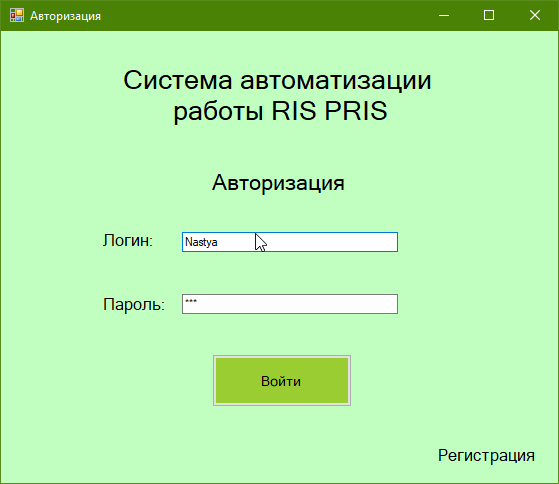


Рисунок 4 – Окно авторизации

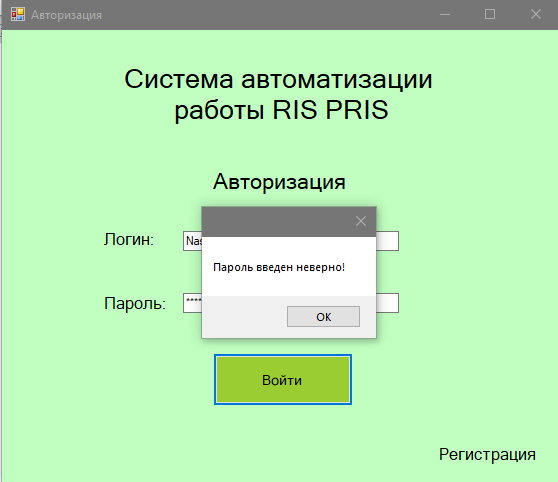


Рисунок 5 – Сообщение о неправильном пароле

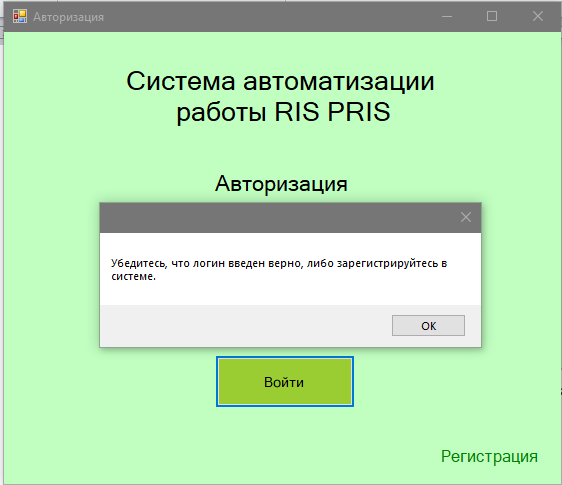


Рисунок 6 – Сообщение об ошибке в логине

Окно «Регистрация» выводится при нажатии на надпись «Регистрация» из формы «Авторизация» (рисунок 7).

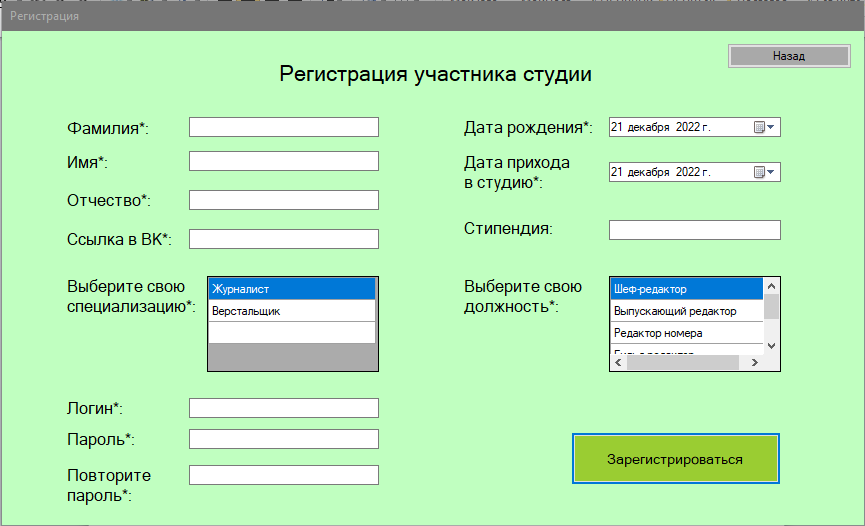


Рисунок 7 – Окно регистрации пользователя

Происходит также проверка на незаполненные поля (рисунок 8):

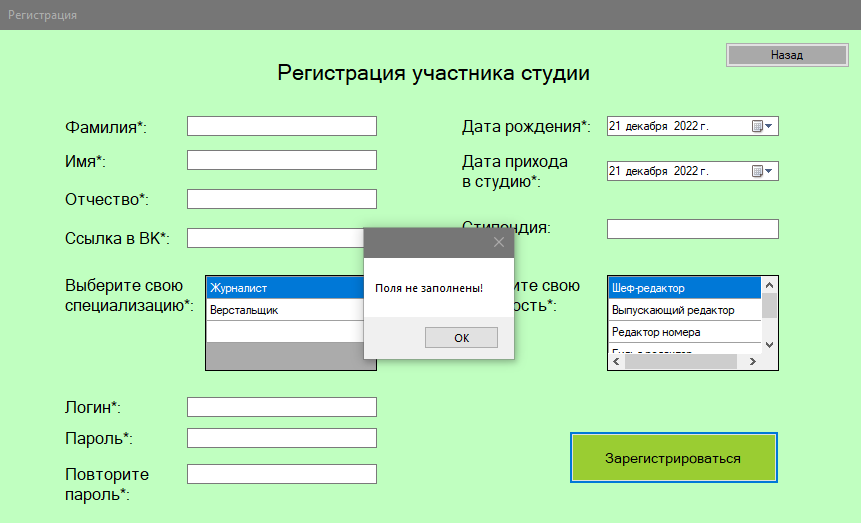


Рисунок 8 – Проверка на незаполненные поля

При успешной авторизации открывается окно ЛК. Причем в момент авторизации происходит проверка по ID специальности Пользователя. Это нужно для того, чтобы открыть специализированную под него форму. Если это журналист, то для него откроется окно ЛК, отображенное на рисунке 9. Если верстальщик – на рисунке 10.

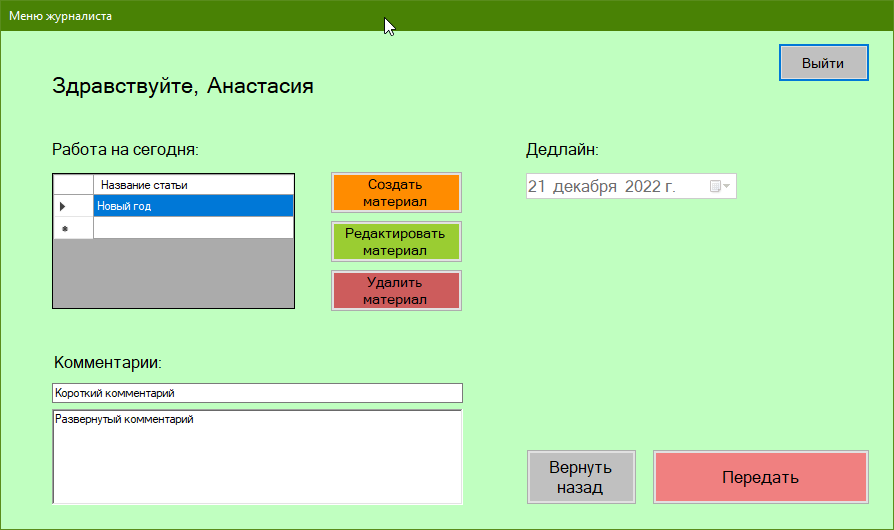


Рисунок 9 – Окно ЛК журналиста

Это нужно для того, чтобы разграничить функции, которые индивидуальны для каждой специальности. Например, только журналист может создавать новый материал и удалить существующий, у верстальщика таких кнопок, соответственно и функций, нет. Зато у верстальщика есть отдельная кнопка для перехода на страницу ресурсов. По нажатию этой кнопки осуществляется переход на внешний ресурс – веб-страницу с ресурсами, необходимыми для работы. Это может быть, к примеру, Яндекс Диск или Google Диск.

Но для всех доступны функции выхода из учетной записи по нажатию кнопки «Выйти», функции редактирования существующего материала, выбора из списка материала и отображение дедлайна его выполнения. Причем дедлайн будет отображаться в зависимости от того, кто сейчас в учетной записи – журналист или верстальщик. У каждого свой дедлайн и отображаться они будут соответственно разные. Также есть возможность посмотреть комментарии к выбранному материалу. Отображаются короткий вариант комментария и длинный, развернутый (рисунок 11). И, наконец, у каждого есть кнопки, отвечающие за передачу материала на уровень выше и уровень ниже. Причем при отправке сохраняются и написанные комментарии. При отправке к этим комментариям прикрепляется ID Пользователя, который их отправил. Также форма приветствует Пользователя по его имени.

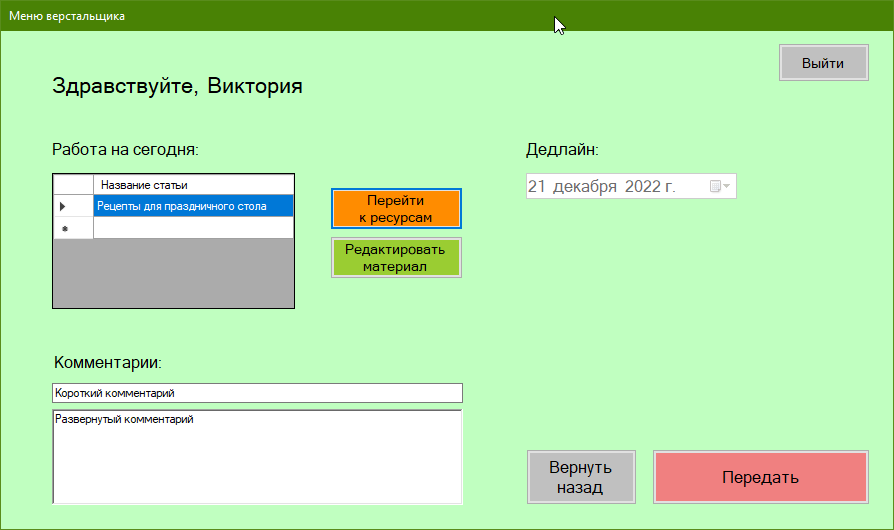


Рисунок 10 – Окно ЛК верстальщика

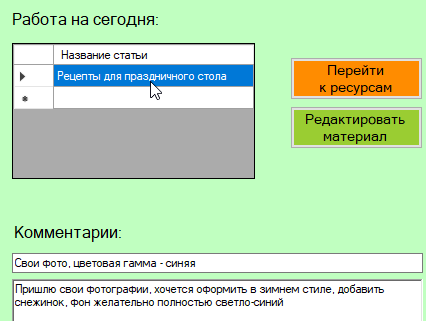


Рисунок 11 – Отображение комментариев

Также форма контролирует выбор материала, например, его нельзя отредактировать, если он не выбран в списке (рисунок 12):

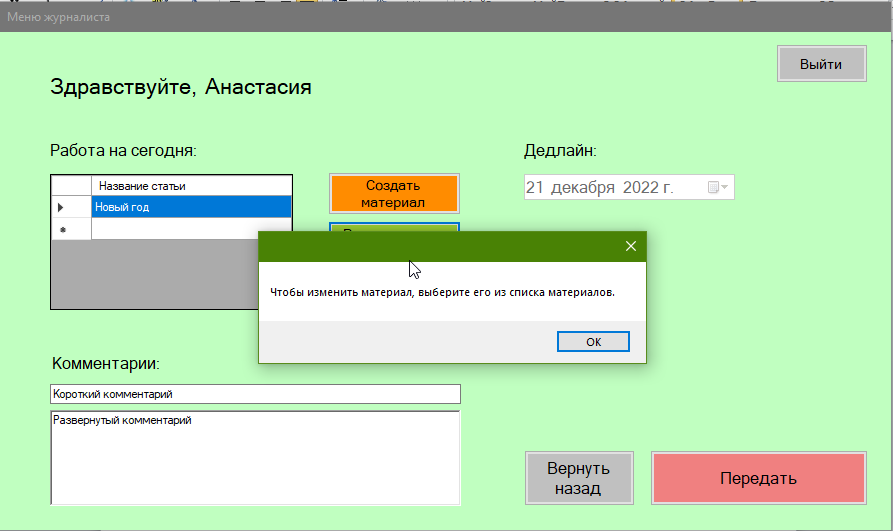


Рисунок 12 – Сообщение о том, что нужно выбрать материал в списке

При удалении материала появляется окно с подтверждением этого действия. При удалении список обновляется, и удаленный материал пропадает (рисунок 13).

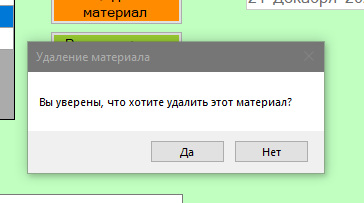


Рисунок 13 – Подтверждение удаления материала

При нажатии на кнопку «Создать материал» открывается окно редактирования с пустыми полями – по сути, никакой материал не редактируется, а создается новый (рисунок 14). Причем, также проверяются на заполнение поля (рисунок 15). Также для обеспечения ограничения целостности в дедлайнах нельзя выбрать дату, меньшую, чем день входа в ЛК (рисунок 16), к тому же добавляются еще 2 дня, чтобы исключить ситуацию, когда дедлайном ставят день входа в ЛК.

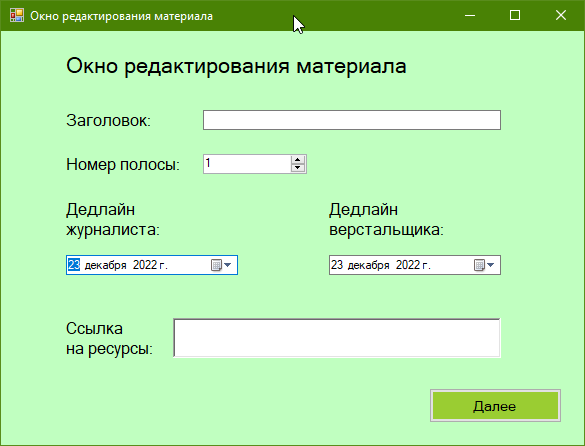


Рисунок 14 – Окно создания нового материала

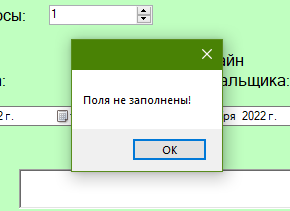


Рисунок 15 – Проверка на заполнение полей

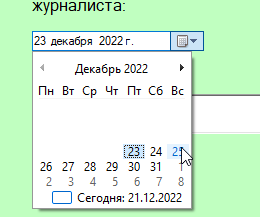


Рисунок 16 – Ограничение на выбор даты

После нажатия кнопки «Далее» введенные данные сохраняются, и осуществляется переход к следующему этапу создания материалу – выбор участников, ответственных за выполнение работы (рисунок 17).

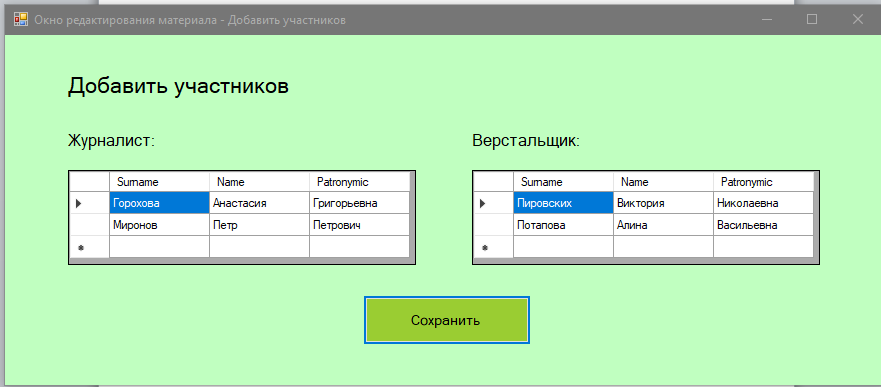


Рисунок 17 – Второй этап создания материала

Если пользователь не выбрал участников, показывается уведомление об этом (рисунок 18):

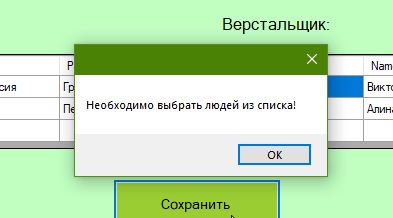


Рисунок 18 – Сообщение о необходимости выбрать участников

При нажатии кнопки «Сохранить» выбранные участники и ID материала записываются в таблицу Creates.

При редактировании формы из ЛК верстальщика все поля, кроме ссылки, недоступны (рисунок 19). При редактировании формы из ЛК журналиста все поля доступны (рисунок 20).

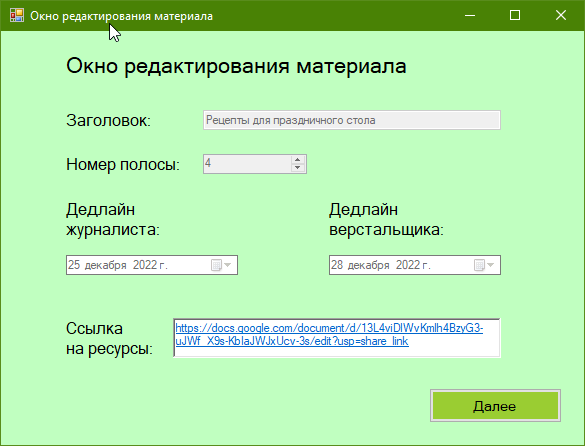


Рисунок 19 – Окно редактирования верстальщика

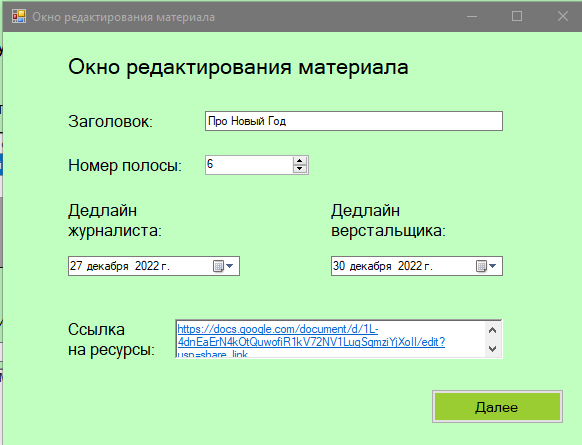


Рисунок 20 – Окно редактирования журналиста

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы был разработан интерфейс Пользователя участника студенческой газеты. Были разработаны:

* База данных для ИС «RIS\_PRIS»;
* 6 форм, составляющих интерфейс для работы с созданной БД.

Полученная Система позволяет осуществлять авторизацию, регистрацию Пользователей, разграничение доступа к данным, создание, изменение и удаление материалов, просмотр материалов, комментариев к ним, переход к ресурсам, относящимся к этим материалам.

Также Система позволяет выполнять главную задачу – осуществлять проход материала через все стадии разработки путем изменения его статуса.

Полученная система может быть еще усовершенствована, например, необходимо обеспечить закрепление комментария не только при передаче материла выше, но и при передаче его ниже. Также была идея реализовать отображение дедлайна в ЛК цветом, зависимости от того, сколько дней до него осталось. Еще можно было бы спроектировать отдельные формы для вышестоящих сотрудников. И в заключение, реализовать в интерфейсе сбор всех материалов при помощи работы с таблицей Release.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 ХалЭ., Джексон К., Дик Дж. Разработка и управление требованиями/Пер. с англ. – Электр. ресурс, 2005. – 229 с.: ил.

2 Коберн А. Современные методы описания функциональных требований к системам/Пер. с англ. – М.: Издательство "Лори", 2012. – 264 с.: ил.

3 Вигерс К., Битти Д. Разработка требований к программному обеспечению/Пер. с англ. – М.: Издательство "БХВ-Петербург", 2014. – 736 с.: ил.

4 Выявление и сбор требований к ПО — ultimate guide, URL:

<https://tproger.ru/articles/vyjavlenie-i-sbor-trebovanij-k-po-ultimate-guide/> (Дата обращения: 18-12-2022)

5 Как передавать данные между формами, URL: <https://www.cyberforum.ru/windows-forms/thread540379.html> (Дата обращения: 15-12-2022)

6 DateTimePicker и MonthCalendar, URL: <https://metanit.com/sharp/windowsforms/4.15.php> (Дата обращения: 19-12-2022)

7 C#: DataGridView и примеры работы с ним, URL: <https://blog.foolsoft.ru/c-datagridview-i-primery-raboty-s-nim/> (Дата обращения: 18-12-2022)

8 Обеспечение целостности данных, URL: <https://www.flenov.info/books/read/transact-sql/14> (Дата обращения: 17-12-2022)

9 Документация и требования, URL: <https://qa-guide.ru/forums/topic/testirovanie_dokumentacii/> (Дата обращения: 16-12-2022)

10 Пример написания функциональных требований к Enterprise-системе, URL: <https://habr.com/ru/post/245625/> (Дата обращения: 15-12-2022)

ПРИЛОЖЕНИЕ А

SQL-запрос создания таблиц базы данных

create table Comment (

ID\_comment int identity,

ID\_material int not null,

ID\_author int not null,

Short\_descr nvarchar(1024) not null,

Long\_descr nvarchar(1024) not null,

constraint PK\_COMMENT primary key (ID\_comment)

)

go

create table Creates (

ID\_member int not null,

ID\_material int not null,

constraint PK\_CREATES primary key (ID\_member, ID\_material)

)

go

create table Material (

ID\_material int identity,

ID\_release int null,

ID\_status int null,

Header nvarchar(1024) not null,

N\_page tinyint null,

Deadline\_write date not null,

Deadline\_design date not null,

constraint PK\_MATERIAL primary key (ID\_material)

)

go

create table Member (

ID\_member int identity,

ID\_spec int null,

ID\_position int not null,

Surname nvarchar(1024) not null,

Name nvarchar(1024) not null,

Patronymic nvarchar(1024) null,

Birthday date not null,

Link\_VK nvarchar(1024) not null,

Date\_of\_taking date not null,

Scholarship decimal(38) not null,

Login nvarchar(1024) not null,

Password nvarchar(1024) not null,

constraint PK\_MEMBER primary key (ID\_member)

)

go

create table Position (

ID\_position int not null,

Position\_name nvarchar(1024) not null,

constraint PK\_POSITION primary key (ID\_position)

)

go

create table Release (

ID\_release int identity,

N\_release int not null,

Month nvarchar(1024) not null,

Year int not null,

constraint PK\_RELEASE primary key (ID\_release)

)

go

create table Resourse (

ID\_resourse int identity,

ID\_material int null,

Link nvarchar(1024) not null,

constraint PK\_RESOURSE primary key (ID\_resourse)

)

go

create table Specialization (

ID\_spec int not null,

Spec\_name nvarchar(1024) not null,

constraint PK\_SPECIALIZATION primary key (ID\_spec)

)

go

create table Status (

ID\_status int not null,

Status\_name nvarchar(1024) not null,

constraint PK\_STATUS primary key (ID\_status)

)

go

alter table Comment

add constraint FK\_COMMENT\_MAY\_HAVE\_MATERIAL foreign key (ID\_material)

references Material (ID\_material)

go

alter table Creates

add constraint FK\_CREATES\_CREATES\_MEMBER foreign key (ID\_member)

references Member (ID\_member)

go

alter table Creates

add constraint FK\_CREATES\_CREATES2\_MATERIAL foreign key (ID\_material)

references Material (ID\_material)

go

alter table Material

add constraint FK\_MATERIAL\_IS\_IN\_STATUS foreign key (ID\_status)

references Status (ID\_status)

go

alter table Material

add constraint FK\_MATERIAL\_PRINT\_RELEASE foreign key (ID\_release)

references Release (ID\_release)

go

alter table Member

add constraint FK\_MEMBER\_HOLD\_POSITION foreign key (ID\_position)

references Position (ID\_position)

go

alter table Member

add constraint FK\_MEMBER\_POSSESSE\_SPECIALI foreign key (ID\_spec)

references Specialization (ID\_spec)

go

alter table Resourse

add constraint FK\_RESOURSE\_HAVE\_MATERIAL foreign key (ID\_material)

references Material (ID\_material)

go

Скрипты хранимых процедур, триггеров и ограничений целостности

--процедура выбора названия статьи

create proc SelectHeaders (@ID\_member int)

as

declare @ID\_spec int

declare @ID\_position int

select @ID\_spec = ID\_spec, @ID\_position = ID\_position from Member where ID\_member = @ID\_member

begin

if (@ID\_spec = 1 and @ID\_position = 5) --статус 1

select Material.ID\_material, Header from Creates, Material, Status

where Creates.ID\_material = Material.ID\_material and Material.ID\_status = Status.ID\_status

and Status.ID\_status = 1 and ID\_member = @ID\_member

if (@ID\_spec = 2 and @ID\_position = 5) --статус 3

select Material.ID\_material, Header from Creates, Material, Status

where Creates.ID\_material = Material.ID\_material and Material.ID\_status = Status.ID\_status

and Status.ID\_status = 3 and ID\_member = @ID\_member

else if (@ID\_spec = 1 and @ID\_position = 3) --статус 2

select Material.ID\_material, Header from Material, Status

where Material.ID\_status = Status.ID\_status and Status.ID\_status = 2

else if ((@ID\_spec = 1 and @ID\_position = 2) or (@ID\_spec = 2 and @ID\_position = 4)) --статус 4

select Material.ID\_material, Header from Material, Status

where Material.ID\_status = Status.ID\_status and Status.ID\_status = 4

else if (@ID\_spec = 1 and @ID\_position = 1) --статус 5

select Material.ID\_material, Header from Material, Status

where Material.ID\_status = Status.ID\_status and Status.ID\_status = 5

else return null

end

-- процедура возврата дедлайна

create proc SelectDeadline @ID\_material int, @ID\_member int

as

declare @ID\_spec int

select @ID\_spec = ID\_spec from Member where ID\_member = @ID\_member

begin

if (@ID\_spec = 1)

select Deadline\_write from Material where ID\_material = @ID\_material

else if (@ID\_spec = 2)

select Deadline\_design from Material where ID\_material = @ID\_material

else return null

end

-- процедура изменения статуса

create proc ChangeStatus @ID\_material int

as

declare @ID\_status int

select @ID\_status = ID\_status from Material where @ID\_material = ID\_material

begin

if (@ID\_status < 5)

update Material set ID\_status = ID\_status + 1 where @ID\_material = ID\_material

else if (@ID\_status >= 5)

update Material set ID\_status = 5 where @ID\_material = ID\_material

end

-- процедура изменения статуса - откат назад

create proc ChangeStatusBack @ID\_material int

as

declare @ID\_status int

select @ID\_status = ID\_status from Material where @ID\_material = ID\_material

begin

if (@ID\_status > 1)

update Material set ID\_status = ID\_status - 1 where @ID\_material = ID\_material

else if (@ID\_status <= 1)

update Material set ID\_status = 1 where @ID\_material = ID\_material

end

--процедура возврата ссылки на ресурс

create proc SelectLink @ID\_material int

as

select Link from Resourse where ID\_material = @ID\_material

-- процедура возврата комментариев

create proc SelectComments @ID\_material int, @ID\_member int

as

declare @ID\_position\_member int, @ID\_position\_author int

select @ID\_position\_member = ID\_position from Member where ID\_member = @ID\_member --ищу позицию того, кому нужен коммент

select @ID\_position\_author = ID\_position from Member, Creates, Material, Comment --ищу позицию того, кто оставил коммент

where Member.ID\_member = Creates.ID\_member and Creates.ID\_material = Material.ID\_material

and Material.ID\_material = Comment.ID\_material and Member.ID\_member = ID\_author

begin

if (@ID\_position\_author = 5 and @ID\_position\_member = 5)

select Short\_descr, Long\_descr from Comment where ID\_material = @ID\_material

else if (@ID\_position\_author > @ID\_position\_member)

select Short\_descr, Long\_descr from Comment where ID\_material = @ID\_material

end

--форма редактирования материала - окно 1

--процедура заполнения формы редактирования

create proc EditFormAdding @ID\_material int

as

select Header, N\_page, Deadline\_write, Deadline\_design, Link from Material, Resourse

where Resourse.ID\_material = Material.ID\_material and Material.ID\_material = @ID\_material

-- процедура обновления материала

create proc UpdateMaterial @ID\_material int, @Link nvarchar(1024), @Header nvarchar(1024), @N\_page int, @Deadline\_write date, @Deadline\_design date

as

begin

update Material set Header = @Header, N\_page = @N\_page, Deadline\_write = @Deadline\_write, Deadline\_design = @Deadline\_design

where ID\_material = @ID\_material

update Resourse set Link = @Link

where ID\_material = @ID\_material

end

-- процедура создания материала

create proc AddMaterial @Header nvarchar(1024), @N\_page int, @Deadline\_write date, @Deadline\_design date

as

begin

insert into Material (ID\_status, Header, N\_page, Deadline\_write, Deadline\_design) values (1, @Header, @N\_page, @Deadline\_write, @Deadline\_design)

end

-- процедура получения ID только что созданного материала

create proc LastMaterial

as

select max(ID\_material) from Material

--форма редактирования материала - окно 2

-- процедура возврата ответственных за материал

create proc MembersDoMaterial @ID\_material int, @ID\_spec int

as

begin

select Creates.ID\_member, Surname, Name, Patronymic from Creates, Member

where Creates.ID\_member = Member.ID\_member and ID\_material = @ID\_material and ID\_spec = @ID\_spec

end

--------------------------------------------------------------------------------------------

--триггер на удаление из бд материала, который привязан в creates

create trigger DeleteMaterial

on Material

instead of delete

as

delete from Creates where ID\_material in (select d.ID\_material from deleted d)

delete from Comment where ID\_material in (select d.ID\_material from deleted d)

delete from Material where ID\_material in (select d.ID\_material from deleted d)

--триггер на удаление из бд участника, который привязан в creates

create trigger DeleteMember

on Member

instead of delete

as

delete from Creates where ID\_member in (select d.ID\_member from deleted d)

delete from Member where ID\_member in (select d.ID\_member from deleted d)

--триггер на совпадение логинов

create trigger EqualLogins

on Member

instead of insert

as

if exists (select Login from Member

where Login in (select Login from inserted))

print 'Такой логин уже есть'

else insert into Member select i.ID\_spec, i.ID\_position, i.Surname, i.Name, i.Patronymic, i.Birthday, i.Link\_VK, i.Date\_of\_taking, i.Scholarship, i.Login, i.Password from inserted i

-- ограничение целостности

ALTER TABLE Material WITH NOCHECK

add check (Deadline\_write >= GETDATE())

ALTER TABLE Material WITH NOCHECK

add check (Deadline\_design >= GETDATE())

ALTER TABLE Member WITH NOCHECK

add check (Date\_Of\_Taking >= '01/09/1999')

Индексы

--Index: Possesse\_FK

create nonclustered index Possesse\_FK on Member (ID\_spec ASC)

go

--Index: Hold\_FK

create nonclustered index Hold\_FK on Member (ID\_position ASC)

go

--Index: May\_have\_FK

create nonclustered index May\_have\_FK on Comment (ID\_material ASC)

go

--Index: Creates\_FK

create nonclustered index Creates\_FK on Creates (ID\_member ASC)

go

--Index: Creates2\_FK

create nonclustered index Creates2\_FK on Creates (ID\_material ASC)

go

--Index: Is\_in\_FK

create nonclustered index Is\_in\_FK on Material (ID\_status ASC)

go

--Index: Print\_FK

create nonclustered index Print\_FK on Material (ID\_release ASC)

go

--Index: Have\_FK

create nonclustered index Have\_FK on Resourse (ID\_material ASC)

go

ПРИЛОЖЕНИЕ В

Код формы авторизации

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 RIS\_PRIS\_Kurs

{

public partial class authorizationForm : Form

{

string connectionString = @"Data Source=VIKTORIA\SQLEXPRESS;Initial Catalog=RIS\_PRIS\_test;Integrated Security=True";

//string connectionString = @"Data Source=VIKTORIA\SQLEXPRESS;Initial Catalog=RIS\_PRIS\_Kurs;Integrated Security=True";

SqlConnection cnn = new SqlConnection();

SqlCommand cmd = new SqlCommand();

string ID\_spec = "";

string Login = "";

string Password = "";

string Name = "";

string ID\_member;

public authorizationForm()

{

InitializeComponent();

// Создание подключения

SqlConnection connection = new SqlConnection(connectionString);

try

{

// Открываем подключение

connection.Open();

Console.WriteLine("Подключение открыто");

}

catch (SqlException ex)

{

Console.WriteLine(ex.Message);

}

//ПОТОМ УБРАТЬ!!!

textBoxLogin.Text = "Nastya";

textBoxPass.Text = "123";

}

private void authorizationForm\_Load(object sender, EventArgs e)

{

}

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

{

// переход к меню

string login = textBoxLogin.Text;

string pass = textBoxPass.Text;

cnn = new SqlConnection(connectionString);

cnn.Open();

cmd.Connection = cnn;

//команды

cmd.CommandText = "SELECT \* FROM Member WHERE Login = '" + login + "'";

//вставляем команду и получаем объект DataReader

SqlDataReader reader = cmd.ExecuteReader();

while (reader.Read())

{

ID\_spec = reader["ID\_spec"].ToString();

Login = reader["Login"].ToString();

Password = reader["Password"].ToString();

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

ID\_member = reader["ID\_member"].ToString();

}

//закрыть DataReader

reader.Close();

cnn.Close();

pass = pass.Replace(" ", "");

Password = Password.Replace(" ", "");

if (textBoxPass.Text != Password && textBoxLogin.Text == Login)

{

MessageBox.Show("Пароль введен неверно!");

textBoxPass.Text = "";

}

else if (textBoxLogin.Text != Login)

{

MessageBox.Show("Убедитесь, что логин введен верно, либо зарегистрируйтесь в системе.");

textBoxPass.Text = "";

registrationLink.ForeColor = Color.Green;

}

else if (textBoxPass.Text == Password && textBoxLogin.Text == Login)

{

// определяем формат ЛК

if (ID\_spec == "2")

{

menuDesignerForm menuDesignerForm = new menuDesignerForm(Login, Name, ID\_member);

menuDesignerForm.Show();

this.Visible = false;

}

else

{

menuWriterForm menuWriterForm = new menuWriterForm(Login, Name, ID\_member);

menuWriterForm.Show();

this.Visible = false;

}

}

}

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

{

// переход к регистрации DONE

registrationForm registrationForm = new registrationForm();

registrationForm.Show();

this.Visible = false;

}

private void authorizationForm\_FormClosing(object sender, FormClosingEventArgs e)

{

// чтобы из-за Visible=false самой первой формы прога не повисла

Application.Exit();

}

}

}

Код формы регистрации

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 RIS\_PRIS\_Kurs

{

public partial class registrationForm : Form

{

string connectionString = @"Data Source=VIKTORIA\SQLEXPRESS;Initial Catalog=RIS\_PRIS\_test;Integrated Security=True";

//string connectionString = @"Data Source=VIKTORIA\SQLEXPRESS;Initial Catalog=RIS\_PRIS\_Kurs;Integrated Security=True";

SqlConnection cnn = new SqlConnection();

SqlCommand cmd = new SqlCommand();

int ID\_spec;

int ID\_position;

public registrationForm()

{

InitializeComponent();

dateOfTaking.MinDate = new DateTime(1999, 9, 1); // 01/09/1999

// заполнение списка специализаций

SqlConnection connection1 = new SqlConnection(connectionString);

string query1 = "select ID\_spec, Spec\_name from Specialization";

SqlDataAdapter sqlDataAdapter1 = new SqlDataAdapter(query1, connectionString);

DataSet dataSet1 = new DataSet();

sqlDataAdapter1.Fill(dataSet1);

listSpec.DataSource = dataSet1.Tables[0];

// сокрытие ID

listSpec.Columns[0].Visible = false;

// настройка

DataGridViewColumn column1 = listSpec.Columns[1];

column1.Width = 167;

listSpec.ColumnHeadersVisible = false;

// заполнение списка позиций

SqlConnection connection = new SqlConnection(connectionString);

string query = "select ID\_position, Position\_name from Position";

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(query, connectionString);

DataSet dataSet = new DataSet();

sqlDataAdapter.Fill(dataSet);

listPosition.DataSource = dataSet.Tables[0];

// сокрытие ID

listPosition.Columns[0].Visible = false;

// настройка

DataGridViewColumn column = listPosition.Columns[1];

column.Width = 160;

listPosition.ColumnHeadersVisible = false;

}

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

{

string Surname = textBoxSurname.Text;

string Name = textBoxName.Text;

string Patronymic = textBoxPatronymic.Text;

string LinkVK = textBoxLinkVK.Text;

string Scholarship = textBoxScholarship.Text;

string Login = textBoxLogin.Text;

string Pass = textBoxPass.Text;

string PassConfirm = textBoxPassConfirm.Text;

DateTime Birthday = dateBirthday.Value;

DateTime Date\_OfTaking = dateOfTaking.Value;

if (textBoxSurname.Text == "" || textBoxName.Text == ""

|| textBoxLinkVK.Text == "" || textBoxScholarship.Text == "" || textBoxLogin.Text == ""

|| textBoxPass.Text == "" || textBoxPassConfirm.Text == "")

{

MessageBox.Show("Поля не заполнены!");

}

else

{

if (Pass != PassConfirm)

{

MessageBox.Show("Введенные пароли не совпадают!");

textBoxPass.Text = "";

textBoxPassConfirm.Text = "";

}

else

{

string sqlExpression = "insert into Member (ID\_spec, ID\_position, Surname, Name, Patronymic, Birthday, Link\_VK, Date\_of\_taking, Scholarship, Login, Password) values (" +

ID\_spec + "," + ID\_position + ",'" + Surname + "','" + Name + "','" + Patronymic + "','" + Birthday + "','" + LinkVK + "','" + Date\_OfTaking + "'," + Scholarship + ",'" + Login + "','" + Pass + "')";

using (SqlConnection cnn = new SqlConnection(connectionString))

{

cnn.Open();

SqlCommand cmd = new SqlCommand(sqlExpression, cnn);

int number = cmd.ExecuteNonQuery();

}

cnn.Close();

// переход к окну авторизации

this.Close();

authorizationForm authorizationForm = new authorizationForm();

authorizationForm.Show();

}

}

}

private void registrationForm\_Load(object sender, EventArgs e)

{

// убирает дефолтные кнопки Windows

this.ControlBox = false;

}

private void listSpec\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

try

{

if (listSpec.Rows[e.RowIndex].Cells[e.ColumnIndex].Value != null)

{

DataGridViewRow row = this.listSpec.Rows[e.RowIndex];

ID\_spec = Convert.ToInt32(row.Cells[0].Value);

}

}

catch

{

Console.WriteLine();

}

}

private void listPosition\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

try

{

if (listPosition.Rows[e.RowIndex].Cells[e.ColumnIndex].Value != null)

{

DataGridViewRow row = this.listPosition.Rows[e.RowIndex];

ID\_position = Convert.ToInt32(row.Cells[0].Value);

}

}

catch

{

Console.WriteLine();

}

}

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

{

// выход из окна регистрации

this.Close();

authorizationForm authorizationForm = new authorizationForm();

authorizationForm.Show();

}

}

}

Код формы ЛК журналиста

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 RIS\_PRIS\_Kurs

{

public partial class menuWriterForm : Form

{

string connectionString = @"Data Source=VIKTORIA\SQLEXPRESS;Initial Catalog=RIS\_PRIS\_test;Integrated Security=True";

//string connectionString = @"Data Source=VIKTORIA\SQLEXPRESS;Initial Catalog=RIS\_PRIS\_Kurs;Integrated Security=True";

SqlConnection cnn = new SqlConnection();

SqlCommand cmd = new SqlCommand();

int chosenID;

int memberID;

string link\_resourse;

public menuWriterForm(String Login, String Name, String ID\_member)

{

InitializeComponent();

nameHello.Text = Name;

memberID = Convert.ToInt32(ID\_member);

// вызов функции заполнения списка материалов

fillMaterialList(memberID);

}

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

{

//1 - режим полного редактирования у формы

editMaterialForm editMaterialForm = new editMaterialForm(chosenID, memberID, 1);

editMaterialForm.Show();

}

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

{

//0 - режим создания у формы

editMaterialForm editMaterialForm = new editMaterialForm(chosenID, memberID, 0);

editMaterialForm.Show();

}

private void menuWriterForm\_Load(object sender, EventArgs e)

{

this.ControlBox = false;

}

private void listBoxMaterial\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

try

{

if (listBoxMaterial.Rows[e.RowIndex].Cells[e.ColumnIndex].Value != null)

{

DataGridViewRow row = this.listBoxMaterial.Rows[e.RowIndex];

chosenID = Convert.ToInt32(row.Cells[0].Value);

// возврат комментариев

cnn = new SqlConnection(connectionString);

cnn.Open();

cmd.Connection = cnn;

cmd.CommandText = "exec SelectComments @ID\_material = " + chosenID + ", @ID\_member = " + memberID;

SqlDataReader reader1 = cmd.ExecuteReader();

string Short\_descr = "";

string Long\_descr = "";

while (reader1.Read())

{

Short\_descr = reader1["Short\_descr"].ToString();

Long\_descr = reader1["Long\_descr"].ToString();

}

reader1.Close();

cnn.Close();

shortDescription.Text = Short\_descr;

longDescription.Text = Long\_descr;

// возврат дедлайна

cnn = new SqlConnection(connectionString);

cnn.Open();

cmd.Connection = cnn;

cmd.CommandText = "exec SelectDeadline @ID\_material =" + chosenID + ", @ID\_member=" + memberID;

SqlDataReader reader2 = cmd.ExecuteReader();

string deadline\_material = "";

while (reader2.Read())

{

deadline\_material = reader2[0].ToString();

}

reader2.Close();

cnn.Close();

deadline.Value = DateTime.Parse(deadline\_material);

}

}

catch

{

Console.WriteLine();

}

}

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

{

// выход из учетной записи

this.Close();

authorizationForm authorizationForm = new authorizationForm();

authorizationForm.Show();

}

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

{

try

{

string short\_descr = shortDescription.Text;

string long\_descr = longDescription.Text;

// если человек не захотел оставлять комментарий

if (shortDescription.Text == "" || longDescription.Text == "")

{

string sqlExpression1 = "exec ChangeStatus @ID\_material = " + chosenID;

using (SqlConnection cnn = new SqlConnection(connectionString))

{

cnn.Open();

SqlCommand cmd = new SqlCommand(sqlExpression1, cnn);

int number = cmd.ExecuteNonQuery();

}

cnn.Close();

// вызов функции заполнения списка материалов

fillMaterialList(memberID);

}

else

{

string sqlExpression = "insert into Comment (ID\_material, ID\_author, Short\_descr, Long\_descr) values (" +

chosenID + "," + memberID + ",'" + short\_descr + "','" + long\_descr + "')";

using (SqlConnection cnn = new SqlConnection(connectionString))

{

cnn.Open();

SqlCommand cmd = new SqlCommand(sqlExpression, cnn);

int number = cmd.ExecuteNonQuery();

}

cnn.Close();

string sqlExpression1 = "exec ChangeStatus @ID\_material = " + chosenID;

using (SqlConnection cnn = new SqlConnection(connectionString))

{

cnn.Open();

SqlCommand cmd = new SqlCommand(sqlExpression1, cnn);

int number = cmd.ExecuteNonQuery();

}

cnn.Close();

// вызов функции заполнения списка материалов

fillMaterialList(memberID);

}

}

catch

{

MessageBox.Show("Чтобы передать материал, выберите его из списка материалов.");

}

}

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

{

DialogResult dialogResult = MessageBox.Show("Вы уверены, что хотите удалить этот материал?", "Удаление материала", MessageBoxButtons.YesNo);

if (dialogResult == DialogResult.Yes)

{

string sqlExpression = "delete from Material where ID\_material =" + chosenID;

using (SqlConnection cnn = new SqlConnection(connectionString))

{

cnn.Open();

SqlCommand cmd = new SqlCommand(sqlExpression, cnn);

int number = cmd.ExecuteNonQuery();

}

cnn.Close();

}

else if (dialogResult == DialogResult.No)

{

//пользователь нажал "нет"

}

// вызов функции заполнения списка материалов

fillMaterialList(memberID);

}

// функция заполнения списка материалов

private void fillMaterialList(int memberID)

{

// заполнение списка материалов в зависимости от прав доступа

SqlConnection connection = new SqlConnection(connectionString);

string query = "exec SelectHeaders @ID\_member =" + memberID;

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(query, connectionString);

DataSet dataSet = new DataSet();

sqlDataAdapter.Fill(dataSet);

listBoxMaterial.DataSource = dataSet.Tables[0];

// сокрытие ID

listBoxMaterial.Columns[0].Visible = false;

// настройка ширины

DataGridViewColumn column = listBoxMaterial.Columns[1];

column.Width = 200;

// название столбца

listBoxMaterial.Columns[1].HeaderText = "Название статьи";

}

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

{

try

{

string sqlExpression1 = "exec ChangeStatusBack @ID\_material = " + chosenID;

using (SqlConnection cnn = new SqlConnection(connectionString))

{

cnn.Open();

SqlCommand cmd = new SqlCommand(sqlExpression1, cnn);

int number = cmd.ExecuteNonQuery();

}

cnn.Close();

// вызов функции заполнения списка материалов

fillMaterialList(memberID);

}

catch

{

MessageBox.Show("Чтобы вернуть материал, выберите его из списка материалов.");

}

}

}

}

Код формы ЛК верстальщика

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 RIS\_PRIS\_Kurs

{

public partial class menuDesignerForm : Form

{

string connectionString = @"Data Source=VIKTORIA\SQLEXPRESS;Initial Catalog=RIS\_PRIS\_test;Integrated Security=True";

//string connectionString = @"Data Source=VIKTORIA\SQLEXPRESS;Initial Catalog=RIS\_PRIS\_Kurs;Integrated Security=True";

SqlConnection cnn = new SqlConnection();

SqlCommand cmd = new SqlCommand();

int chosenID;

int memberID;

string link\_resourse;

public menuDesignerForm(String Login, String Name, String ID\_member)

{

InitializeComponent();

nameHello.Text = Name;

memberID = Convert.ToInt32(ID\_member);

// вызов функции заполнения списка материалов

fillMaterialList(memberID);

}

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

{

try

{

System.Diagnostics.Process.Start(link\_resourse);

}

catch

{

MessageBox.Show("Не найден ресурс для этого материала!");

}

}

private void menuDesignerForm\_Load(object sender, EventArgs e)

{

// убирает дефолтные кнопки Windows

this.ControlBox = false;

}

private void listBoxMaterial\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

try

{

if (listBoxMaterial.Rows[e.RowIndex].Cells[e.ColumnIndex].Value != null)

{

DataGridViewRow row = this.listBoxMaterial.Rows[e.RowIndex];

chosenID = Convert.ToInt32(row.Cells[0].Value);

// возврат ссылки

cnn = new SqlConnection(connectionString);

cnn.Open();

cmd.Connection = cnn;

cmd.CommandText = "exec SelectLink @ID\_material =" + chosenID;

SqlDataReader reader = cmd.ExecuteReader();

while (reader.Read())

{

link\_resourse = reader["Link"].ToString();

}

reader.Close();

cnn.Close();

// возврат комментариев

cnn = new SqlConnection(connectionString);

cnn.Open();

cmd.Connection = cnn;

cmd.CommandText = "exec SelectComments @ID\_material = " + chosenID + ", @ID\_member = " + memberID;

SqlDataReader reader1 = cmd.ExecuteReader();

string Short\_descr = "";

string Long\_descr = "";

while (reader1.Read())

{

Short\_descr = reader1["Short\_descr"].ToString();

Long\_descr = reader1["Long\_descr"].ToString();

}

reader1.Close();

cnn.Close();

shortDescription.Text = Short\_descr;

longDescription.Text = Long\_descr;

// возврат дедлайна

cnn = new SqlConnection(connectionString);

cnn.Open();

cmd.Connection = cnn;

cmd.CommandText = "exec SelectDeadline @ID\_material =" + chosenID + ", @ID\_member=" + memberID;

SqlDataReader reader2 = cmd.ExecuteReader();

string deadline\_material = "";

while (reader2.Read())

{

deadline\_material = reader2[0].ToString();

}

reader2.Close();

cnn.Close();

deadline.Value = DateTime.Parse(deadline\_material);

}

}

catch

{

Console.WriteLine();

}

}

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

{

//2 - режим неполного редактирования у формы

editMaterialForm editMaterialForm = new editMaterialForm(chosenID, memberID, 2);

editMaterialForm.Show();

}

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

{

// выход из учетной записи

this.Close();

authorizationForm authorizationForm = new authorizationForm();

authorizationForm.Show();

}

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

{

try

{

string short\_descr = shortDescription.Text;

string long\_descr = longDescription.Text;

string sqlExpression = "insert into Comment (ID\_material, ID\_author, Short\_descr, Long\_descr) values (" +

chosenID + "," + memberID + ",'" + short\_descr + "','" + long\_descr + "')";

using (SqlConnection cnn = new SqlConnection(connectionString))

{

cnn.Open();

SqlCommand cmd = new SqlCommand(sqlExpression, cnn);

int number = cmd.ExecuteNonQuery();

}

cnn.Close();

string sqlExpression1 = "exec ChangeStatus @ID\_material = " + chosenID;

using (SqlConnection cnn = new SqlConnection(connectionString))

{

cnn.Open();

SqlCommand cmd = new SqlCommand(sqlExpression1, cnn);

int number = cmd.ExecuteNonQuery();

}

cnn.Close();

// вызов функции заполнения списка материалов

fillMaterialList(memberID);

}

catch

{

MessageBox.Show("Чтобы передать материал, выберите его из списка материалов.");

}

}

// функция заполнения списка материалов

private void fillMaterialList(int memberID)

{

// заполнение списка материалов в зависимости от прав доступа

SqlConnection connection = new SqlConnection(connectionString);

string query = "exec SelectHeaders @ID\_member =" + memberID;

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(query, connectionString);

DataSet dataSet = new DataSet();

sqlDataAdapter.Fill(dataSet);

listBoxMaterial.DataSource = dataSet.Tables[0];

// сокрытие ID

listBoxMaterial.Columns[0].Visible = false;

// настройка ширины

DataGridViewColumn column = listBoxMaterial.Columns[1];

column.Width = 200;

// название столбца

listBoxMaterial.Columns[1].HeaderText = "Название статьи";

}

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

{

try

{

string sqlExpression1 = "exec ChangeStatusBack @ID\_material = " + chosenID;

using (SqlConnection cnn = new SqlConnection(connectionString))

{

cnn.Open();

SqlCommand cmd = new SqlCommand(sqlExpression1, cnn);

int number = cmd.ExecuteNonQuery();

}

cnn.Close();

// вызов функции заполнения списка материалов

fillMaterialList(memberID);

}

catch

{

MessageBox.Show("Чтобы вернуть материал, выберите его из списка материалов.");

}

}

}

}

Код формы редактирования материалов

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 RIS\_PRIS\_Kurs

{

public partial class editMaterialForm : Form

{

string connectionString = @"Data Source=VIKTORIA\SQLEXPRESS;Initial Catalog=RIS\_PRIS\_test;Integrated Security=True";

//string connectionString = @"Data Source=VIKTORIA\SQLEXPRESS;Initial Catalog=RIS\_PRIS\_Kurs;Integrated Security=True";

SqlConnection cnn = new SqlConnection();

SqlCommand cmd = new SqlCommand();

int chosenID;

int memberID;

int typeForm;

public editMaterialForm(int chosenID\_param, int ID\_member, int Type)

{

InitializeComponent();

chosenID = chosenID\_param;

memberID = ID\_member;

typeForm = Type;

// для ограничения целостности не даем выбрать дату, меньше сегодня + еще пару дней

DateTime dateToday = DateTime.Today;

deadlineWrite.MinDate = dateToday.AddDays(2);

deadlineDesign.MinDate = dateToday.AddDays(2);

}

private void editMaterialForm\_Load(object sender, EventArgs e)

{

if (typeForm == 2) //2 - режим неполного редактирования у формы

{

// возврат значений полей для выбранного материала

cnn = new SqlConnection(connectionString);

cnn.Open();

cmd.Connection = cnn;

cmd.CommandText = "exec EditFormAdding @ID\_material =" + chosenID;

SqlDataReader reader = cmd.ExecuteReader();

string Header = "";

int N\_page = 1;

string deadline\_write = "";

string deadline\_design = "";

string link\_resourse = "";

while (reader.Read())

{

Header = reader["Header"].ToString();

N\_page = Convert.ToInt32(reader["N\_page"]);

deadline\_write = reader["Deadline\_write"].ToString();

deadline\_design = reader["Deadline\_design"].ToString();

link\_resourse = reader["Link"].ToString();

}

reader.Close();

cnn.Close();

// заполнение полей формы

textBoxHeader.Text = Header;

numericUpDownNpage.Value = N\_page;

try

{

deadlineWrite.Value = DateTime.Parse(deadline\_write);

deadlineDesign.Value = DateTime.Parse(deadline\_design);

}

catch

{

MessageBox.Show("Чтобы изменить материал, выберите его из списка материалов.");

this.Close();

}

textBoxLink.Text = link\_resourse;

// блокировка полей

textBoxHeader.Enabled = false;

numericUpDownNpage.Enabled = false;

deadlineWrite.Enabled = false;

deadlineDesign.Enabled = false;

}

else if (typeForm == 1) //1 - режим полного редактирования у формы

{

// возврат значений полей для выбранного материала

cnn = new SqlConnection(connectionString);

cnn.Open();

cmd.Connection = cnn;

cmd.CommandText = "exec EditFormAdding @ID\_material =" + chosenID;

SqlDataReader reader = cmd.ExecuteReader();

string Header = "";

int N\_page = 1;

string deadline\_write = "";

string deadline\_design = "";

string link\_resourse = "";

while (reader.Read())

{

Header = reader["Header"].ToString();

N\_page = Convert.ToInt32(reader["N\_page"]);

deadline\_write = reader["Deadline\_write"].ToString();

deadline\_design = reader["Deadline\_design"].ToString();

link\_resourse = reader["Link"].ToString();

}

reader.Close();

cnn.Close();

// заполнение полей формы

textBoxHeader.Text = Header;

numericUpDownNpage.Value = N\_page;

// потому что ругается из-за пустого списка

try

{

deadlineWrite.Value = DateTime.Parse(deadline\_write);

deadlineDesign.Value = DateTime.Parse(deadline\_design);

}

catch

{

MessageBox.Show("Чтобы изменить материал, выберите его из списка материалов.");

this.Close();

}

textBoxLink.Text = link\_resourse;

}

else if (typeForm == 0) //0 - режим создания у формы

{

// просто пустые поля

}

}

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

{

if (textBoxHeader.Text == "" || textBoxLink.Text == "")

{

MessageBox.Show("Поля не заполнены!");

}

else

{

if (typeForm == 2) //2 - режим неполного редактирования у формы

{

string update\_link = textBoxLink.Text;

string sqlExpression = "update Resourse set Link = '" + update\_link + "' where ID\_material = " + chosenID;

using (SqlConnection cnn = new SqlConnection(connectionString))

{

cnn.Open();

SqlCommand cmd = new SqlCommand(sqlExpression, cnn);

int number = cmd.ExecuteNonQuery();

}

cnn.Close();

editMaterialMemberForm editMaterialMemberForm = new editMaterialMemberForm(chosenID, memberID, 2, 0); // 0 просто чтоб не пусто

editMaterialMemberForm.Show();

this.Close();

}

else if (typeForm == 1) //1 - режим полного редактирования у формы

{

string update\_Header = textBoxHeader.Text;

int update\_N\_page = Convert.ToInt32(numericUpDownNpage.Value);

DateTime update\_deadline\_write = deadlineWrite.Value;

DateTime update\_deadline\_design = deadlineDesign.Value;

string update\_link = textBoxLink.Text;

string sqlExpression = "exec UpdateMaterial @ID\_material=" + chosenID + ", @Link='" + update\_link +

"', @Header='" + update\_Header + "', @N\_page=" + update\_N\_page + ", @Deadline\_write='" + update\_deadline\_write +

"', @Deadline\_design='" + update\_deadline\_design + "'";

using (SqlConnection cnn = new SqlConnection(connectionString))

{

cnn.Open();

SqlCommand cmd = new SqlCommand(sqlExpression, cnn);

int number = cmd.ExecuteNonQuery();

}

cnn.Close();

editMaterialMemberForm editMaterialMemberForm = new editMaterialMemberForm(chosenID, memberID, 1, 0); // 0 просто чтоб не пусто

editMaterialMemberForm.Show();

this.Close();

}

else if (typeForm == 0) //0 - режим создания у формы

{

string add\_Header = textBoxHeader.Text;

int add\_N\_page = Convert.ToInt32(numericUpDownNpage.Value);

DateTime add\_deadline\_write = deadlineWrite.Value;

DateTime add\_deadline\_design = deadlineDesign.Value;

string sqlExpression = "exec AddMaterial @Header='" + add\_Header + "', @N\_page=" + add\_N\_page + ", @Deadline\_write='" + add\_deadline\_write +

"', @Deadline\_design='" + add\_deadline\_design + "'";

using (SqlConnection cnn = new SqlConnection(connectionString))

{

cnn.Open();

SqlCommand cmd = new SqlCommand(sqlExpression, cnn);

int number = cmd.ExecuteNonQuery();

cnn.Close();

// возврат id только что созданного материала

//cnn = new SqlConnection(connectionString);

cnn.Open();

cmd.Connection = cnn;

cmd.CommandText = "exec LastMaterial";

SqlDataReader reader = cmd.ExecuteReader();

int idLastMaterial = 0;

while (reader.Read())

{

idLastMaterial = Convert.ToInt32(reader[0]);

}

reader.Close();

cnn.Close();

editMaterialMemberForm editMaterialMemberForm = new editMaterialMemberForm(chosenID, memberID, 0, idLastMaterial);

editMaterialMemberForm.Show();

this.Close();

}

}

}

}

}

}

Код формы редактирования материалов – Добавление участников

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 RIS\_PRIS\_Kurs

{

public partial class editMaterialMemberForm : Form

{

string connectionString = @"Data Source=VIKTORIA\SQLEXPRESS;Initial Catalog=RIS\_PRIS\_test;Integrated Security=True";

//string connectionString = @"Data Source=VIKTORIA\SQLEXPRESS;Initial Catalog=RIS\_PRIS\_Kurs;Integrated Security=True";

SqlConnection cnn = new SqlConnection();

SqlCommand cmd = new SqlCommand();

int chosenID;

int memberID;

int typeForm;

int idLastMaterial;

int chosenWriter;

int chosenDesigner;

public editMaterialMemberForm(int chosenID\_param, int ID\_member, int Type, int LastMaterialID)

{

InitializeComponent();

chosenID = chosenID\_param;

memberID = ID\_member;

typeForm = Type;

idLastMaterial = LastMaterialID;

if (typeForm == 2) // 2 - режим Enabled = false

{

// заполнение списка журналистов, ответственных за материал

SqlConnection connection = new SqlConnection(connectionString);

string query = "exec MembersDoMaterial @ID\_material =" + chosenID + ", @ID\_spec=" + 1;

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(query, connectionString);

DataSet dataSet = new DataSet();

sqlDataAdapter.Fill(dataSet);

listWriters.DataSource = dataSet.Tables[0];

// сокрытие ID

listWriters.Columns[0].Visible = false;

// доступность столбца

listWriters.Enabled = false;

connection.Close();

// заполнение списка верстальщиков, ответственных за материал

SqlConnection connection1 = new SqlConnection(connectionString);

string query1 = "exec MembersDoMaterial @ID\_material =" + chosenID + ", @ID\_spec=" + 2;

SqlDataAdapter sqlDataAdapter1 = new SqlDataAdapter(query1, connectionString);

DataSet dataSet1 = new DataSet();

sqlDataAdapter1.Fill(dataSet1);

listDesigners.DataSource = dataSet1.Tables[0];

// сокрытие ID

listDesigners.Columns[0].Visible = false;

// доступность столбца

listDesigners.Enabled = false;

connection1.Close();

}

else if (typeForm == 1) // 1 - режим вставки только ответственных за материал

{

// заполнение списка журналистов, ответственных за материал

SqlConnection connection = new SqlConnection(connectionString);

string query = "exec MembersDoMaterial @ID\_material =" + chosenID + ", @ID\_spec=" + 1;

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(query, connectionString);

DataSet dataSet = new DataSet();

sqlDataAdapter.Fill(dataSet);

listWriters.DataSource = dataSet.Tables[0];

// сокрытие ID

listWriters.Columns[0].Visible = false;

connection.Close();

// заполнение списка верстальщиков, ответственных за материал

SqlConnection connection1 = new SqlConnection(connectionString);

string query1 = "exec MembersDoMaterial @ID\_material =" + chosenID + ", @ID\_spec=" + 2;

SqlDataAdapter sqlDataAdapter1 = new SqlDataAdapter(query1, connectionString);

DataSet dataSet1 = new DataSet();

sqlDataAdapter1.Fill(dataSet1);

listDesigners.DataSource = dataSet1.Tables[0];

// сокрытие ID

listDesigners.Columns[0].Visible = false;

connection1.Close();

}

else if (typeForm == 0) // 0 - режим вставки всех участников

{

// заполнение списка журналистов

SqlConnection connection = new SqlConnection(connectionString);

string query = "select ID\_member, Surname, Name, Patronymic from Member where ID\_spec = 1 and ID\_position = 5";

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(query, connectionString);

DataSet dataSet = new DataSet();

sqlDataAdapter.Fill(dataSet);

listWriters.DataSource = dataSet.Tables[0];

// сокрытие ID

listWriters.Columns[0].Visible = false;

connection.Close();

// заполнение списка верстальщиков

SqlConnection connection1 = new SqlConnection(connectionString);

string query1 = "select ID\_member, Surname, Name, Patronymic from Member where ID\_spec = 2 and ID\_position = 5";

SqlDataAdapter sqlDataAdapter1 = new SqlDataAdapter(query1, connectionString);

DataSet dataSet1 = new DataSet();

sqlDataAdapter1.Fill(dataSet1);

listDesigners.DataSource = dataSet1.Tables[0];

// сокрытие ID

listDesigners.Columns[0].Visible = false;

connection1.Close();

}

}

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

{

// добавление/обновление участников у материала

if (typeForm == 1)

{

//по id нахожу материал и ему insert в creates участников, которые выбраны

// это изменение уже существующего то есть я делаю update где idmaterial = idLastMaterial и перезаписываю номера участников

string sqlExpression = "update Creates set ID\_member = " + chosenWriter + " where ID\_material = " + idLastMaterial;

using (SqlConnection cnn = new SqlConnection(connectionString))

{

cnn.Open();

SqlCommand cmd = new SqlCommand(sqlExpression, cnn);

int number = cmd.ExecuteNonQuery();

}

cnn.Close();

string sqlExpression1 = "update Creates set ID\_member = " + chosenDesigner + " where ID\_material = " + idLastMaterial;

using (SqlConnection cnn = new SqlConnection(connectionString))

{

cnn.Open();

SqlCommand cmd = new SqlCommand(sqlExpression1, cnn);

int number = cmd.ExecuteNonQuery();

}

cnn.Close();

this.Close();

}

else if (typeForm == 0)

{

// 0 это создание нового то есть нужно в creates записать idmaterial = idLastMaterial и id участников

//idLastMaterial

string sqlExpression = "insert into Creates (ID\_member, ID\_material) values (" + chosenWriter + "," + idLastMaterial + ")";

using (SqlConnection cnn = new SqlConnection(connectionString))

{

cnn.Open();

SqlCommand cmd = new SqlCommand(sqlExpression, cnn);

try

{

int number = cmd.ExecuteNonQuery();

cnn.Close();

}

catch

{

MessageBox.Show("Необходимо выбрать людей из списка!");

}

}

string sqlExpression1 = "insert into Creates (ID\_member, ID\_material) values (" + chosenDesigner + "," + idLastMaterial + ")";

using (SqlConnection cnn1 = new SqlConnection(connectionString))

{

cnn1.Open();

SqlCommand cmd1 = new SqlCommand(sqlExpression1, cnn1);

try

{

int number = cmd1.ExecuteNonQuery();

cnn.Close();

}

catch

{

Console.WriteLine();

}

}

this.Close();

}

this.Close();

}

private void editMaterialMemberForm\_Load(object sender, EventArgs e)

{

}

private void listWriters\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

try

{

if (listWriters.Rows[e.RowIndex].Cells[e.ColumnIndex].Value != null)

{

DataGridViewRow row = this.listWriters.Rows[e.RowIndex];

chosenWriter = Convert.ToInt32(row.Cells[0].Value);

}

}

catch

{

Console.WriteLine();

}

}

private void listDesigners\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

try

{

if (listDesigners.Rows[e.RowIndex].Cells[e.ColumnIndex].Value != null)

{

DataGridViewRow row = this.listDesigners.Rows[e.RowIndex];

chosenDesigner = Convert.ToInt32(row.Cells[0].Value);

}

}

catch

{

Console.WriteLine();

}

}

}

}