|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ | | | | | | | | |
|  | | |  | | |  | | |
| ФГБОУ ВО «Пермский государственный  национальный исследовательский университет» | | | | | | | | |
|  | | |  | | |  | | |
|  | | ОТЧЕТ  по лабораторной работе № 3 «Использование форм для модификации данных» по дисциплине «Учебная практика по БД и СУБД» | | | | |  | |
|  | | |  | | |  | | |
|  | Работу выполнил  студент гр. \_\_\_\_\_\_\_  Зимин И. В.\_\_\_\_\_\_\_\_\_  (подпись)  «\_\_\_» \_\_\_\_\_\_\_\_\_ 2018 | | |  | Проверил  \_\_\_\_\_\_ кафедры МОВС  (доц., ст. преп., асс.)  Постаногов И. С.\_\_\_\_\_  (подпись)  «\_\_\_» \_\_\_\_\_\_\_\_ 2018 | | |  |
|  |  | | |  |  | | |  |
| Пермь 2018 | | | | | | | | |

1 Постановка задачи

Написать программу, которая позволяет просматривать, добавлять, изменять и удалять записи в таблице зарегистрированных пользователей. Для просмотра должен использоваться элемент управления ListView. Для добавления и изменения записей должны использоваться формы. Формы должны быть защищены от SQL-инъекций.

Провести тестирование корректности работы программы.

Написать отчёт о проделанной работе.

2 Анализ предметной области

Для работы приложения требуется всего одна таблица «Users», в ней хранятся данные о пользователях:

* Логин: хранится в открытом виде
* Пароль: хранится в виде хэш-кода.
* Соль: хранится в виде строки.
* Дата регистрации пользователя: хранится в формате гггг-мм-дд.

Структура таблицы представлена на таблице 2.

Поле Login уникально, оно имеет тип nvarchar(50), так как длина логина конкретного пользователя может существенно различаться по длине, но он должен быть короче 50 символов. Это ограничение на длину я наложил исходя из здравых рассуждений, что логин вряд ли кому-либо вздумается вводить настолько длинный логин, чтобы превысить 50 данное ограничение.

Поле Password имеет постоянную длину 40 символа, так как для хеширования пароля использовалась хэш-функция SHA1, выходной поток которой составляет 20 байт и после конвертации в строку принимает размер 40 символов.

Поле Salt имеет постоянную длину 40 символа и создавалась программным путём с помощью криптостойкой последовательности байтов.

Поле RegistrationDate имеет соответствующий тип date, так как поле содержит нечто иное, как конкретную дату.

Важным аспектом является то, что все поля не допускают значение NULL, так как отсутствие хоть одного из них приведёт к неправильной работе программы.

Таблица 2. - Структура таблицы «Users»

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Разрешить NULL |
| Login | nvarchar(50) | нет |
| Password | char(40) | нет |
| Salt | char(40) | нет |
| RegistrationDate | date | нет |

3 Описание функционала разработанной программы

Приложение представляет собой две формы: форма отображения и форма изменения/добавления записей о пользователях.

Форма отображения записей содержит: меню с одной вкладкой «Правка», в которой есть команды добавить, изменить или удалить записи, а также место под списковое представление данных о пользователях, которое будет выглядеть как таблица, названия столбцов которой соответствуют названиям столбцов в таблице «Users», за исключением соли, она нам не понадобится, так как даже факт о том, что мы знаем соль нам не поможет получить каких-либо данных о не хешированном пароле. Данная форма представлена на рисунке 3.1

Форма изменения/добавления записей о пользователях содержит три поля ввода: логина, пароля и даты, а также кнопку «Добавить/изменить» в зависимости от выбранных действий. Данная форма изображена на рисунке 3.2.

При запуске приложения отображается список зарегистрированных пользователей. Приложение позволяет:

- Добавить нового пользователя

- Изменить одну/несколько записей. Причем изменять можно любое количество полей, при этом остальные остаются неизменными

- Удалить одну/несколько записей

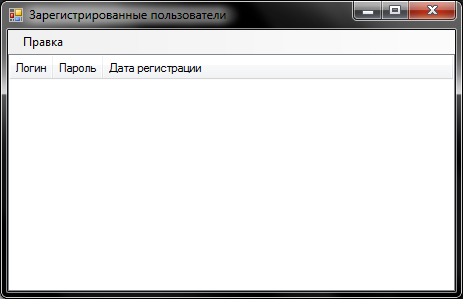


Рисунок 3.1 - Форма регистрации

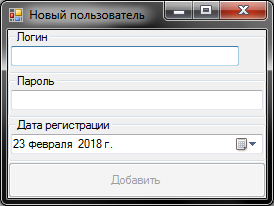


Рисунок 3.2 – Форма аутентификации

4 Описание алгоритма

* Шаг 1. Отображаем данные о уже зарегистрированных пользователях
* Шаг 2. Обработка нажатия элемента правки «Добавить».

- При нажатии элемента правки «Добавить» отображаем форму добавления/изменения данных о пользователе, при этом надпись на кнопке формы будет «Добавить».

- Проверяем, есть ли уже пользователь в таблице с введенным логином. Для этого, при каждом вводе символа в поле просматриваем записи в таблице. При выполнении данной проверки, должно возвращаться число – количество идентификаторов записей, у которых значение поля логина равняется введенном логину в приложении. Если вернулось число больше нуля, то выводим ошибку «Логин занят» и блокируем кнопку добавления, иначе – запускаем проверку на корректность введенных данных (число символов в каждом поле больше нуля, логин не занят).

- Если все поля заполнены корректно, то разблокируем кнопку «Добавить»

* Шаг 3. Обработка нажатия элемента правки «Изменить».

- Для изменения данных необходимо выделить нужных нам пользователей. Формочки для изменения отдельного пользователя будут отображаться по очереди, начиная с первой выделенной.

- При нажатии элемента правки «Изменить» отображаем форму добавления/изменения данных о пользователе, при этом надпись на кнопке будет соответствовать действию.

- Предоставляем возможность изменить любое поля, заполнив поля формы текущими данными, поле ввода пароля можно оставить пустым если оно не подлежит изменению.

- Если пользователь ввел существующий логин, то проделываем действия, описанные в шаге 2.

- Если все поля заполнены корректно, то разблокируем кнопку «Изменить»

* Шаг 4. Обработка нажатия элемента правки «Удалить».

- Для изменения данных необходимо выделить нужных нам пользователей. И нажать в меню правки кнопку «Удалить».

5 Описание SQL-запросов

В программе используется шесть SQL-запросов. Два SELECT-запроса и по оному INSERT и DELETE запросу и два UPDATE-запроса.

Первый представляет из себя SELECT – запрос, который необходим для отображения данных о зарегистрированных пользователях. Соль отображать не требуется, поэтому в выборку она не была включена. Запрос выполняется автоматически при запуске приложения и имеет следующий вид:

SELECT login, password, registerDate FROM Users

Второй запрос требуется для проверки корректности ввода логина, выглядит он следующим образом:

SELECT COUNT(\*) FROM Users WHERE login = @currentLogin

Данный запрос позволяет определить количество записей в БД с данным логином и так как значение логина в БД уникально запрос может вернуть либо 1, это будет означать, что данный логин в БД уже существует и его нельзя использовать для регистрации, либо 0, это будет означать, что логина в БД нет, и им можно воспользоваться.

INSERT – запрос имеет следующий вид:

INSERT INTO Users(Login, Password, Salt, RegistrationDate)

VALUES (@curLogin, @curPassword, @curSalt, @curRegDate)

Данный запрос выполняется при нажатии кнопки «Добавить» и вставляет в БД полученный из диалогового окна логин, сгенерированную соль, хэш пароля и текущую дату.

Для того чтобы изменить данные о пользователе потребовалось 2 запроса. Первый из них выполняется в случае, когда пароль не подлежит изменению. Он имеет следующий вид:

UPDATE Users SET login = @Login, registerDate = @RegistrationDate

WHERE login = @oldLogin

Данный запрос был выделен в отдельный так как мы, даже зная соль и хэш пароля не сможем оставить неизменным исходный пароль, обновляя все поля, это связано с тем, что нам мало этих данных для получения пароля в не хешированном виде. В остальных случаях выполняется запрос в котором подлежат обновлению все поля таблицы «Users» и так как этот запрос будет выполняться в случае изменения пароля, то необходимо генерировать новую соль. Запрос имеет вид:

UPDATE Users SET login = @Login, password = @Password,

salt = @Salt, registerDate = @RegistrationDate

WHERE login = @oldLogin

Запрос на удаление данных о пользователе выглядит следующим образом:

DELETE FROM Users WHERE login = @Login

То есть мы удаляем целиком строку из БД, значение логина в которой совпадает со значением логина в выделенной строке на форме.

6 Реализация на C#

Для изменения и удаления аккаунтов должны быть выбраны пользователи. При выборе нескольких строк изменение\удаление происходит последовательно, т.е. по одному.

При нажатии на добавление или изменение появляется форма, которая в которой вы можете задать логин, пароль и дату регистрации.

Добавление и изменение обрабатываются при нажатии кнопки ОК.

Хэширование пароля происходит в базе данных.

Для преобразования даты из базы данных к строковому типу используется явное приведение к типу DataTime, далее вызывается метод ToLongDateString(). Он преобразует в строку.

Добавление реализуется первым описанным выше запросом. При попытке создания аккаунта с уже существующим логином будет выведено предупреждение.

Изменение реализуется следующим образом:

Логин меняется, если введённый логин отличается от того, который был в базе данных ранее.

Поле пароля изначально пустое, если вы вводите туда какие-либо символы, то изменяется текущий пароль. Соль создается заново.

При загрузке формы для изменения выставляется значение даты из базы данных. Дата обновляется, если она не совпадает с той, что хранится в базе данных.

Для добавления шапки в ListView используется команда добавления столбца lvUser.Columns.Add("Название\_столбца");

Для добавления нового элемента в ListView мы создаём новый объект типа ListViewItem, который содержит массив строковых переменных. Каждый элемент массива отвечает за столбец в ListView.

Для изменения выбранной строки используется команда, которая использует текст столбца.

selectedItem.SubItems[Номер\_столбца].Text = «Новое значение текста»

Для удаления выбранного элемента используется метод Remove класса SelectedItem.

selectedItem.Remove();

Формирование строки для подключения к базе данных выполняется с помощью класса SqlConnectionStringBuilder.

Далее нам нужно открыть подключение к базе данных при помощи класса SqlConnection.

Описание команды при помощи класса SqlCommand.

SqlCommands Command = new SqlCommand()

Для считывания данных из БД используется объект SqlCommand. Используется параметризированный ввод для защиты от SQL инъекций.

Выборка данных из базы данных производится методом ExecuteReader().

Удаление, изменение и добавление вызываются с помощью метода ExecuteNonQuery().

Проверка на уникальность логина выполняется методом ExecuteScalar(), так как он возвращает количество найденных записей, если оно равно 0, то логин уникален.

Для получения случайной соли используется метод GetBytes класса RNGCryptoServiceProvider. Далее полученный массив байтов преобразуется к строке и возвращается в основную программу.

С помощью метода BitConverter мы преобразуем массив байтов к строке, благодаря Replace мы убираем все символы «-».

7 Тестирование программы

Чтобы протестировать корректность работы программы можно все тесты разбить на 3 класса.

Класс №1: Тесты для проверки добавления нового пользователя

1. Поля ввода логина и пароля пусты
2. Пусто одно из полей
3. Ввод несуществующего логина и несуществующего пароля
4. Ввод существующего логина и несуществующего пароля
5. Ввод несуществующего логина, но существующего пароля.\
6. Ввод логина длиннее 50 символов.

Класс №2: Тесты для проверки изменения данных о пользователе

1. Изменить логин на существующий
2. Увеличить длину логина до длины, превышающей 50 символов.
3. Изменить только логин или только пароль, или только дату
4. Изменение нескольких записей

Класс №3: Тесты для проверки удаления данных

1) Удаление одной записи

2) Удаление нескольких записей

Для проверки данного набора тестов достаточно наполнить БД двумя строками:

|  |  |  |  |
| --- | --- | --- | --- |
| Login | Password | Salt | Date |
| Fred | 728F5B79CD46D312D8CA  A461A33BDB28EA0C578E | 945F95BCC4BDC8F08E93  B151561D67AF84C84692 | 2018-02-02 |
| Hendrik | 8SDFGE9CD46D312D8CA  6F5F95BCC4BDC8F08E93 | 2DAF92A4FBDC8F08E93  AC595BB4BDC8F08E93 | 2018-03-02 |

При вводе логина, длина которого превышала 50 символов была выведена соответствующая ошибка.

Программа на данных тестах отработала корректно. При необходимости были выведены соответствующие сообщения.