Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Дисциплина: Системное программное обеспечение вычислительных машин

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

на тему

АВТОНОМНЫЙ МЕНЕДЖЕР ПАРОЛЕЙ ДЛЯ ОС LINUX

БГУИР КП 1–40 02 01 01 305 ПЗ

Студент: А. В. Гуринович

Руководитель: А. А. Климец

МИНСК 2021

Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет: ФКСиС. Кафедра: ЭВМ.

Специальность: 40 02 01 «Вычислительные машины, системы и сети».

Специализация: 40 02 01-01 «Проектирование и применение локальных компьютерных сетей».

УТВЕРЖДАЮ

Заведующий кафедрой ЭВМ

\_\_\_\_\_\_\_\_\_ Б.В.Никульшин

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2021 г.

ЗАДАНИЕ

по курсовому проекту студента

Гуриновича Андрея Викторовича

**1** Тема проекта: «Автономный менеджер паролей для ОС Linux».

**2** Срок сдачи студентом законченного проекта: 20 мая 2021 г.

**3** Исходные данные к проекту:

**3.1** Язык программирования: С++.

**3.2** Среда разработки: Xcode, Qt Creator.

**3.3** Метод шифрования: AES (FIPS 197).

**3.4** Операционная система: Linux (macOS).

**4** Содержание пояснительной записки (перечень подлежащих разработке

вопросов):

Введение 1. Обзор литературы. 2. Системное проектирование.   
3. Функциональное проектирование. 4. Разработка программных модулей.   
5. Программа и методика испытаний. 6. Руководство пользователя. Заключение. Список использованных источников. Приложения.

**5** Перечень графического материала (с точным указанием обязательных

чертежей):

**5.1** Автономный менеджер паролей для ОС Linux.

Схема структурная.

**5.2** Автономный менеджер паролей для ОС Linux.

Диаграмма классов.

**5.3** Автономный менеджер паролей для ОС Linux.

Схема программы.

КАЛЕНДАРНЫЙ ПЛАН

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование этапов  дипломного проекта | Объем  этапа,  % | Срок выполнения этапа | Примечания |
| Подбор и изучение литературы. Сравнение аналогов. | 10 | 23.03 – 05.04 |  |
| Структурное проектирование | 15 | 05.04 – 12.04 |  |
| Функциональное проектирование | 25 | 12.04 – 24.04 |  |
| Разработка программных модулей | 20 | 24.04 – 08.05 |  |
| Программа и методика испытаний | 10 | 8.05 – 15.05 |  |
| Оформление пояснительной записки | 15 | 20.05 – 30.05 |  |

Дата выдачи задания: 22.02.2021

Руководитель А. А. Климец

ЗАДАНИЕ ПРИНЯЛ К ИСПОЛНЕНИЮ \_\_\_\_\_\_\_\_

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 5](#_Toc72874379)

[1 ОБЗОР ЛИТЕРАТУРЫ 6](#_Toc72874380)

[1.1 Обзор аналогов 6](#_Toc72874381)

[1.2 Обзор стандарта шифрования Advanced Encryption Standard 8](#_Toc72874382)

[1.3 Обзор Qt 8](#_Toc72874383)

[2 СИСТЕМНОЕ ПРОЕКТИРОВАНИЕ 10](#_Toc72874384)

[2.1 Обзор используемых сторонних компонентов 10](#_Toc72874385)

[2.2 Структура программы 10](#_Toc72874386)

[3 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ 12](#_Toc72874387)

[3.1 Класс AES 12](#_Toc72874388)

[3.2 Класс Bundle 12](#_Toc72874389)

[3.3 Класс BundleUndo 12](#_Toc72874390)

[3.4 Класс EditWindow 12](#_Toc72874391)

[3.5 Класс File 12](#_Toc72874392)

[3.6 Класс PasswordWindow 13](#_Toc72874393)

[3.7 Класс Settings 13](#_Toc72874394)

[3.8 Класс StartWindow 13](#_Toc72874395)

[4 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ 15](#_Toc72874396)

[4.1 Шифрование алгоритмом AES 15](#_Toc72874397)

[4.2 Расшифровка алгоритмом AES 15](#_Toc72874398)

[4.3 Асинхронная запись в файл 16](#_Toc72874399)

[4.4 Асинхронное чтение из файла 16](#_Toc72874400)

[5 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 18](#_Toc72874401)

[ЗАКЛЮЧЕНИЕ 24](#_Toc72874402)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 25](#_Toc72874403)

[ПРИЛОЖЕНИЕ А 26](#_Toc72874404)

[ПРИЛОЖЕНИЕ Б 27](#_Toc72874405)

[ПРИЛОЖЕНИЕ В 28](#_Toc72874406)

[ПРИЛОЖЕНИЕ Г 29](#_Toc72874407)

[ПРИЛОЖЕНИЕ Д 30](#_Toc72874408)

# ВВЕДЕНИЕ

Развитие информатизации и интеграция информационных технологии в жизнь современного общества привела к постоянной необходимости взаимодействия человека и различных информационных систем. К примеру, для оплаты жилищно-коммунальных услуг используют интернет-банкинг, для заказа товаров – различные Интернет-магазины, для обмена документами используется электронная почта.

Все эти информационные системы объединяет необходимость идентификации конкретного человека, будь то потребитель услуг, администратор или менеджер. Как правило такая идентификация осуществляется через связку имени пользователя и пароля. Именем пользователя чаще всего бывает адрес электронной почты, номер телефона или придуманное человеком сочетание символов (никнейм). В качестве пароля используют сочетание символов известное только данному человеку.

Эта мера безопасности обеспечивает достаточную защиту от взлома аккаунтов (идентификация злоумышленником под видом другого человека), однако она также создаёт некоторые проблемы.

Учитывая необходимость человека постоянно вступать в контакт с разными информационными системами и проблематичность запоминания разных имён пользователя и паролей, он, как правило, будет использовать одинаковые имена пользователя и пароли для разных систем. Из-за такой политики, при получении сочетания имени пользователя и пароля для одной системы, злоумышленник может использовать их же в других системах, что позволит ему получить доступ к большему количество систем нелегально.

Для уменьшения вероятности таких случаев рекомендуется использовать разные имена пользователей и пароли в разных информационных системах, однако это приводит к обострению проблемы запоминания (хранения) этих сочетаний. Для минимизации этой проблемы существуют специальных программы для хранения связок имён пользователя и паролей. Менеджеры паролей.

Такие программы имеют собственную базу данных, которая располагается либо локально на устройстве пользователя, либо на сервере. Для получения доступа к ним пользователю необходимо идентифицировать себя, чаще всего это происходит с использованием пароля. Пользователь может добавлять, изменять и удалять связки в этой базе через программу.

Менеджеры паролей использует пароль для шифрования и расшифровки данных. Самой программе пароль неизвестен, она не хранит его в базе данных вместе со связками или где-либо ещё. База данных в зашифрованном виде представляет из себя неинтерпретируемый набор данных, расшифровать который без пароля практически невозможно.

Главным недостатком менеджеров паролей является то, что при получении злоумышленником пароля менеджера паролей (мастер-пароля), он получает доступ ко всем паролям в этой базе данных.

# 1 ОБЗОР ЛИТЕРАТУРЫ

## 1.1 Обзор аналогов

В данном подразделе описаны два существующих менеджера паролей: их основной функционал и особенности.

### 1.1.1 Служба связки ключей Apple

Данный проприетарный продукт является стандартным менеджером паролей на операционных системах (ОС) компании Apple [1]. Как и большинство программного обеспечения (ПО) Apple, является закрытым и доступным только на ОС данной компании. Скриншот приложения приведён на рисунке 1.1.

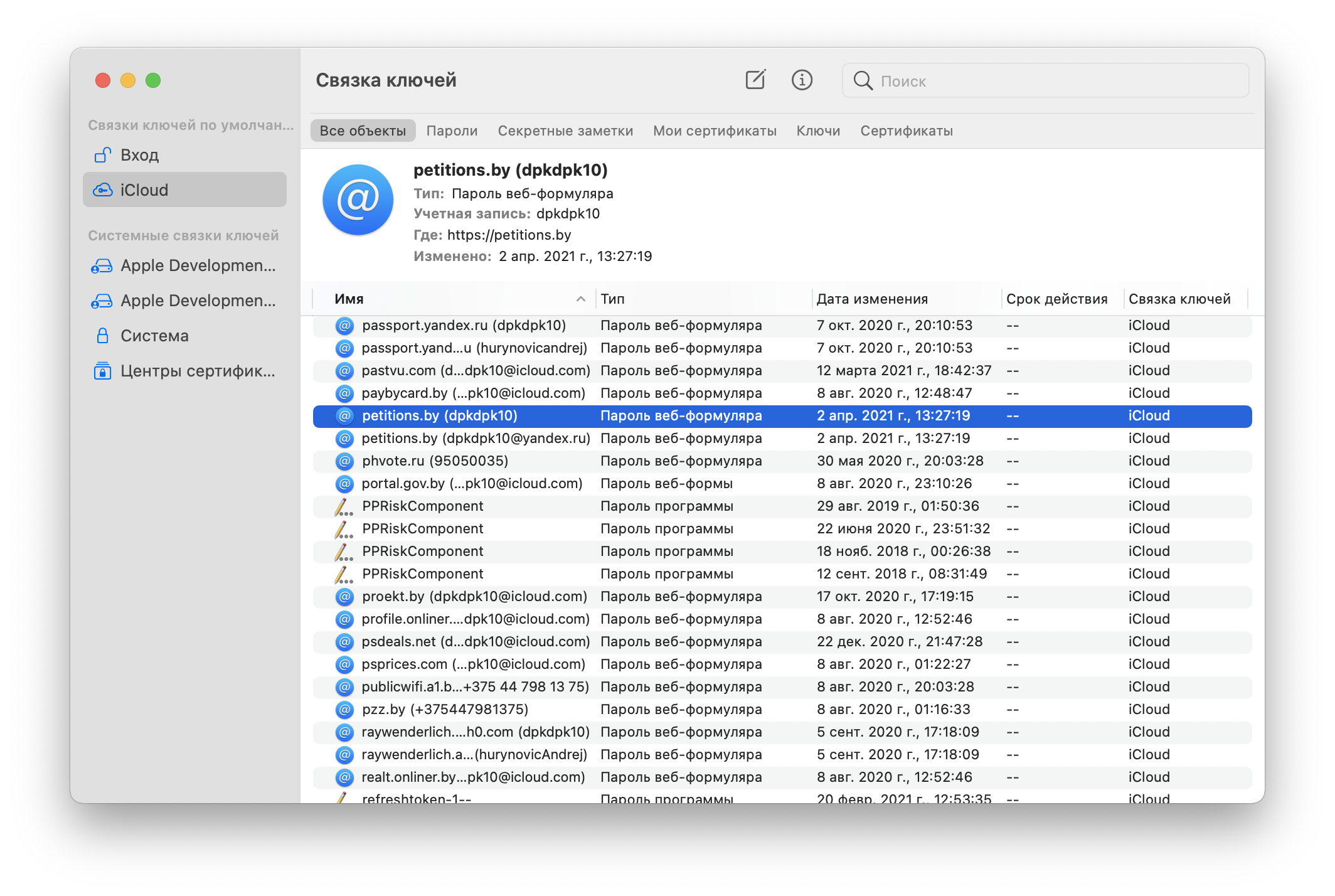


Рисунок 1.1 – Интерфейс службы связки ключей Apple

Хранение данных проходит в зашифрованном виде: используется алгоритм AES (Advanced Encryption Standard) с размером ключа в 256 бит. Система позволяет иметь разные связки (наборы данных) на одном аккаунте, это могут быть, к примеру, личная и корпоративная связка. Данных между устройствами синхронизируются через сеть Интернет. Для доступа к паролям необходима идентифицироваться посредством пароля или сканера отпечатков пальцев TouchID, или сканера лица FaceID. Система интегрирована в Интернет-браузер Safari, что добавляет возможность автоматического заполнения имён пользователей и паролей в соответствующие поля на сайтах в сети.

### 1.1.2 Bitwarden

Данный продукт имеет противоположную предыдущему идеологию: не является проприетарным, имеет открытый исходный код и доступен в качестве приложений на большинстве современных ОС, а также в версии для Интернет-браузеров [2]. Разработан компанией Bitwarden Inc. Скриншот версии приложения для macOS приведён на рисунке 1.2

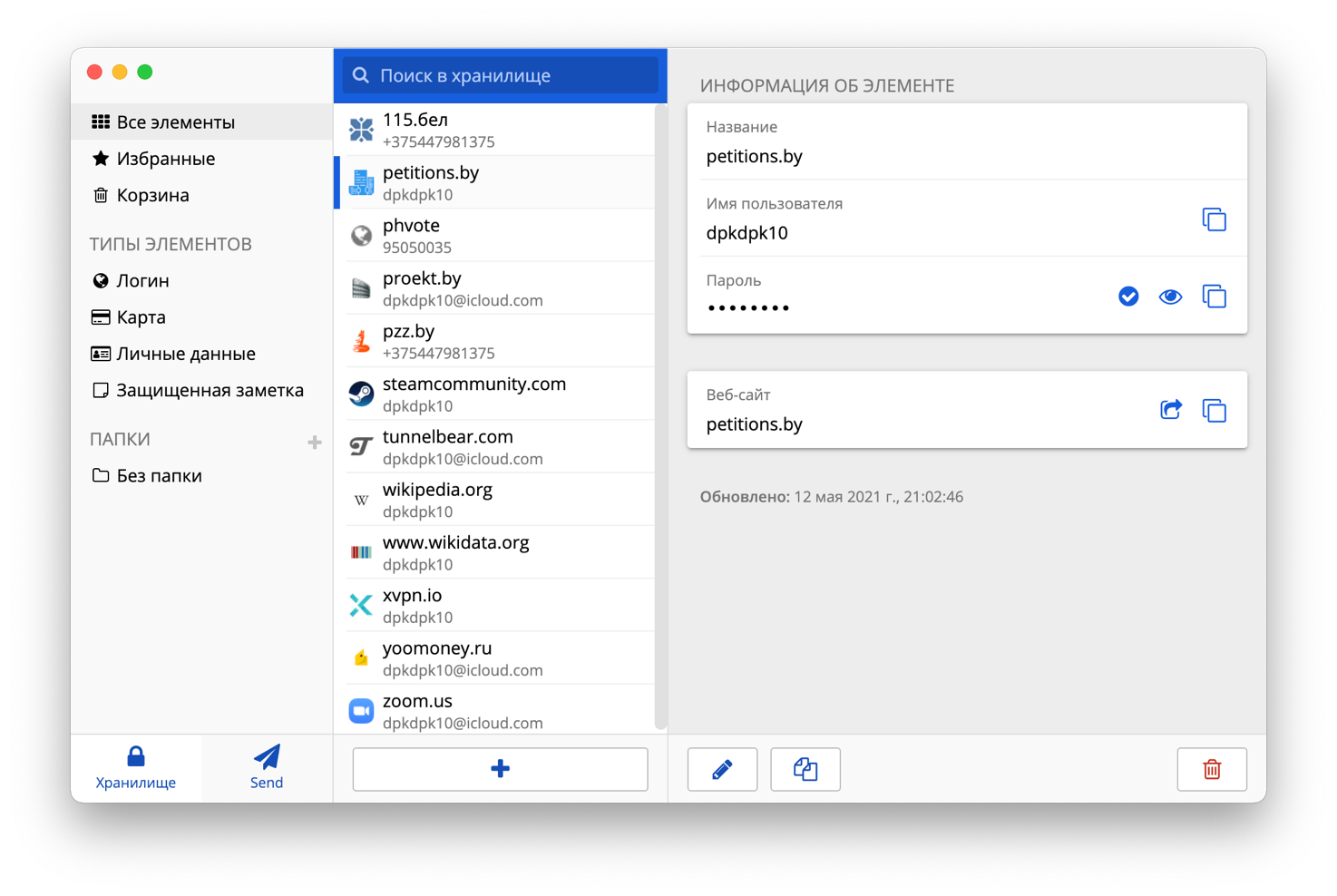
****

Рисунок 1.2 – Интерфейс Bitwarden

Шифрование данных в этом менеджере выполнено, как и в предыдущем аналоге, шифрованием по алгоритму AES с размером ключа в 256 бит. Присутствует синхронизация данных через сеть Интернет, плагины для популярных браузеров, которые обеспечивают возможность быстрого добавления новых данных и автоматического заполнения соответствующих полей. Благодаря открытому исходному коду существует возможность более тщательного изучения алгоритма работы программы, сторонние люди могут найти уязвимость и сообщить о ней разработчикам, как и использовать её в своих целях.

## 1.2 Обзор стандарта шифрования Advanced Encryption Standard

Стандарт разработан в ходе конкурса, объявленного в 1997 году Национальным институтом стандартов и технологий Соединённых Штатов Америки. (США) В 2000 году победителем конкурса был объявлен алгоритм Rijndael.

AES является симметричным алгоритмом блочного шифрования с размером блока в 128 бит и переменным размером ключа: 128 бит, 196 бит, 256 бит – алгоритм регламентирован федеральным стандартом обработки информации США 197 [3]. Современные процессоры на архитектуре x86 поддерживают наборы инструкций для аппаратного шифрования и расшифровки по алгоритму AES.

Симметричность алгоритма означает, что как для шифрования, так и для расшифровки необходимо использовать один и тот же ключ. Блочное шифрование означает, что шифрование применяется для блока с заданным стандартом алгоритма размером. Если шифруемые данные на разделяются на равное количество блоков – необходимо дополнить последнюю часть данных до размера блока.

Суть алгоритма AES достаточно проста для понимания, однако он основан специальных таблицах GF(28) и SBox, метод вычисления которых труден. Таблицы имеют особенности при операциях математики и бинарной логики над ними, эти особенности используют при шифровании и расшифровке. Эти таблицы необязательно вычислять при шифровании каждого блока, так как они всегда одинаковы. Для экономии вычислительных мощностей эти таблицы можно записать как константы. Алгоритм шифрования AES описан в разделе

## 1.3 Обзор Qt

Фреймворк для разработки кроссплатформенного ПО, разработка ведётся на языке программирования C++. Qt позволяет компилировать написанное с его помощью программное обеспечение для большинства современных операционных систем путём компиляции программы для каждой системы без изменения исходного кода.

Qt включает в себя основные классы для создания прикладного ПО: элементы графического пользовательского интерфейса (GUI), классы для работы с сетью, базами данных, переработанные стандартные C++ (String, к примеру, расширен и называется «QString»).

Qt Creator – интегрированная среда разработки. Она имеет встроенный визуальный редактор, который позволяет редактировать GUI в реальном времени, с отображением конечного варианта интерфейса при редактировании. Такой принцип называют: «What You See Is What You Get » (что видишь, то и получишь).   
 Скриншот среды разработки Qt Creator представлен на рисунке 1.3, скриншот среды редактирования GUI Qt Designer представлен на рисунке 1.4.

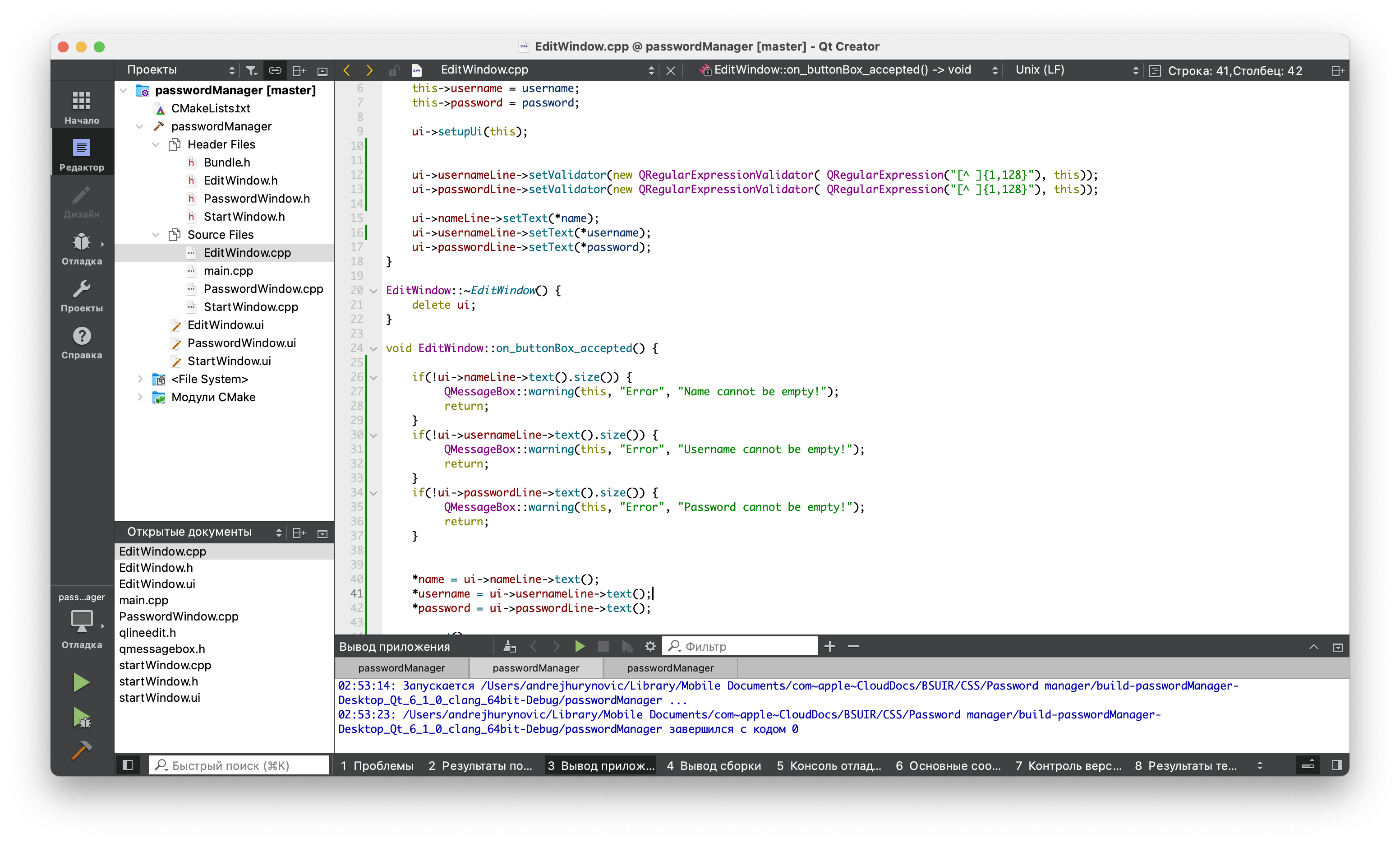


Рисунок 1.3 – Интерфейс Qt Creator

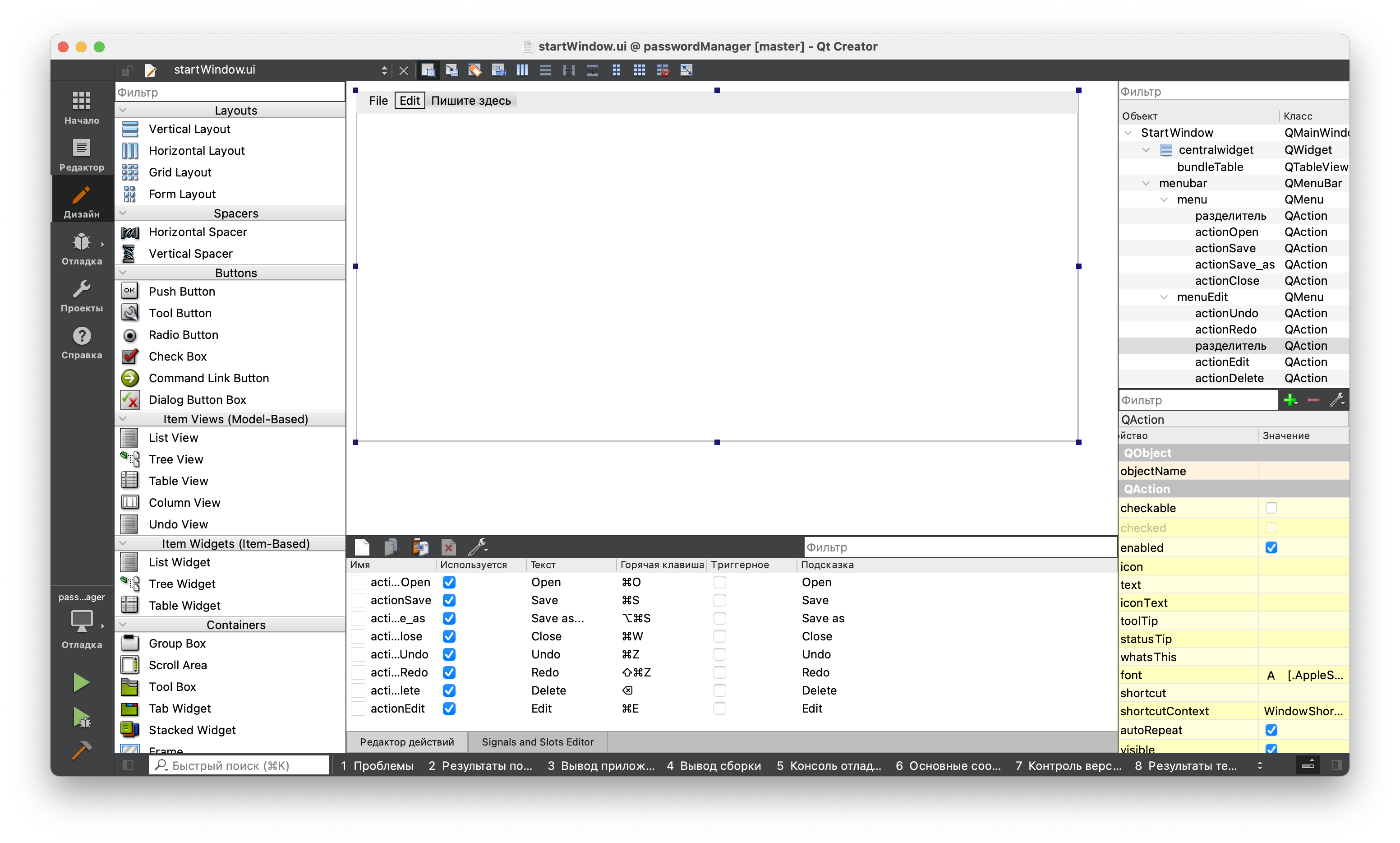


Рисунок 1.4 – Интерфейс Qt Designer

# 2 СИСТЕМНОЕ ПРОЕКТИРОВАНИЕ

## 2.1 Обзор используемых сторонних компонентов

### 2.1.1 Реализация AES

В связи с большим количеством математических алгоритмов, опасности при работе с памятью в C и C++ и отсутствием какой-либо связи с системным программным обеспечением, в частности с Portable Operating System Interface (POSIX), вместо написания собственных функций шифрования и расшифровки по алгоритму AES, было использовано существующее решение.

Реализация AES на C++ разработана SergeyBel и распространяется под лицензией MIT, что позволяет использовать и редактировать исходные коды, при условии указания того, что данный код распространяется под данной лицензией. [4] Текст лицензии приведён в начале заголовочного файла «AES.h», приведённого листинге программы. Листинг находится в приложении Д.

### 2.1.2 Фреймворк Qt для графического пользовательского интерфейса

Фреймворк подробно описан в подразделе 1.3. Для создания программы использовались элементы из QtWidgets. На данном элементе построен весь графический интерфейс. Qt предоставляет возможность высокоуровневого взаимодействия с системой, так QFileDialog позволяет использовать стандартный браузер файлов macOS или Windows с помощью лишь одной функции, при этом имея полноценный функционал, а QMenuBar позволяет использовать системную строку меню создавая в Qt Designer элементарные объекты и функции от них.

## 2.2 Структура программы

Структурная схема приведена в приложении Б. В структуре данной программы существует 5 блоков, каждый из них взаимодействует как минимум с одним другим блоком. Структуризация программы необходима, так как это позволяет упростить восприятие кода и менять конкретные блоки, к примеру, в блоке шифрования существуют две функции: encrypt и decrypt – которые являются «каналами» для взаимодействия с блоком. Таким образом в случае необходимости замены алгоритма шифрования, можно заменить этот блок, оставив только названия этих функций. Ниже приведены все блоки программы и их описание.

### 2.2.1 Блок файловых операций

Представлен классом File; отвечает за создание, открытие, закрытие очистку файлов, запись и чтение информации из них асинхронным методом. Взаимодействует с блоком шифрования: передаёт ему прочитанные данных для расшифровки и последующего вывода данных пользователю; получает от блока шифрования данные для записи в файл.

### 2.2.2 Блок шифрования

Представлен классом AES; отвечает за шифрование и расшифровки 128-битных блоков данных, создание расписания ключей из хеша строки с ключом. Может использовать 128-битный, 196-битный и 256-битный ключ. Взаимодействует с блоком файловых операции (см. пункт 2.2.1) и блоком взаимодействия с пользователем: передаёт ему расшифрованные данные для вывода пользователю; получает от блока данные для шифрования и ключ, после шифрования данные передаются в блок файловых операций, где записываются в файл.

### 2.2.3 Блок настроек

Представлен классом Settings; отвечает за хранение предпочтений пользователя, таких как количество битов в ключе шифрование или возможность автоматического открытия последнего используемого до закрытия программы файла при её повторном открытии. Связан только с блоком взаимодействия с пользователем, так как именно он является управляющим: получает настройки из блока настроек или передаёт их ему.

## 

### 2.2.4 Блок отмены операций

Представлен классом BundleUndo; отвечает за отмену (undo) и восстановление отменённых (redo) операций. Хранит в себе операции, их тип и изменения, которые они несли, по команде из блока взаимодействия с пользователем, который является единственным связанным с ним блоком, выполняет действия по отмене указанной операции.

### 2.2.5 Блок взаимодействия с пользователем

Представлен классами StartWindow, PasswordWindow, EditWindow, SettingsWindow; отвечает за получения и информации от пользователя, его идентификацию и вывод информации пользователю. По сути, совмещает в себе View и Controller из парадигмы Model-View-Controller. Взаимодействует с блоками шифрования, настроек и отмены операций, подробно суть взаимодействия описана в пунктах 2.2.1, 2.2.2, 2.2.3, 2.2.4.

# 3 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ

В данном разделе приведены основные классы и их функции, геттеры, сеттеры и деструкторы опущены. Диаграмма классов представлена в приложении В. Принадлежность классов к блокам указана

## 3.1 Класс AES

unsigned char \*EncryptECB(unsigned char in[], unsigned int inLen, unsigned char key[], unsigned int &outLen) – зашифровать данные.

unsigned char \*DecryptECB(unsigned char in[], unsigned int inLen, unsigned char key[]) – расшифровать данные.

## 3.2 Класс Bundle

Bundle(string name = string(""), string username = string(""), string password = string("")) – конструктор с параметрами по умолчанию.

void encrypt(QString) – зашифровать данную связку.

void decrypt(QString) – расшифровать данную связку.

## 3.3 Класс BundleUndo

enum UndoOperation {create, remove, edit} – enum для вида опреации, произведённой над связкой.

enum UndoOperation type

BundleUndo() – конструктор.

void undo() – отмена действия.

void redo()– восстановление отменённого действия.

## 3.4 Класс EditWindow

explicit EditWindow(QString \*name = NULL, QString \*username = NULL, QString \*password = NULL, QWidget \*parent = nullptr) – явный конструктор с параметрами по умолчанию.

void on\_buttonBox\_accepted() – исполняемая при нажатии кнопки «Okey» функция.

QString \*name – указатель на название связки.

QString \*username – указатель на имя пользователя.

QString \*password – указатель на пароль.

## 3.5 Класс File

QString fileName ­– название файла.

int descriptor ­– дескриптор файла.

aiocb aio ­– структура для асинхронного чтения или записи.

bool newFile() – создание нового файла.

bool openFile() – открытие существующего файла.

bool saveFile() – сохранение файла.

bool saveFileAs() – сохранение файла под другим именем или в другой существующий файл.

void closeFile() – закрыть файл.

bool openFileAndGetDescriptor(bool) – открыть файл и получить его дескриптор.

void read(void\*, long long size, long long offset) – асинхронно прочитать информацию из файла.

void write(void\*, long long size, long long offset) – асинхронно записать информацию в файл.

## 3.6 Класс PasswordWindow

## 

explicit PasswordWindow(QWidget \*parent = nullptr) – явный конструктор с параметрами по умолчанию.

QString password – пароль.

## 3.7 Класс Settings

int keySize – размер ключа шифрования.

bool openOnEnter – открывать ли последний открытый файл при входе в программу.

void loadSettingsFromFile() – загрузить данные из файла.

void saveSettingsToFile() – загрузить данные в файл.

## 3.8 Класс StartWindow

PasswordWindow \*passwordWindow – указатель на окно, в котором вводится пароль.

EditWindow \*editWindow – указатель на окно редактирования связки.

QStandardItemModel \*bundlesModel – указатель на Model, который содержит связки.

File file – файл.

vector<Bundle> bundles – вектор связок.

StartWindow(QWidget \*parent = nullptr) – явный конструктор с параметрами по умолчанию.

void updateTable() – обновить данные в таблице.

void bundleFunctions(bool) – переключить функций, предназначенные для управления связками, в строке меню.

void undoFunctions(bool) – переключить функций, предназначенные для управления отменой действий, в строке меню.

void fileFunctions(bool) – переключить функций, предназначенные для управления файлами, в строке меню.

void on\_actionNew\_triggered() – активируются при нажатии кнопки «New» в строке меню.

void on\_actionOpen\_triggered() – активируются при нажатии кнопки «Open» в строке меню.

void on\_actionSave\_triggered() – активируются при нажатии кнопки «Save» в строке меню.

void on\_actionSave\_as\_triggered() – активируются при нажатии кнопки «Save as…» в строке меню.

void on\_actionClose\_triggered() – активируются при нажатии кнопки «Close» в строке меню.

void on\_actionUndo\_triggered() – активируются при нажатии кнопки «Undo» в строке меню.

void on\_actionRedo\_triggered() – активируются при нажатии кнопки «Redo» в строке меню.

void on\_actionAdd\_triggered() – активируются при нажатии кнопки «Add» в строке меню.

void on\_actionDelete\_triggered() – активируются при нажатии кнопки «Delete» в строке меню.

void on\_actionEdit\_triggered() – активируются при нажатии кнопки «Edit» в строке меню.

void on\_bundleTable\_doubleClicked(const QModelIndex &index) – активируются при двойном нажатии по связке.

# 4 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ

## 4.1 Шифрование алгоритмом AES

Так как подробное описание всего кода с комментариями займёт от 10 страниц, здесь приведено краткое описание алгоритма, ознакомиться с кодом можно в листинге программы, который находится в приложении Д.

Алгоритм преобразует 128-битный блок в другой блок без изменения длины. Для преобразования применяется расписание ключей, получаемое из хеша пароля специальным алгоритмом. 128-битный блок в AES представляется в виде матрицы 4 × 4. Длина ключа равна 4 × Nk байт. Алгоритм шифрования блока состоит из Nr раундов – применений одной и той же группы преобразований к 128-битному блоку данных. Стандартом FIPS 197 допускаются значения Nk иNr, которые указаны в таблице 4.1.

Таблица 4.1 – соотношения длинных ключа в битах, Nk иNr.

|  |  |  |
| --- | --- | --- |
| Длина ключа в битах | Nk | Nr |
| 128 | 4 | 10 |
| 196 | 6 | 12 |
| 256 | 8 | 14 |

Ниже приведены шаги алгоритма для шифрования:

1. Получить пароль от пользователя.

2. Получить хеш от пароля.

3. Преобразовать хеш в ключ по алгоритму, который описан в стандарте AES [3].

4. Разбить полученные данные на блоки по 128 бит.

5. В случае, если последний блок по размеру менее 128 бит, дополнить его нулями до полного размера.

6. Записать блок в виде матрицы размером 4 × 4.

7. Nr раз применить к данной матрице операции описанные в следующих четырёх шагах.

8. addRoundKey(): взять из расписания ключей одну матрицу размером 4 × Nb и поэлементно прибавить её к матрице состояния (результата).

9. subButes(): заменить каждый элемент матрицы состояния соответствующим элементом таблицы SBox.

10. shiftRows(): сдвинуть i-ую строку матрицы состояний на i позиций влево.

11. mixColumns(): умножить i-ую строку матрицы состояний на особую матрицу размера 4 × 4.

## 4.2 Расшифровка алгоритмом AES

1. Получить пароль от пользователя.

2. Получить хеш от пароля.

3. Преобразовать хеш в ключ по алгоритму, описанному в стандарте AES.

4. Разбить полученные данные на блоки по 128 бит.

5. В случае, если последний блок по размеру менее 128 бит, дополнить его нулями до полного размера.

6. Записать блок в виде матрицы размером 4 × 4.

7. Nr раз применить к данной матрице операции описанные в следующих четырёх шагах.

8. invMixColumns(): поделить i-ую строку матрицы состояний на особую матрицу размера 4 × 4.

9. invShiftRows (): сдвинуть i-ую строку матрицы состояний на i позиций вправо.

10. invSubButes(): заменить каждый элемент матрицы состояния соответствующим элементом таблицы InvSBox.

11. addRoundKey(): взять из расписания ключей одну матрицу размером  
4 × Nb и поэлементно прибавить её к матрице состояния (результата).

## 4.3 Асинхронная запись в файл

void write(void\* source, long long size = 16, long long offset = 0) {

aio.aio\_buf = source; //Указатель на место в памяти, из которого будет считываться информация для записи в файл.

aio.aio\_offset = offset; //Отступ в байтах от начала файла от которого необходимо записать данные.

aio.aio\_nbytes = size; //Размер данных для записи в байтах.

aio.aio\_lio\_opcode = LIO\_WRITE; //Код выполняемой операции, в данном случае, код записи.

if (aio\_write(*&aio*) == -1) { //В случае возникновения ошибки при запросе на асинхронную запись.

qDebug() << "Error at aio\_write(): " << strerror(errno); //Вывести в консоль сообщение об ошибке.

}

while(aio\_error(&aio) == EINPROGRESS); //До тех пор, пока система сообщает о том, что запись не была окончена.

aio\_return(*&aio*); //Уведомить систему о том, что информация о завершении данной операции принята и ресурс свободен.

}

## 4.4 Асинхронное чтение из файла

void read(void\* destination, long long size = 16, long long offset = 0) {

aio.aio\_buf = destination; //Указатель на место в памяти, куда будет считываться информация из файла .

aio.aio\_offset = offset; //Отступ в байтах от начала файла от которого необходимо прочитать данные.

aio.aio\_nbytes = size; //Размер данных для чтения в байтах

aio.aio\_lio\_opcode = LIO\_READ; //Код выполняемой операции, в данном случае, код чтения.

if (aio\_read(*&aio*) == -1) { //В случае возникновения ошибки при запросе на асинхронную запись

qDebug() << "Error at aio\_read(): " << strerror(errno); //Вывести в консоль сообщение об ошибке.

}

while(aio\_error(&aio) == EINPROGRESS); //До тех пор, пока система сообщает о том, что чтение не было окончено.

aio\_return(*&aio*); //Уведомить систему о том, что информация о завершении данной операции принята и ресурс свободен.

}

# 5 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

При первом запуске программы необходимо создать новый файл. Для этого необходимо нажать на File/New в строке меню. Появится меню создания файла, скриншот меню приведён на рисунке 5.1. Схема программы приведена в приложении Г.

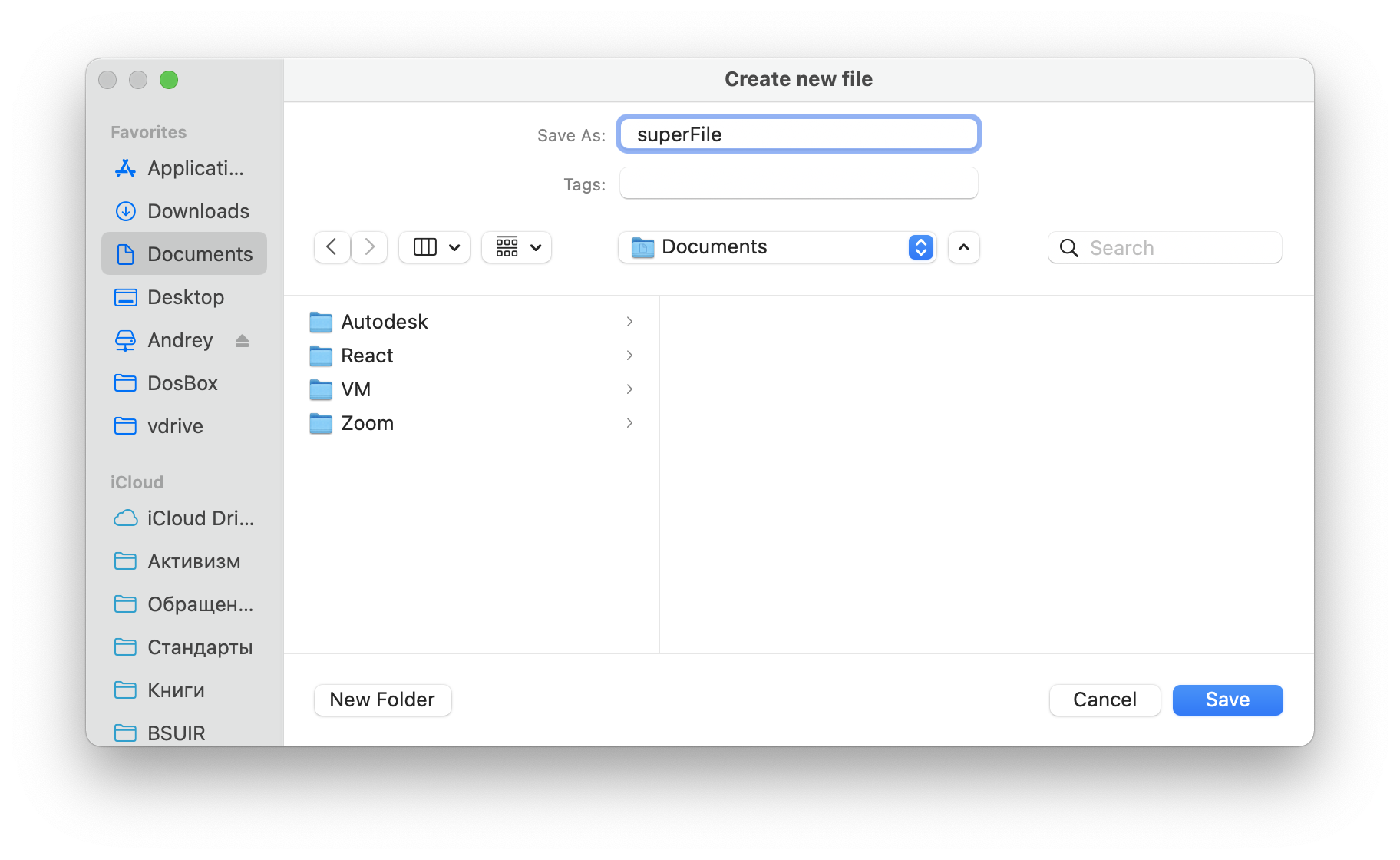


Рисунок 5.1 – Окно создания файла

После создания файла можно создавать, изменять и удалять связки. Для создания нужно выбрать Edit/New. Скриншот окна создания связки приведён на рисунке 5.2.

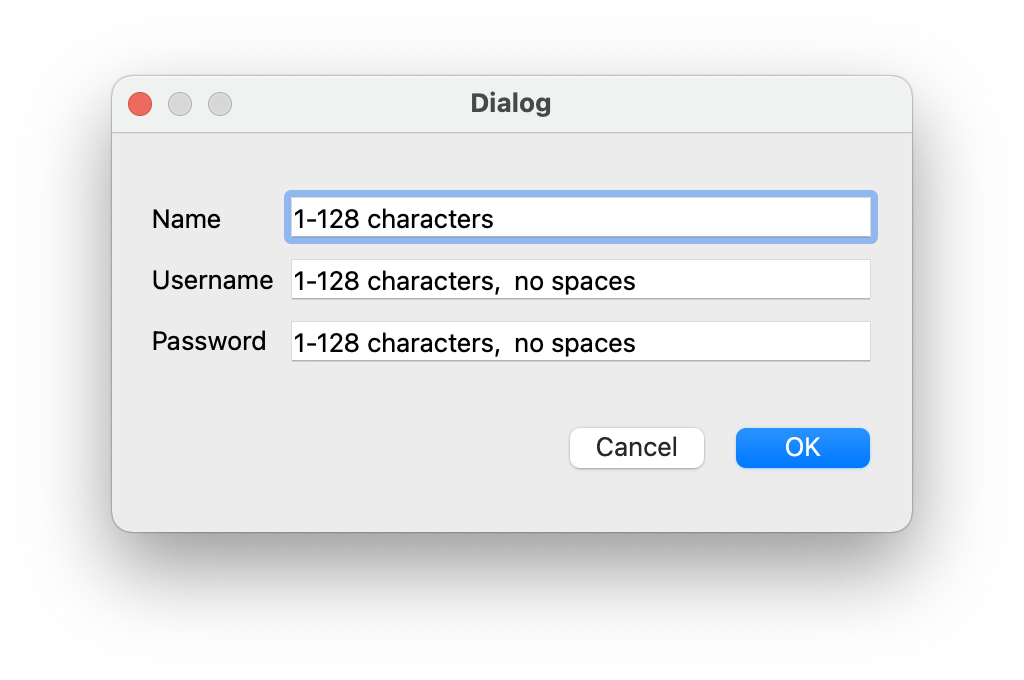


Рисунок 5.2 – Окно создания связки

В полях указаны правила ввода данных. Заполним данными: первое поле – название связки, второе – имя пользователя, третье – пароль. Заполненный вариант окна приведён на рисунке 5.3.

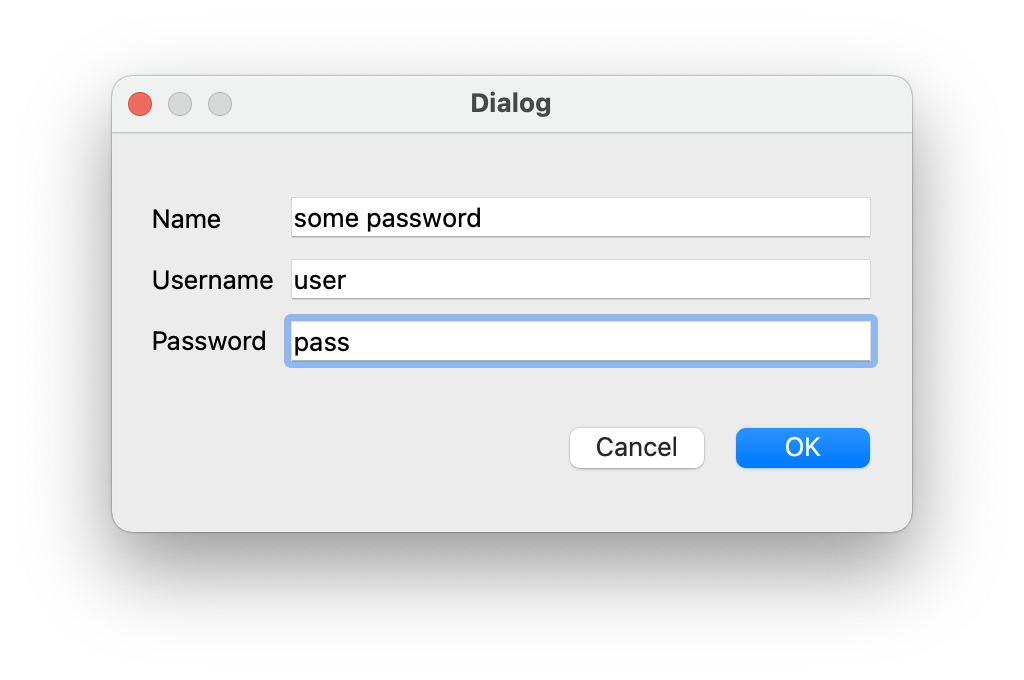


Рисунок 5.3 – Окно создания связки с заполненными полями

Теперь нажмём на кнопку «OK», чтобы подтвердить создание связки. В случае необходимости отмены операции необходимо нажать на кнопку «Cancel».

После подтверждения операции окно закрывается и происходит возвращение главного окна в фокус. Главное окно с созданной связкой приведено на рисунке 5.4.

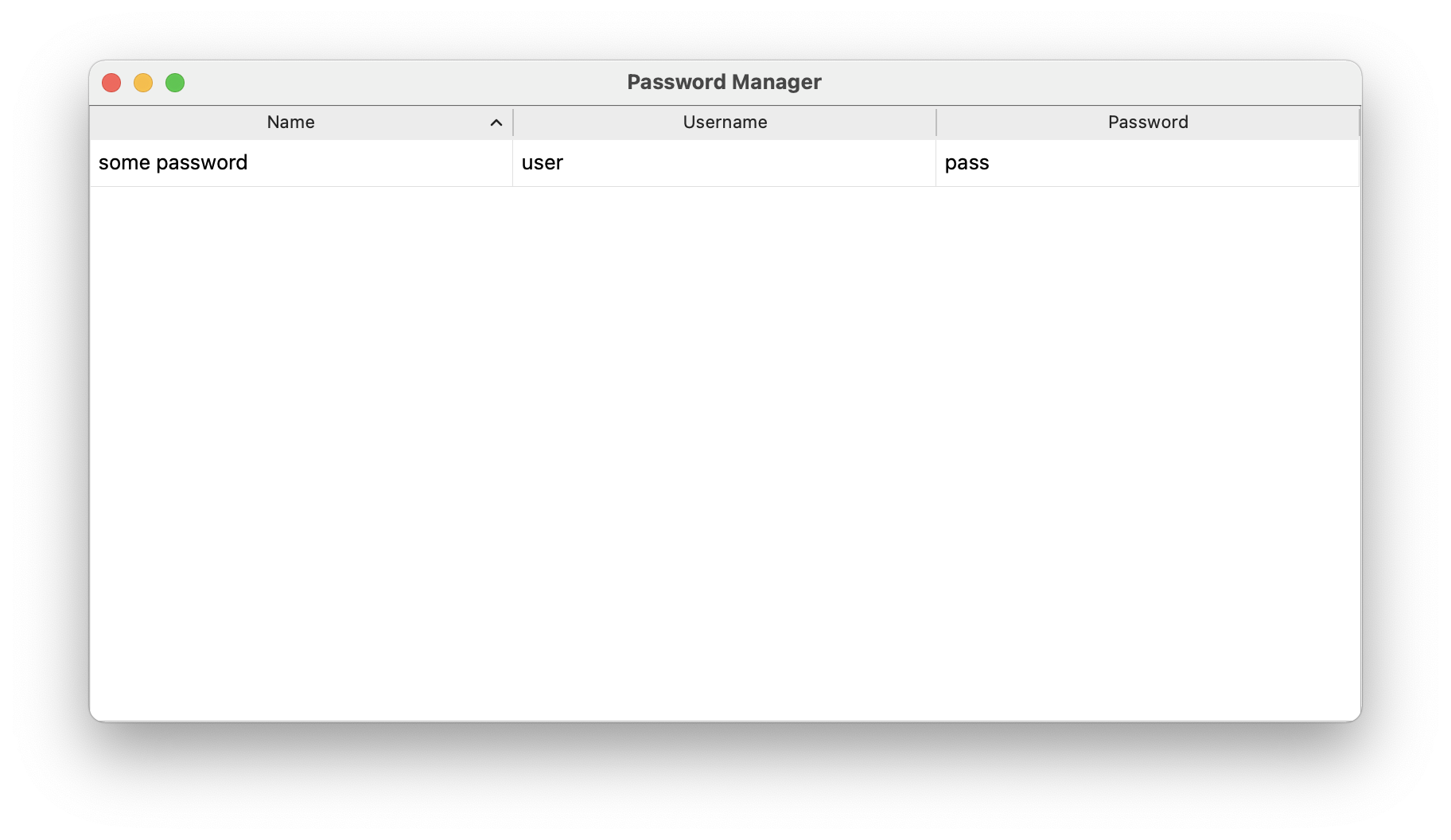
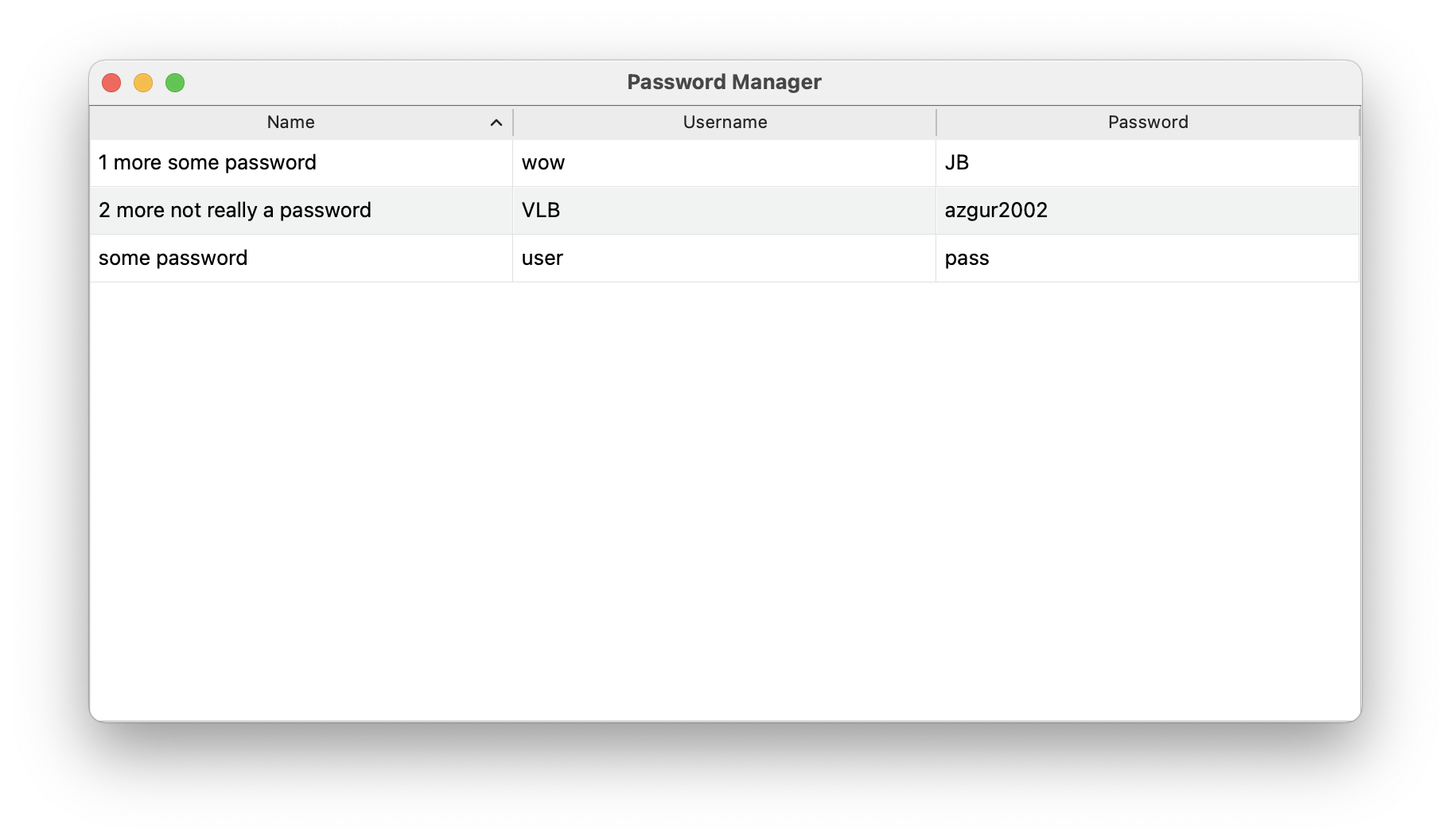


Рисунок 5.4 – Главное окно с созданной связкой

Создадим ещё две связки. По нажатию на один из заголовков произойдёт сортировка по нему. Нажмём на заголовок «Name». Результат сортировки приведён на рисунке 5.5.

  
Рисунок 5.5 – Главное окно с отсортированными по имени связками

Теперь отменим последнее действие: создание новой связки – нажмём в панели меню Edit/Undo, результат операции приведён на рисунке 5.6.

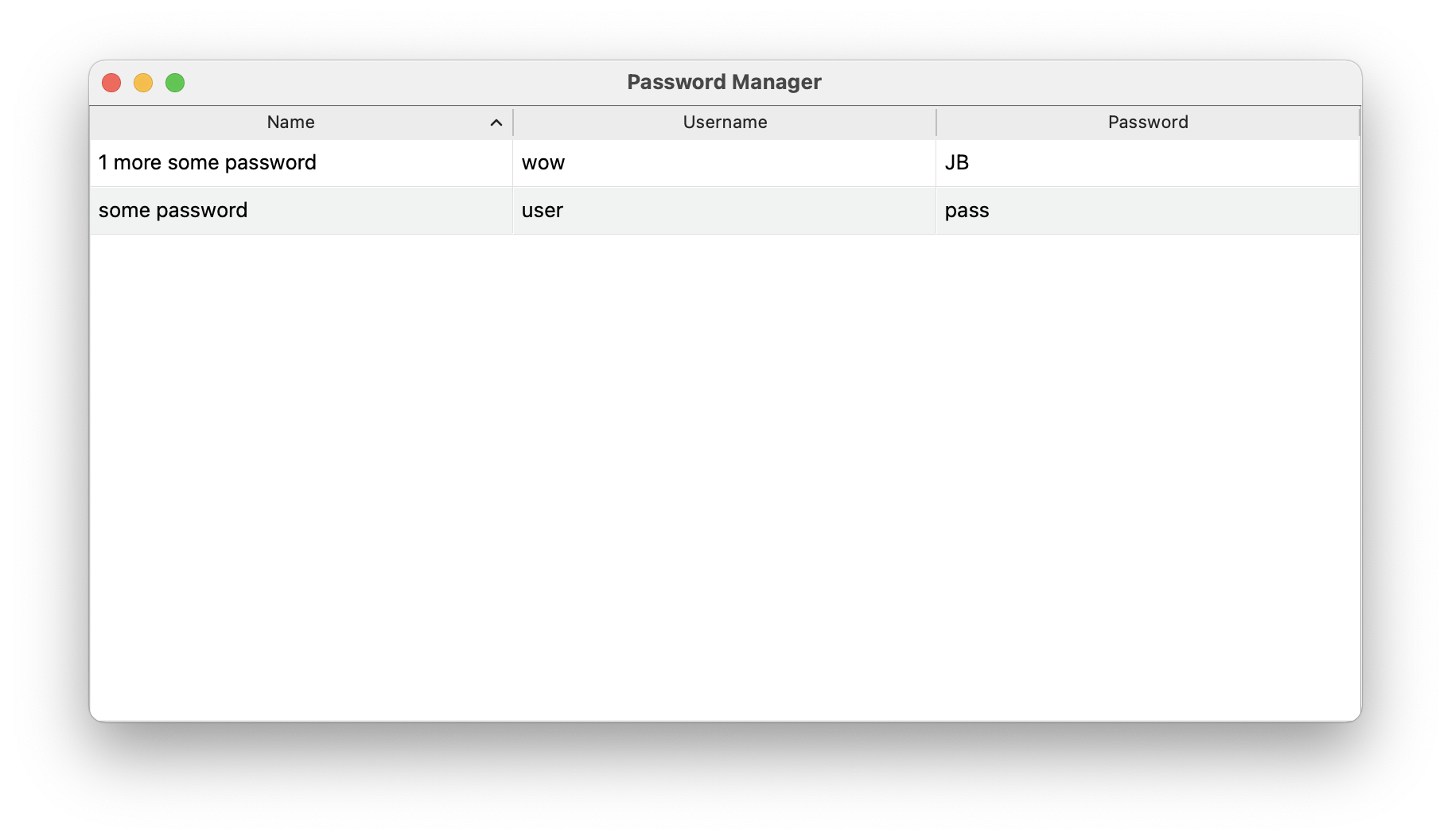


Рисунок 5.6 – Главное окно с отменённым последним действием

Отредактируем связку «some password», для этого дважды кликнем по ней, откроется такое же окно, как и при создании связки, но оно будет заполнено существующей информацией. Скриншоты окна до и после редактирования приведены на рисунках 5.7 и 5.8.

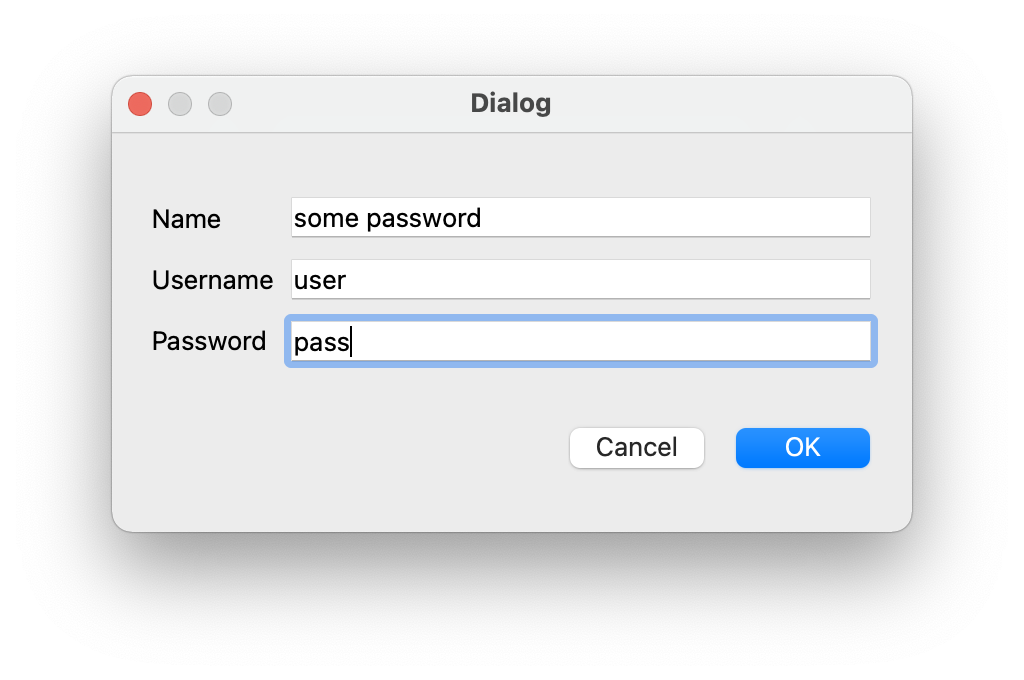


Рисунок 5.7 – Окно редактирования до редактирования

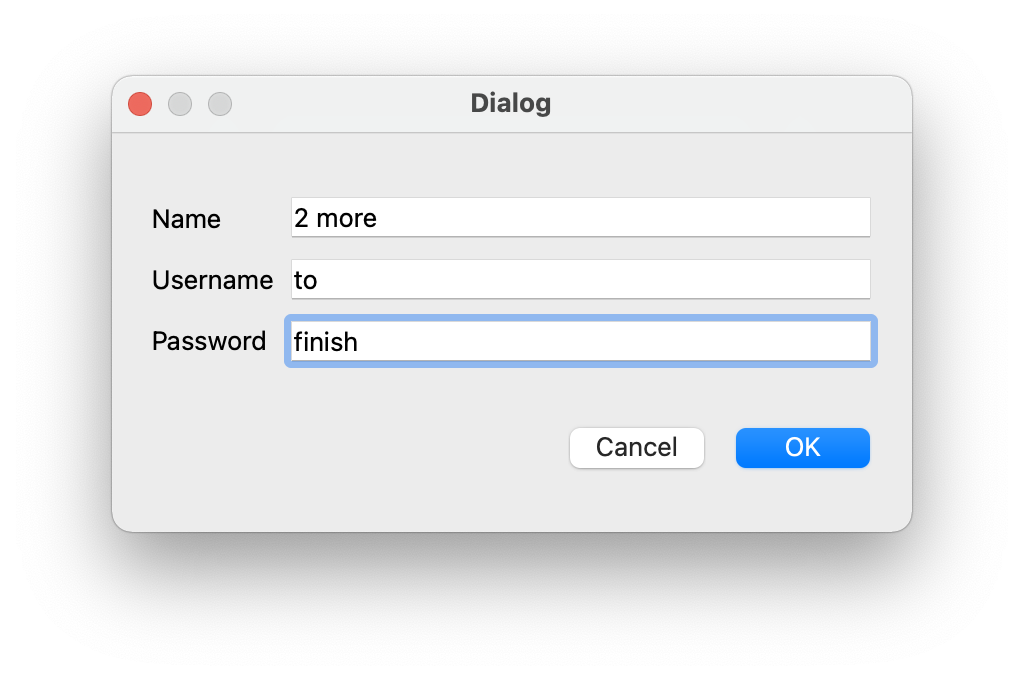


Рисунок 5.8 – Окно редактирования после редактирования

Изменения отображаются и на главном экране, скриншот которого приведён на рисунке 5.9. Видно, что ячейки поменялись местами из-за того, что значение. одной из них было изменено. Это произошло из-за сортировки по названию связки.

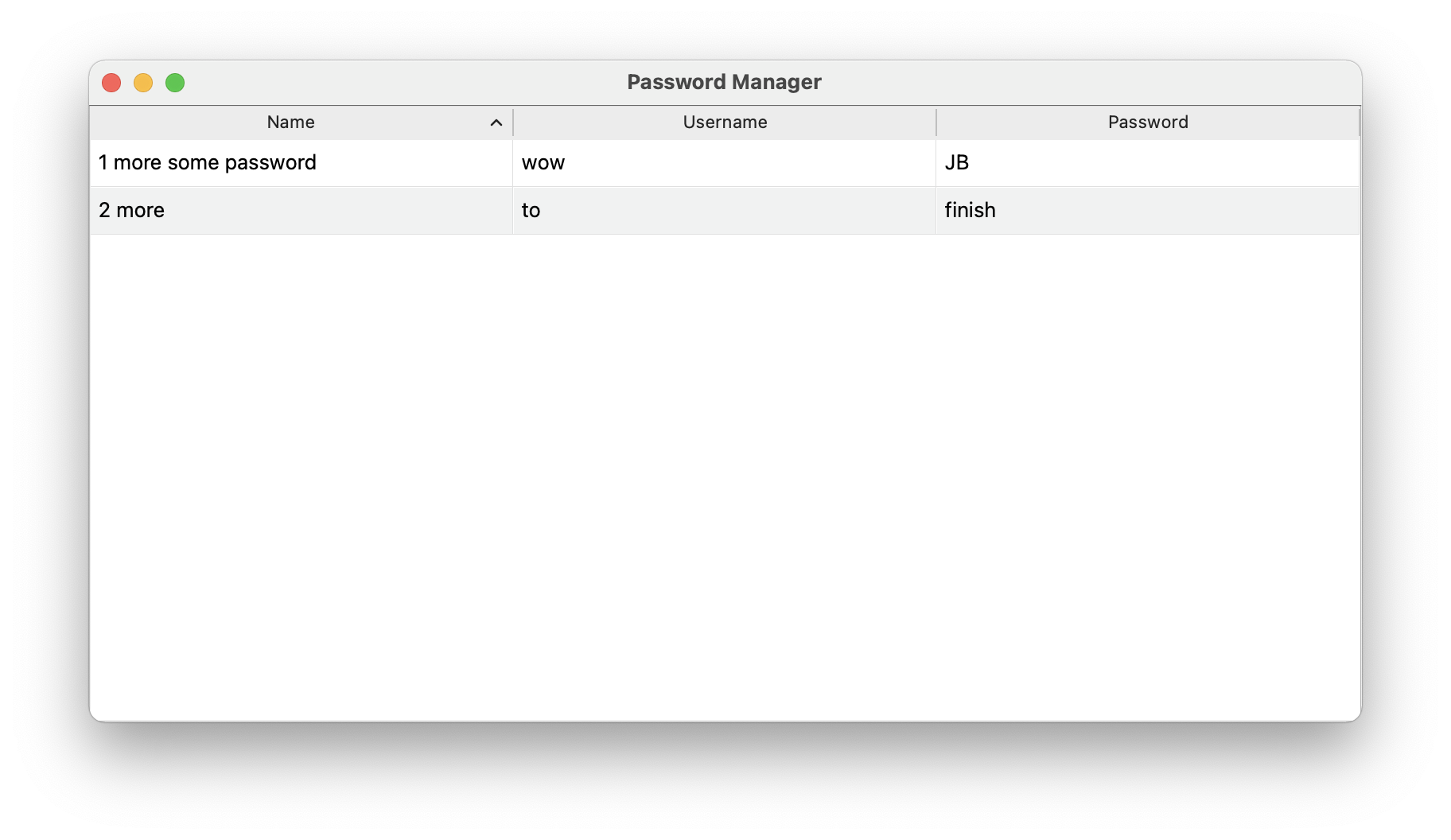


Рисунок 5.9 – Главное окно с отредактированной связкой

Теперь используем для работы сочетания клавиш, они обозначены в панели меню. Скриншот панели меню приведён на рисунке 5.10.

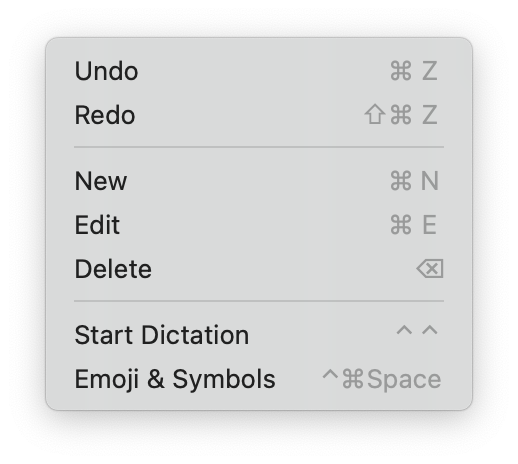


Рисунок 5.10 – Панель меню с подсказками о сочетания клавиш.

Нажмём на клавишу «Delete», при выбранное ячейке с названием «1 more some password». Ячейка удалена, результат приведён на рисунке 5.11.

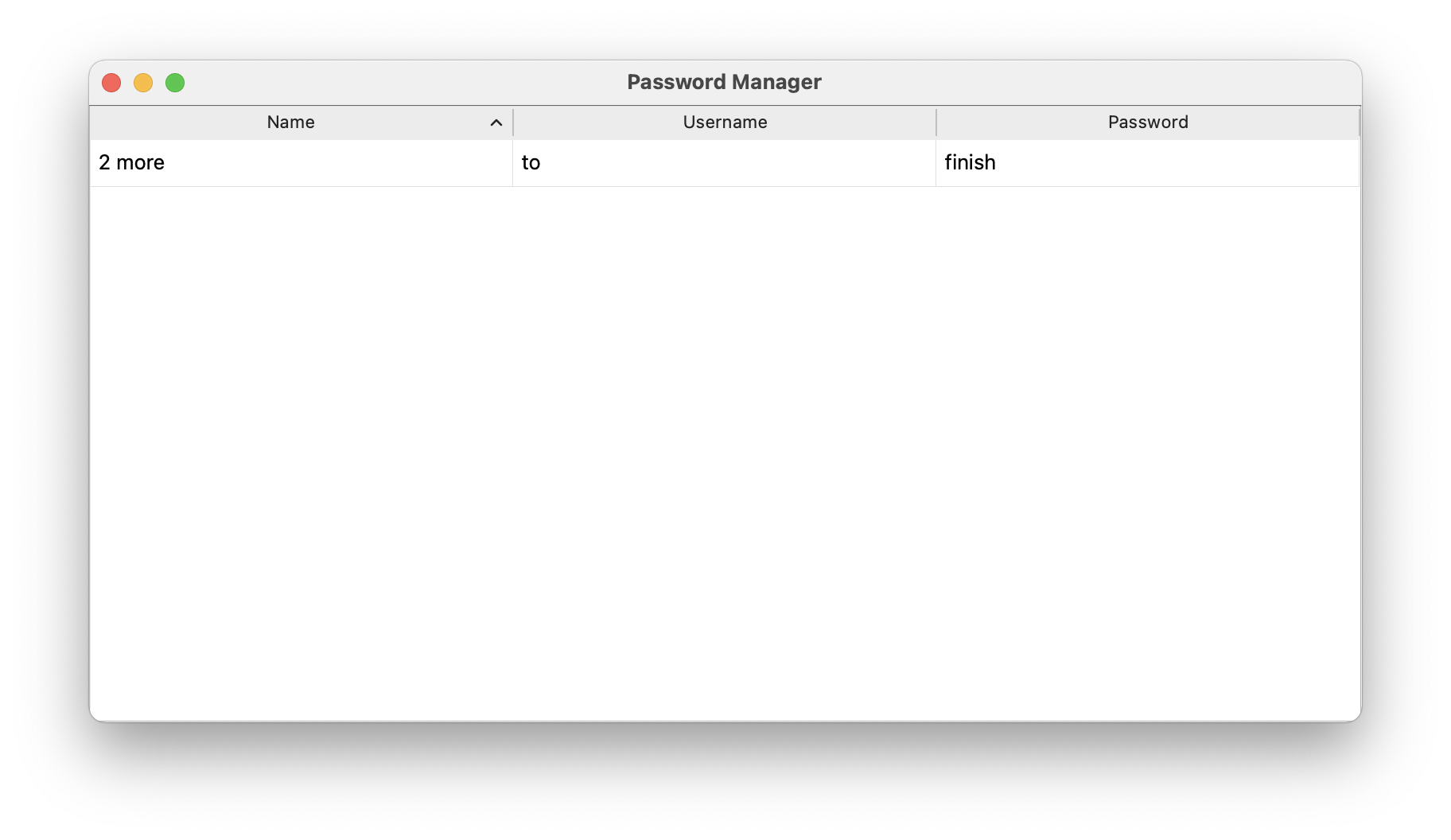


Рисунок 5.11 – Главное окно с удалённой с помощью сочетания клавиш связкой

Для корректного завершения работы программы необходимо сохранить изменения, для этого необходимо перейти в панель меню File/Save, либо использовать соответствующее сочетание клавиш. Этот пункт необязателен, так как перед закрытием программы любым из системных методов, появится меню, в которому можно выбрать сохранять ли изменения данных.

# ЗАКЛЮЧЕНИЕ

В ходе выполнения данного курсового проекта были закреплены знания из курса Системного программного обеспечения вычислительных машин. Дополнительно изучена работа с файлами в POSIX: атрибуты системного вызова «open» и асинхронное чтение и запись.

Изучены основы фреймворка Qt, возможность создания на нём кроссплатформенного прикладного ПО, расширены знания о модели MVC, которые до этого были получены мной при изучении фреймворка Apple Cocoa. Qt предоставляет широкие возможности, однако, из-за кроссплатформенности обладает рядом недостатков: недостаточная интеграция с ОС, необходимость вручную создавать стили и иные элементы, которые работают автоматически в проприетарных фреймворках, таких как Cocoa или Windows Forms. Последние работают на более высоком уровне, но при этом не обладают кроссплатформенностью. При создании приложений, которые предназначены только для macOS или Window лучше использовать соответствующие фреймворки, однако при необходимости создания кроссплатформенного ПО или ПО для X Window System, Qt будет отличным вариантом, которые позволит написать одно приложение сразу для всех платформ без изучения фреймворков каждой.

Изучены основы криптографии: типы алгоритмов шифрований и их свойства, история AES и криптографии в целом. Изучен существующий пример реализации алгоритма AES на С++.

Создано приложение, которое имеет широкий спектр возможностей: хранение, изменение и удаление связок имя пользователя и пароля, с возможностью отмены действий; сортировка; задание настроек; проверка вводимых данных на корректность. Однако очевидна неспособность данного ПО конкурировать с другими менеджерами паролей, из-за локальности данной реализации. Если пользователь захочет воспользоваться своими данными на другом устройстве, ему придётся переслать данные на это устройство вручную, что, очевидно, неконкурентоспособно. Для создания полноценного конкурентоспособного продукта менеджеру паролей необходима синхронизация через сеть Интернет.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Apple Developer [Электронный ресурс]. – Keychain Services. – Режим доступа: https://developer.apple.com/documentation/security/keychain\_services – Дата доступа: 20.05.2021

[2] Bitwarden [Электронный ресурс]. – How Bitwarden works. – Режим доступа: https://bitwarden.com/products/#how-bitwarden-works – Дата доступа: 20.05.2021

[3] National Institute of Standards and Technology [Электронный ресурс]. – Processing Standards Publication 197 – Режим доступа: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf – Дата доступа: 20.05.2021

[4] GitHub [Электронный ресурс]. – C++ AES (Advanced Encryption Standard) implementation – Режим доступа: https://github.com/SergeyBel/AES – Дата доступа: 20.05.2021

# ПРИЛОЖЕНИЕ А

(Обязательное)

**Ведомость документов**

# ПРИЛОЖЕНИЕ Б

(Обязательное)

**Структурная схема**

# ПРИЛОЖЕНИЕ В

(Обязательное)

**Диаграмма классов**

# ПРИЛОЖЕНИЕ Г

(Обязательное)

**Схема программы**

# ПРИЛОЖЕНИЕ Д

(Обязательное)

**Код программы**