2 ОСНОВНАЯ ЧАСТЬ

**2.1 Определение структурной (иерархической) схемы решения задачи**

В рамках данного курсового проекта необходимо спроектировать простейший сервер для моделирования работы серверного хранилища. Необходимо реализовать требуемые возможности, изображенные в виде иерархической схемы ниже.

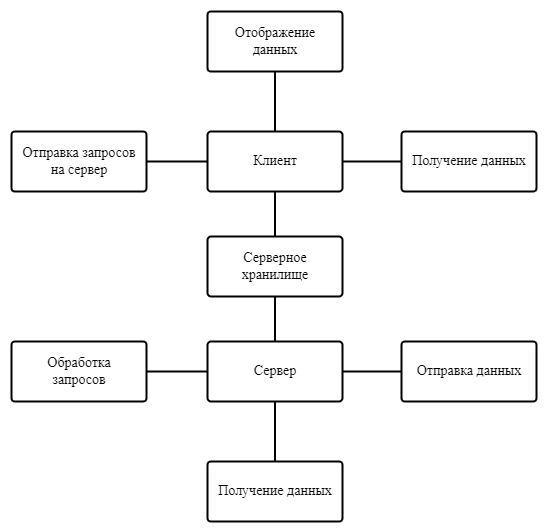


Рисунок 7 — Иерархическая схема решения задачи

**2.2** **Определение основных этапов проектирования**

Среди этапов проектирования приложения выделены следующие:

* анализ требований и составление иерархической схемы решения задачи;
* анализ требуемых методов решения задачи;
* выбор программного обеспечения для реализации программного продукта;
* разработка программного обеспечения;
* разработка интерфейсной составляющей проекта;
* разработка алгоритмов работы программного продукта;
* обработка исключительных ситуаций;
* тестирование программного продукта;
* составление технической документации по проекту.

2.3 Анализ ограничений и исключительных ситуаций для алгоритмов работы с файлами

В процессе разработки программного продукта возникал ряд исключительных ситуаций, требующих обработки. Основные ограничения введены для обеспечения корректности присоединения и отсоединения пользователя к серверу. Необходимо рассмотреть следующие особые ситуации: подключение к серверу, когда соединение закрыто, и передача потерянного файла.

Обеспечение корректности исключительных ситуаций позволяет избежать ошибок при работе основного алгоритма программы.

2.4 Разработка основных алгоритмов задачи

Основным алгоритмом работы программы является общение сервера с клиентом и с помощью протокола TCP. Пользователи вводят IP и имя. Если данные корректны, то происходит подключение к серверу. Сервер слушает запросы клиента, обрабатывает их, после чего производит отправку данных клиенту.

Блок-схема алгоритма подключения к серверу представлена на рисунке 8.

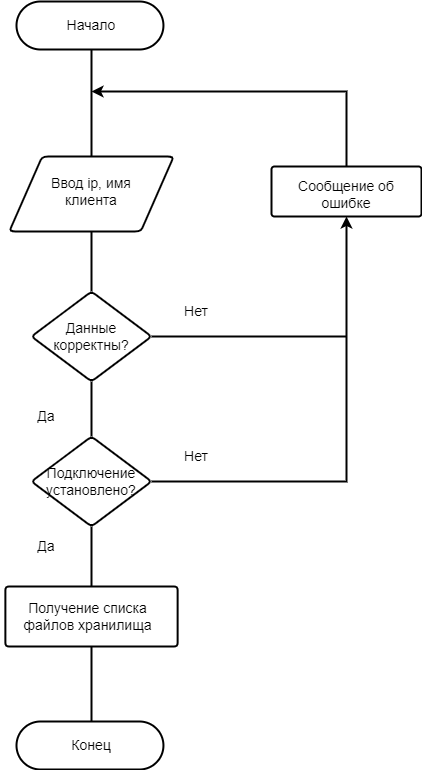


Рисунок 8 — Блок-схема алгоритма присоединения к серверу

Взаимодействия между сервером и клиентом происходит благодаря командам, которые сообщают, какие действия должны быть выполнены, и передаются в начале каждого запроса. Эти команды являются средством общения и позволяют серверу и клиенту понимать друг друга и выполнять необходимые действия. Далее приведен список этих команд:

* ‘c’ – сервер получает информацию о том, что подключился новый клиент и отправляет всем пользователем список имен подключенных клиентов;
* ‘l’ – сервер получает запрос на список файлов хранилища и отправляет его клиенту;
* ‘s’ – сервер получает запрос от клиента на скачивание выбранного файла из хранилища и отправляет этот файл клиенту;
* ‘x’ – сервер получает запрос на удаление выбранного файла из хранилища, удаляет его и отправляет всем пользователям обновленный список файлов;
* ‘f’ – сервер получает информацию о том, что клиент загружает в хранилище новый файл, после сохранения которого, отправляет всем пользователям обновленный список файлов;
* ‘d’ – сервер получает информацию о том, что клиент отключился и отправляет всем пользователям обновленный список клиентов.

2.5 Разработка архитектуры программы

Проект разбит на два модуля: северная часть и клиентская.

Серверная часть комплекса разбита на 3 класса:

* Program.cs
* ServerCloud.cs
* ConnectedClient.cs

Каждый класс отвечает за свой перечень задач, взаимодействуя с методами из других классов для получения необходимых данных.

В таблице 4 описаны основные классы программы.

Таблица 4 — Классы и методы приложения

|  |  |
| --- | --- |
| **Название функции** | **Описание** |
| class Program | Класс для запуска сервера |
| class ServerCloud | Класс содержит поля и методы для хранения и обработки запросов и ответов от клиента и сервера |
| class ConnectedClient | Класс содержит поля для хранения информации о подключенном клиенте |

Продолжение таблицы 4

|  |  |
| --- | --- |
| **Название функции** | **Описание** |
| void ReadClient() | Метод для прослушивания клиента |
| void SendList(TcpClient tcpClient) | Метод для отправки клиенту списка файлов |
| Task<int> StartServer() | Метод для запуска сервера |

Клиентская часть комплекса разбита на 2 класса:

* Program.cs;
* Form1.cs.

В таблице 5 описаны основные классы программы.

Таблица 5 — Классы и методы приложения

|  |  |
| --- | --- |
| **Название функции** | **Описание** |
| class Program | Класс для запуска программы |
| class Form1 | Класс для основного окна приложения |
| void ClientStart() | Метод для подключения к серверу |
| void Disconect() | Метод для отключения |
| void Mes() | Метод для прослушивания сервера |

**2.6 Разработка дисплейных фрагментов**

Интерфейс клиентской части программного комплекса состоит из панели авторизации, панели пользователей (в ней отображается список подключенных клиентов), панели файлов и кнопок управления (рисунок 9).

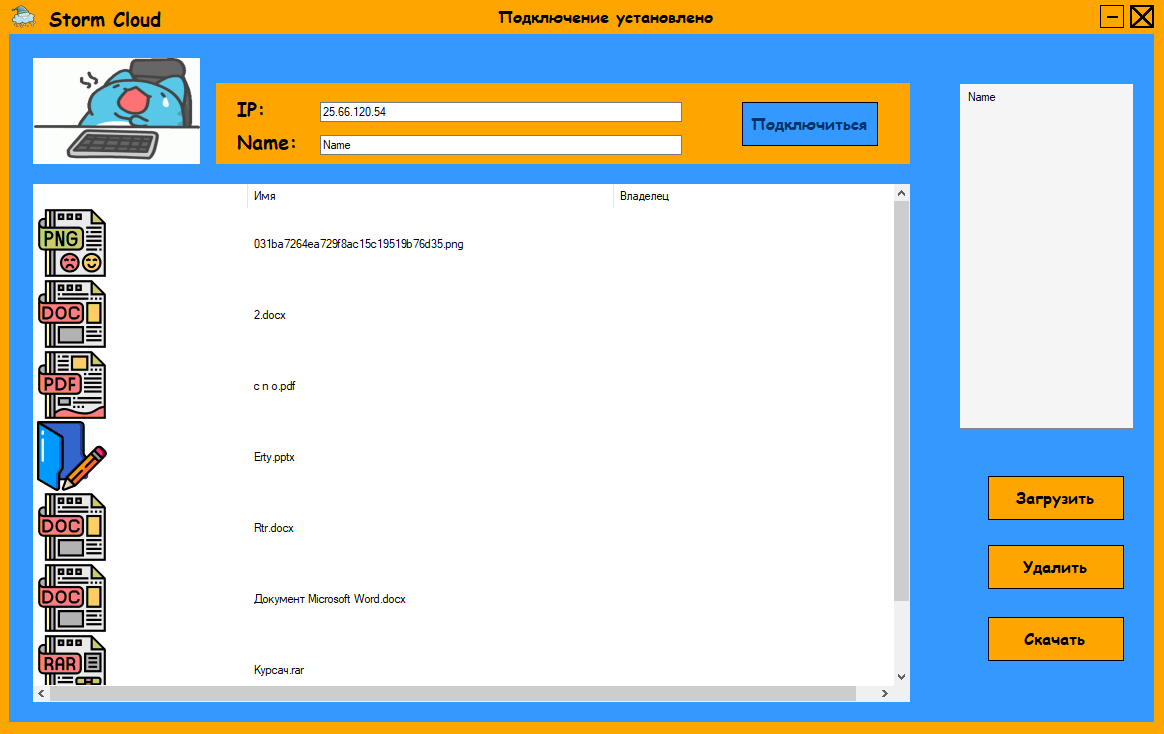


Рисунок 9 – Главное окно программы

Интерфейс серверной части представляет собой консоль. На экран выводится информация о подключении клиента (рисунок 10).

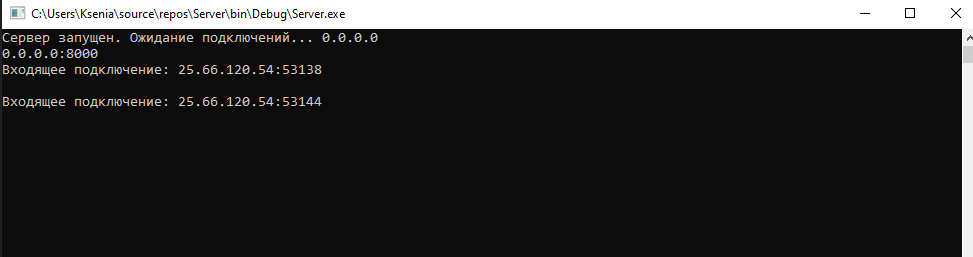


Рисунок 10 – Серверная часть программы

**2.7 Написание и отладка модулей программы**

Отладка программного комплекса производилась путем проверки поведения различных частей программного кода при изменяющихся входных параметрах. В процессе отладки были выявлены и исправлены следующие ошибки:

* закрытое соединение при подключении пользователя;
* загрузка или скачивание поврежденных файлов;
* некорректное отключение от сервера;
* некорректное обновление содержимого хранилища.

**2.8 Тестирование разработанного программного продукта**

Для тестирования программного комплекса попробуем подключиться к серверу без ввода имени пользователя. Результат представлен на рисунке 11.

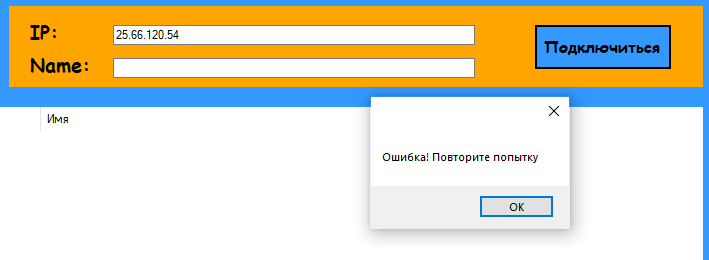


Рисунок 11 – Результат работы программы

Также попытаемся подключиться к закрытому серверу. Результат представлен на рисунке 12.



Рисунок 12 – Результат работы программы

Далее зайдем в программу и попробуем загрузить файл. Результат представлен на рисунках 13 -14.

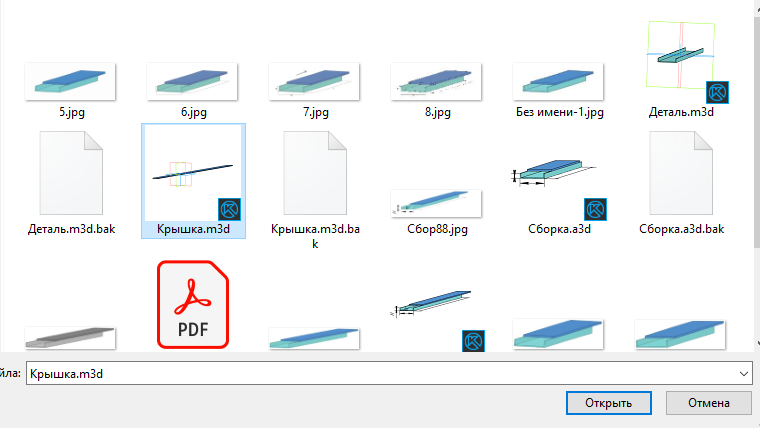


Рисунок 13 – Выбор файла для загрузки

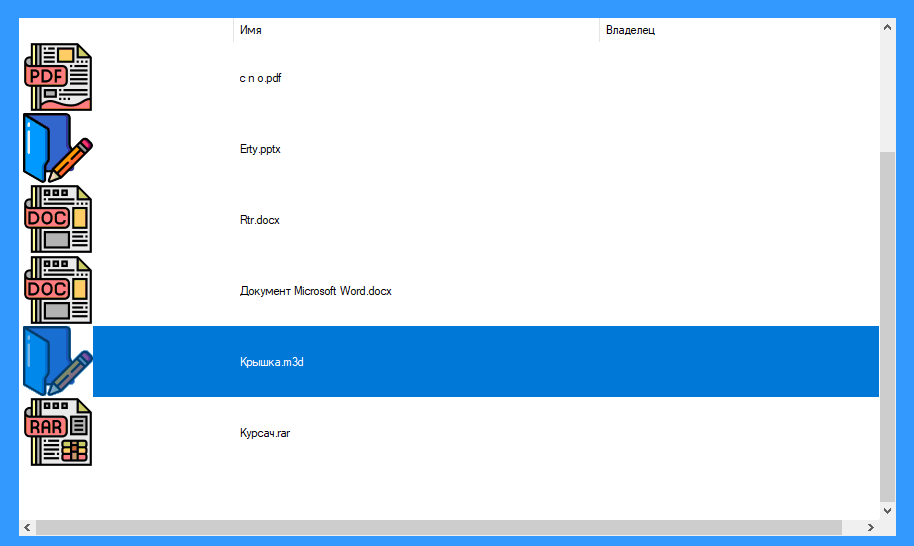


Рисунок 14 – Отображение загруженного файла в списке файлов хранилища

Далее попробуем скачать выбранный файл. Результат представлен на рисунках 15-16.

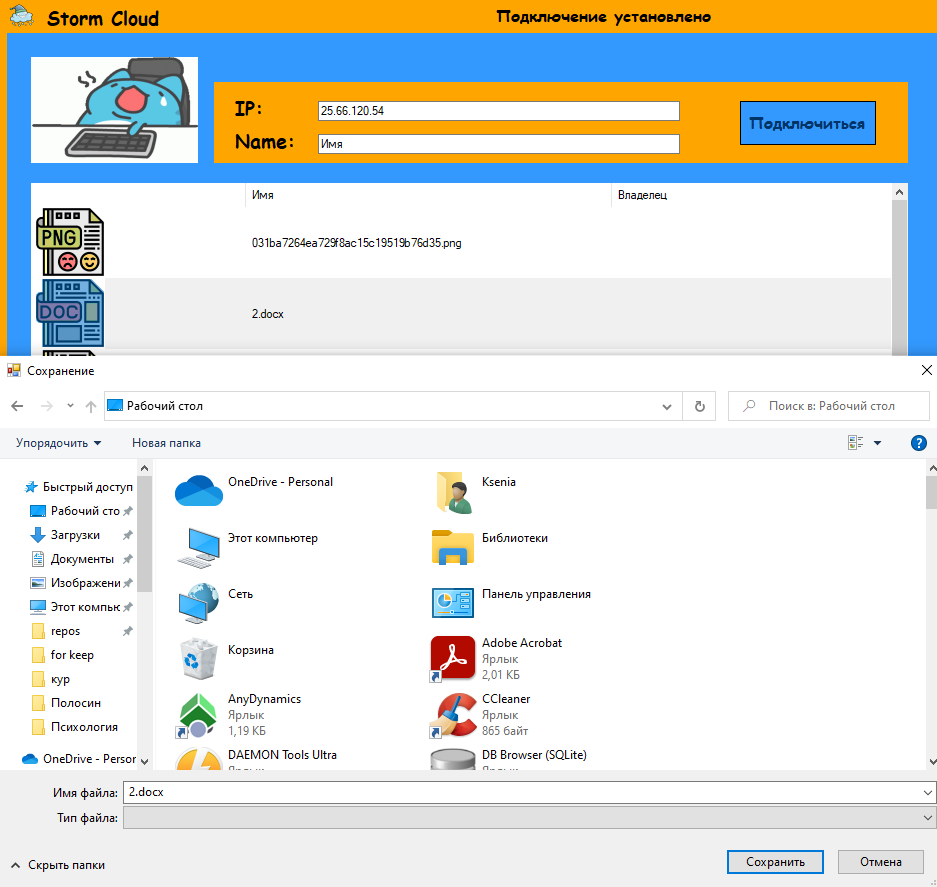


Рисунок 15 – Выбор пути для скачивания файла



Рисунок 16 – Результат работы программы

**ВЫВОДЫ**

В процессе выполнения данной курсовой работы был создан программный комплекс, реализующий серверное хранилище под ОС Windows\*, получены навыки создания и подключения библиотек классов, создания инсталляционного пакета. В ходе разработки были выполнены следующие задачи:

• обоснование выбора инструментальных средств разработки;

• разработка структурной схемы решения задачи;

• разработка структуры данных, определение диапазона изменений;

• анализ ограничений и исключительных ситуаций;

• разработка основных алгоритмов программы;

• разработка архитектуры и дисплейных фрагментов приложения;

• создание и отладка модулей программного комплекса;

• создание справочной системы программного продукта;

• создание инсталляционного пакета программы;

• тестирование разработанного программного продукта.

Разработанное программное обеспечение позволяет пользователю экономить место на своем устройстве, иметь доступ к файлам с разных устройств. В дальнейшей разработке программного комплекса предполагается исправить ошибки, которые возникают при многопользовательской работе. Также планируется доработать список владельцев файлов.