Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовому проекту
на тему

Многопоточный FTP-сервер с возможностью запроса каталогов и передачи их в архивированном виде.

по дисциплине

«Системное программное обеспечение вычислительных машин»

КП БГУИР 1-40 02 01.310 ПЗ

Выполнил:
Студент гр. 050503
Казак И. А.

Руководитель:
Глоба А. А.

Минск 2022

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

ЗАДАНИЕ
по курсовой работе студента
Казак Ивана Александровича

1. **1** Тема работы: «Многопоточный FTP-сервер с возможностью запроса каталогов и передачи их в архивированном виде»
2. Срок сдачи студентом законченной работы: 28.05.2022
3. Исходные данные к работе:
   3.1. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. В режиме реального времени контролировать подключение и отключение клиентов. По запросу предоставлять подробную информацию о файлах на сервере. Программа должна иметь возможность вернуть каталог в архивном виде.
4. Содержание пояснительной записки (перечень подлежащих разработке вопросов):
   4.1. Введение. 1. Обзор методов и алгоритмов решения поставленной задачи. 2. Обоснование выбранных методов и алгоритмов. 3. Описание программы для программиста. 4. Описание алгоритмов решения задачи. 5. Руководство пользователя. Заключение. Список литературы.
5. Перечень графического материала:
6. Диаграмма - классов.
7. Блок - схема.

КАЛЕНДАРНЫЙ ПЛАН

Наименование этапов курсовой работы	Объём этапа, %	Срок выполнения этапа	Примечания
Разработка структурной схемы программы	10	07.03–17.03	С выполнением чертежей
Разработка диаграммы классов	15	18.03–08.04	С выполнением чертежей
Разработка основного функционала приложения с выполнением блок-схем	50	09.04–10.05	С выполнением чертежа
Разработка интерфейса	15	11.05–25.05	
Завершение оформления пояснительной записки	10	26.05–10.06	

Дата выдачи задания: 12.02.2022 г.

РУКОВОДИТЕЛЬ _____ *А.А. Глоба*
(подпись)

Задание принял к исполнению _____ *И.А. Казак*
(дата и подпись студента)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Курсовое проектирование является обязательным элементом подготовки специалиста с высшим образованием и является одной из форм текущей аттестации студента по учебной дисциплине.

Данная курсовая работа посвящена разработке многопоточного FTP-сервера с возможностью запроса каталогов и передачи их в архивированном виде.

Для реализации поставленной задачи был выбран язык программирования C++, а также, для разработки удобного пользовательского интерфейса, Фреймворк Qt. Как один из базовых и самых распространенных языков программирования C++ включает в себя процедурное программирование, позволяет создать интерактивный интерфейс, что значительно расширяет круг его применения и упрощает написание проектов высокого уровня сложности. Кроме того, этот язык наиболее приближен к системе и системным функциям среди высокоуровневых языков, таким образом являясь наиболее подходящим для решения поставленной задачи.

Данная разработка может быть использована в качестве удобного хранилища файлов и управления доступом к этим файлам. У данного проекта крайне широкая целевая аудитория так как он позволяет не имея серьезных навыков делиться и хранить свои фалы.

1. ОБЗОР ЛИТЕРАТУРЫ

1.1. Анализ существующих аналогов

Тема курсового проекта была выбрана в первую очередь для углубления знаний по языку C++ и в области объектно-ориентированного программирования, а также получения знаний в проектировании пользовательских приложений, поэтому моей целью не является разработать конкурентоспособный продукт. Тем не менее, чтобы создать корректно работающее приложение, нужно иметь представление о существующих аналогах, об их недостатках, преимуществах и реализованных внутри функциях.

FTP расшифровывается как File Transfer Protocol — протокол передачи файлов. Он отличается от других протоколов тем, что если в процессе передачи возникает какая-то ошибка, то процесс останавливается и выводится сообщение для пользователя. Если ошибок не было, значит, пользователь получил именно тот файл, который нужен, в целости и без недостающих элементов.

Для работы по FTP нужны двое: FTP-сервер и FTP-клиент. Что делает сервер:

- обеспечивает доступ по логину и паролю к нужным файлам;
- показывает пользователю только те файлы и папки, которые он может просматривать или загружать в них;
- следит за качеством передачи и смотрит, чтобы не было ошибок;
- управляет параметрами соединения в пассивном режиме.

FTP-клиент – это файловый менеджер, который осуществляет подключение к удаленному серверу для передачи данных. Для подключения клиента к удаленному серверу нужны следующие данные:

- логин,
- пароль,
- хост (имя сервера),
- номер порта (по умолчанию 21 для FTP-соединения).

1.1.1. FileZilla Server

FileZilla Server — это серверное приложение с открытым исходным кодом для Windows. Он может администрировать как локальный сервер, так и удаленный FTP-сервер. Поддерживает обмен данными по протоколу FTP. Интерфейс FileZilla Server изображен на рисунке 1.1:

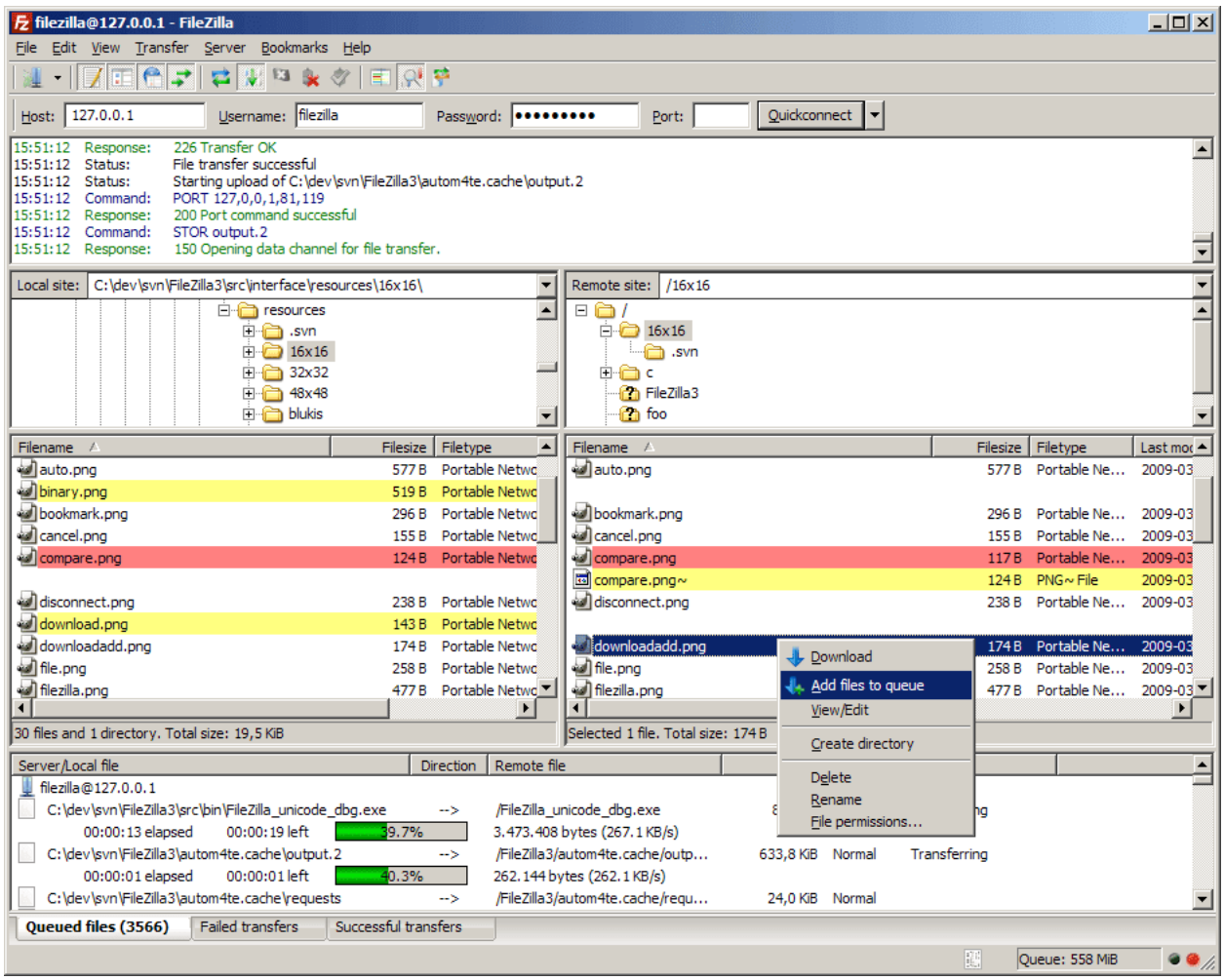

Рисунок 1.1 — Скриншот программы FileZilla Server

Достоинства FileZilla Server:

- Возможность передачи файлов одновременно.
- Поддерживает безопасную передачу файлов.
- Закладки для быстрого подключения.

Недостатки FileZilla Server:

- Не удается редактировать файлы из приложения.
- Отсутствие автоматического обновления видов папок.

1.1.2. FTP-сервер Xlight

Xlight — это бесплатный FTP-сервер, который выглядит намного современнее, чем FileZilla, а также содержит множество настроек.

FTP-сервер Xlight может использовать SSL и может требовать от клиентов использования сертификата. Он также поддерживает ODBC, Active Directory и аутентификацию LDAP. Интерфейс FTP-сервер Xlight изображен на рисунке 1.2:

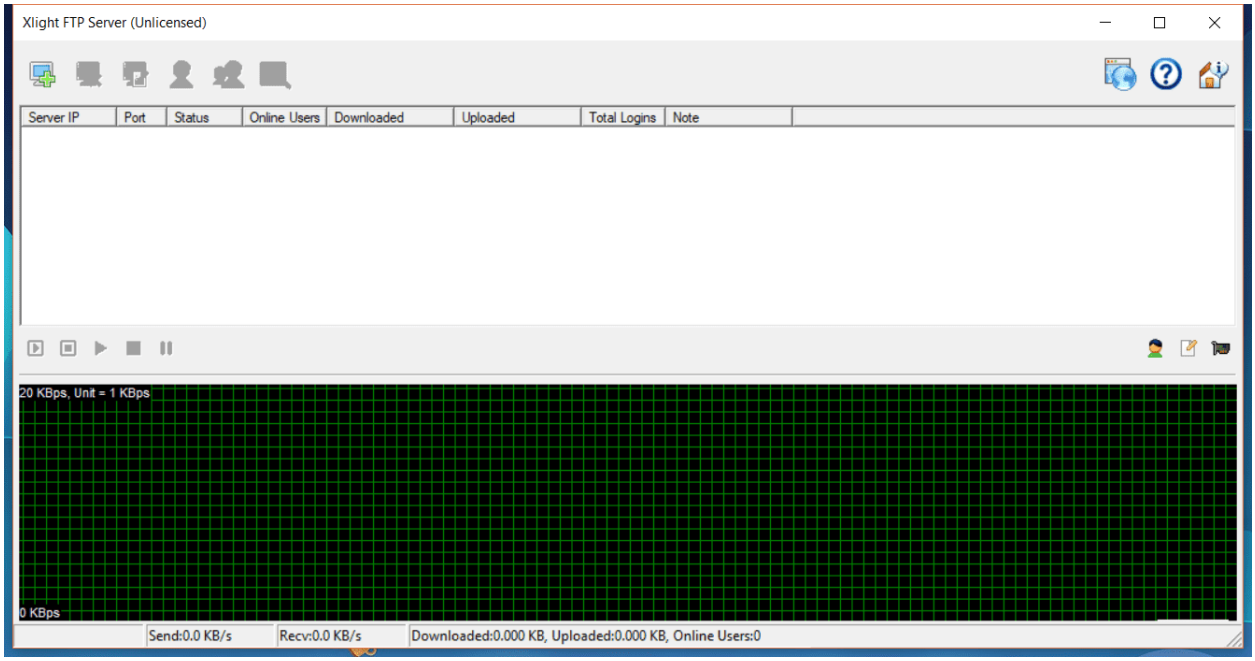

Рисунок 1.2 — Скриншот программы FTP-сервер Xlight

Достоинства FTP-сервер Xlight:

- Поддерживает безопасную передачу файлов.
- Функция удаленного администрирования.
- Поддерживает несколько подключений одновременно.

Недостатки FTP-сервер Xlight:

- Сложнее использовать для новичков FTP.
- Может быть сложным в настройке.

1.1.3. Complete FTP

Complete FTP — еще один бесплатный Windows FTP-сервер, который поддерживает FTP и FTPS. Эта программа имеет полный графический интерфейс пользователя и очень проста в использовании. Сам интерфейс довольно прост, но все настройки скрыты в боковом меню и просты для доступа. Интерфейс Complete FTP изображен на рисунке 1.3:

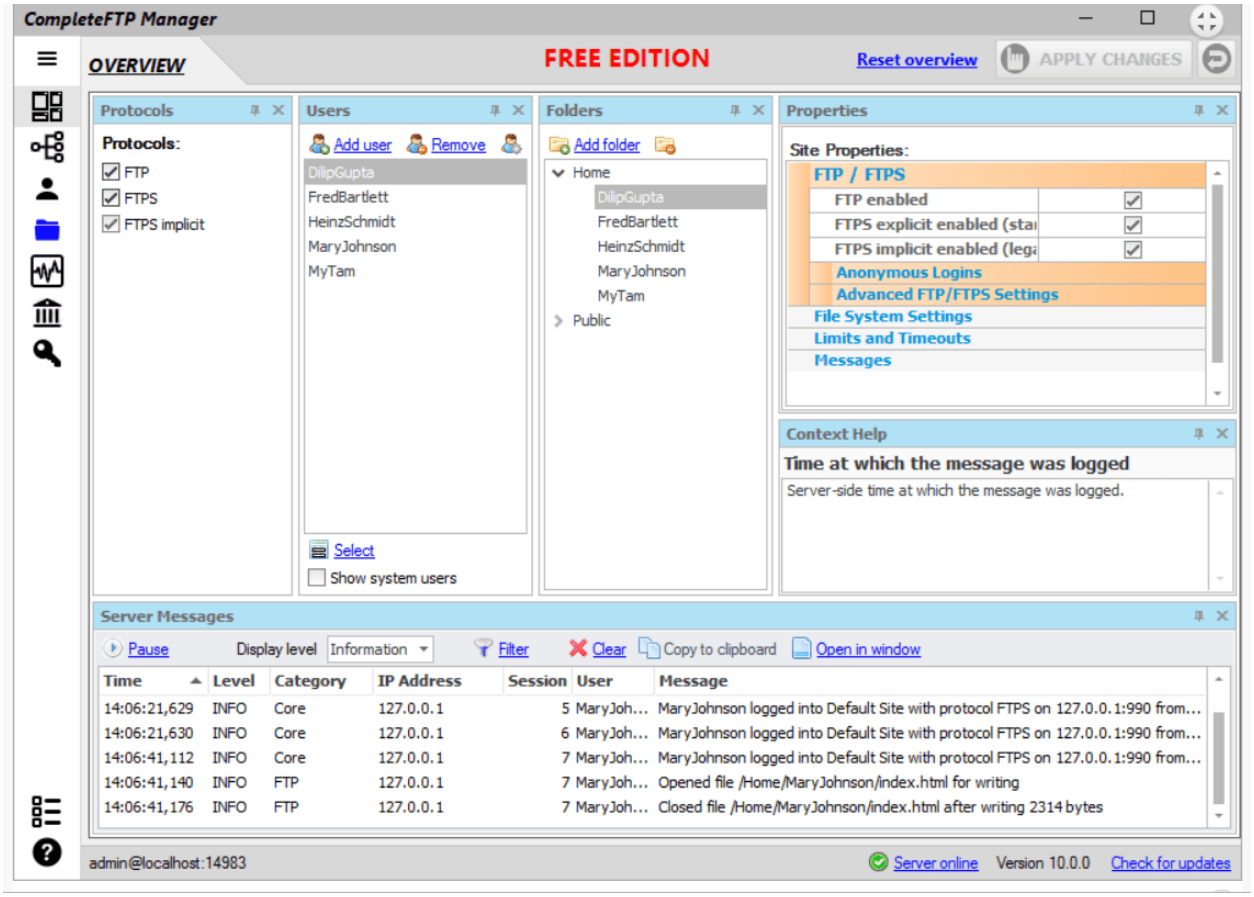

Рисунок 1.3 — Скриншот программы Complete FTP

Достоинства Complete FTP:

- Поддерживает зашифрованные передачи файлов.
- Много вариантов настройки.

Недостатки Complete FTP:

- Полное меню скрыто по умолчанию.
- Иногда возникают проблемы с производительностью.

1.2. Постановка задачи

После рассмотрения аналогов можно сказать, что все они обладают большим количеством функций, которые невозможно реализовать в курсовом проекте за данный период времени. Поэтому были выбраны несколько ключевых возможностей, которые будут выполнены в рамках одного семестра:

- Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню.
- В программе должна быть предусмотрена возможность хранения информации.
- Программа должна поддерживать обмен файлами.
- В программе должна быть реализована работа с протоколом FTP.
- Должна быть реализована возможность авторизации пользователей.

В качестве языка программирования выбран C++, по причине быстрой производительности, требуемой для обработки большого количества объектов и наличия опыта в использования данного языка. В качестве реализация графического интерфейса используется фреймворк Qt, основанный на языке C++. Qt обладает большим количеством базовых классов, которые позволяют создавать собственные классы для реализации графического интерфейса, удобной системой общения между виджетами приложения с помощью системы сигналов и слотов и хорошей документацией, позволяющей в быстрые сроки разбираться в устройстве Qt.

Данный список средств позволяет реализовать все задачи, выбранные для курсового проекта.

2. СИСТЕМНОЕ ПРОГРАММИРОВАНИЕ

После определения требований к функционалу разрабатываемого приложения его следует разбить на функциональные блоки. Такой подход упростит понимание проекта, позволит устранить проблемы в архитектуре, обеспечит гибкость и масштабируемость программного продукта в будущем путем добавления новых блоков.

2.1. Модуль авторизации

Модуль авторизации пользователя необходим для входа в приложение, это необходимо для выполнения различных действий, таких как, например, доступ к скачиванию файлов.

Модуль предоставляет доступ к функциям приложения. Его задача заключается в авторизации пользователей в приложении, определении имени и всех необходимых данных пользователя, зашедшего в приложение.

2.2 Модуль работы с приложением

Модуль работы с приложением необходим для взаимодействия пользователя с приложением и является неотъемлемой его частью. Тут осуществляется весь необходимый функционал и возможности:

- Просмотр файлов на сервере
- Скачивание файлов
- Получение каталогов в виде архива

2.3 Модуль пользовательского интерфейса

Модуль пользовательского интерфейса предназначен для взаимодействия пользователя с приложением, основываясь на представлении и визуализации данных.

Для разрабатываемого проекта создается простейший пользовательский интерфейс с помощью фреймворка Qt. Данный пользовательский интерфейс представляет собой набор пунктов меню, с которыми может взаимодействовать пользователь.

2.4 Модуль чтения и записи данных

Модуль чтения и записи данных необходим для взаимодействия с данными, которые хранит приложение, что очень важно при работе с проектом.

Данный модуль должен отвечать за считывание информации от пользователя при авторизации и оправки запросов, её записи в используемую область памяти для дальнейшей работы, а также считывание и запись всей другой необходимой информации по мере работы с приложением.

2.6 Модуль преобразования данных

Модуль преобразования данных необходим для преобразования данных, считываемых от пользователя или из приложения для корректной работы с ними.

Очень часто одни и те же данные используются для разных целей, например, считанное время, может сравниваться с другими данными в формате QTime или же выводиться на экран в формате QString, поэтому для корректной работы и отображения этих данных необходимо их преобразовать к нужному типу.

2.7 Модуль сети

Модуль сети предназначен для подключения пользователя к серверу и необходим для их взаимодействия в реальном времени.

Данный модуль должен отвечать за передачу файлов по протоколу FTP между пользователям и сервером, отслеживать активность клиента в приложении.

3 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ

В данном разделе описывается функционирование и структура разрабатываемого приложения.

Описание классов приложения

3.1 MainWindow

Данный класс является графическим представлением программы, он представляет собой окно, в котором происходит работа в приложении.

Поля:

- User *user - пользователь работающий в приложении.
- Server server - объект сервера для его инициализации.
- make_dir *make - объект для создания нового каталога.
- void clicked_button() - метод который обрабатывает сигнал с кнопок перехода по папкам
- void on_add_file_clicked - срабатывает при нажатии на кнопку добавления файла.
- void on_back_clicked() - метод срабатывает при нажатии на клавишу назад.
- void on_delete_file_clicked() - метод отвечает за удаления файла и обработку нажатия клавивиши удаления.

3.2 file

Структура содержащая информацию о файле.

Поля:

- QString name - содержит имя файла.
- QString type - содержит тип файла (директория, ссылка, регулярный файл).

3.3 Dir

Данный класс отвечает за хранение и получение информации о директории.

Поля:

- std::vector<struct file> all_files - хранит информацию о всех файлах в текущем каталоге
- int num - количество файлов в текущем каталоге
- void dirWalk - принимает char* path - путь к директории. Отвечает за рекурсивный обход директории.
- void current_dir - принимает char* path - путь к директории. Отвечает за обход директории.
- int get_num - возвращает количество файлов в текущем каталоге;
- void set_empty() - обнуляет счетчик количества файлов в директории и отчищает массив файлов.
- std::vector<struct file> get_all_files - возвращает массив содержащий информацию о все файлах.

3.4 User

Данный класс отвечает за хранение о работающем пользователе.

Поля:

- QString current_dir - строка хранящая путь к текущей директории.
- int id - хранит id пользователя
- Dir* all_files - содержит информацию о всех файлах в текущем каталоге.
- Int get_id - возвращает id пользователя.
- void set_current_dir - принимает int id - id пользователя. Устанавливает id пользователя.
- QString get_current_dir - возвращает путь к текущему каталогу.
- Dir* get_all_files - возвращает объект содержащий информацию о всех файлах в текущем каталоге.
- void set_current_dir - принимает QString current_dir - путь к каталогу. Устанавливает текущую директорию.

3.4 Server

Данный класс инициализирует сервер для управления входящими соединениями, а так же обрабатывает входящие команды.

Поля:

- int s20 - содержит значения сокета для 20 порта.

- int s21 - содержит значения сокета для 21 порта.

- int num_of_users21 - содержит количество пользователей 21 порта.

- int sa20 - файловый дескриптор для 20 порта.

- int sa21[10] - массив файловых дескрипторов для 21 порта.

- char buffer[10][1024] - буфер для чтения и отправки данных.

- struct sockaddr_in channel - структура содержащая настройки сокета.

- pthread_t tid - id потока.

- QStack<int>indexes - стек который хранит доступные id для пользователей.

- User *user - объект хранящий информацию о пользователе.

- static void* start_server21 - принимает void *s - указатель на объект Server. Функция ожидания запроса на 21 порту.

- static void* new_connection21 - принимает void *s - указатель на объект Server. Функция ожидает входных данных от клиента на 21 порту.

- int menu - функция обрабатывает команды пришедшие от клиента.

3.5 make_dir

Данный класс отвечает за создания папок.

Поля:

- User *user - пользователь-хозяин сервера.

- void on_ok_clicked() - метод отвечающий за обработку сигнала с нажатой кнопки создания.

- void mainWindow() - сигнал который отправляется в MainWindow для обновления таблицы с файлами.

- void set_user() - метод который используется для передачи пользователя из класса MainWindow в make_dir.

4 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ

4.1 Метод new_connection21() класса Server.

Шаг 1. Начало.

Шаг 2. Создаем нового юзера и указываем корневой каталог.

Шаг 3. Запускаем бесконечный цикл.

Шаг 4. Блокируем семафор.

Шаг 5. Читаем из сокета команду.

Шаг 6. Если количество прочитанных байт <= 0, то продолжаем чтение.

Шаг 7. Отправляем команду методу обработки команд.

Шаг 8. Проверяем код возврата метода с -1, что говорило бы о том что клиент отключился.

Шаг 9. Если клиент не отключился возвращаемся в начало цикла.

Шаг 10. Иначе закрываем файловый дескриптор и заносим в стек id пользователя.

Шаг 11. Конец.

4.2 Метод showAll() класса MainWindow

Шаг 1. Начало.

Шаг 2. Зададим количество столбцов равное 5.

Шаг 3. Если мы находим в корне, то скроем кнопку Назад, иначе покажем ее.

Шаг 4. Вызываем у пользователя метод отвечающий за обход каталога.

Шаг 5. Заходим в цикл по каждому файлу в каталоге.

Шаг 6. Добавляем кнопку выбора в 0 колонку.

Шаг 7. В 1 колонку добавляем имя файла.

Шаг 8. В 3 колонку пишем тип(файл, папка).

Шаг 9. Если тип - папка, то создаем кнопку для перехода в нее и добавляем ее в 4 колонку таблицы.

Шаг 10. Если тип - папка, то заносим в 5 колонку 1, иначе 0.

Шаг 11. Очищаем массив хранящий все каталоги.

Шаг 12. Конец.

5 ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ

Тестирование — это наблюдение за работой приложения в разных искусственно созданных ситуациях. Данные, полученные в ходе тестирования, важны при планировании последующей стратегии развития приложения. Это своего рода диагностика, которая влияет на многие дальнейшие действия.

5.1 Тестирование работы приложения со стороны сервера.

Данное приложение работает по клиент-серверной модели, со стороны сервера программа может добавлять, удалять и создавать различные папки

Пример добавления файл на сервер представлен на рисунках 5.1.1, 5.1.2

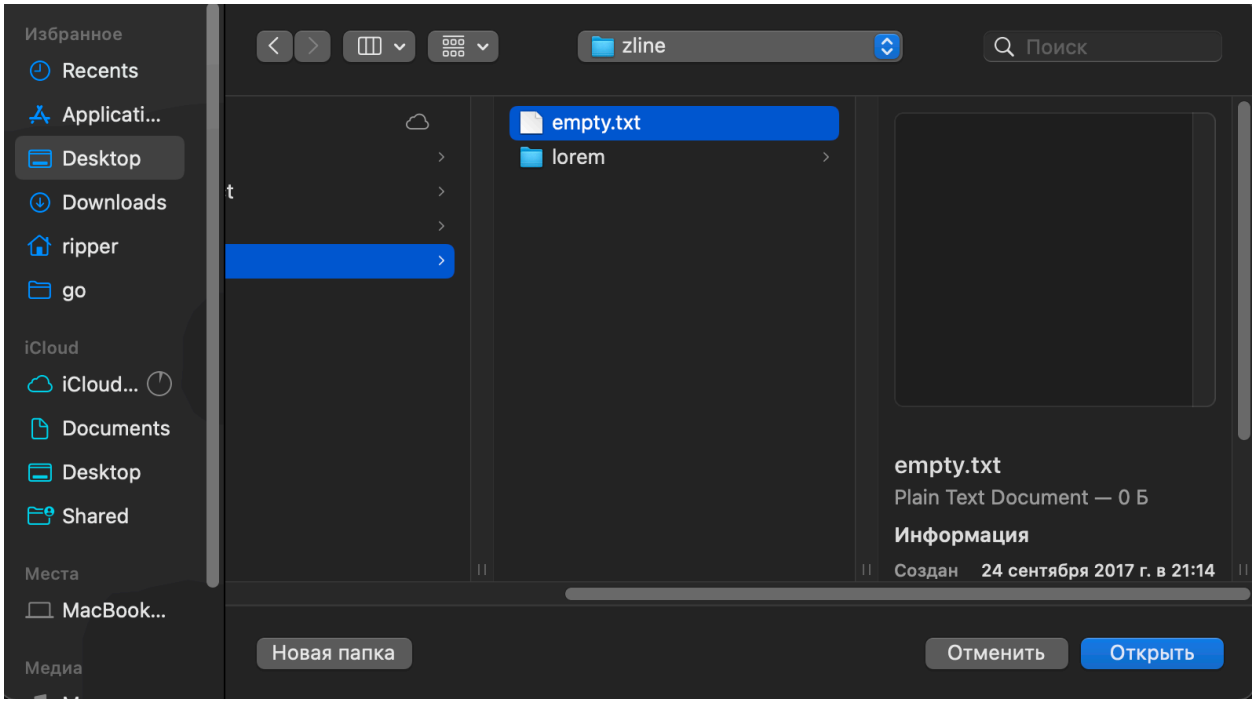

Рисунок 5.1.1 — Добавление файла на сервер

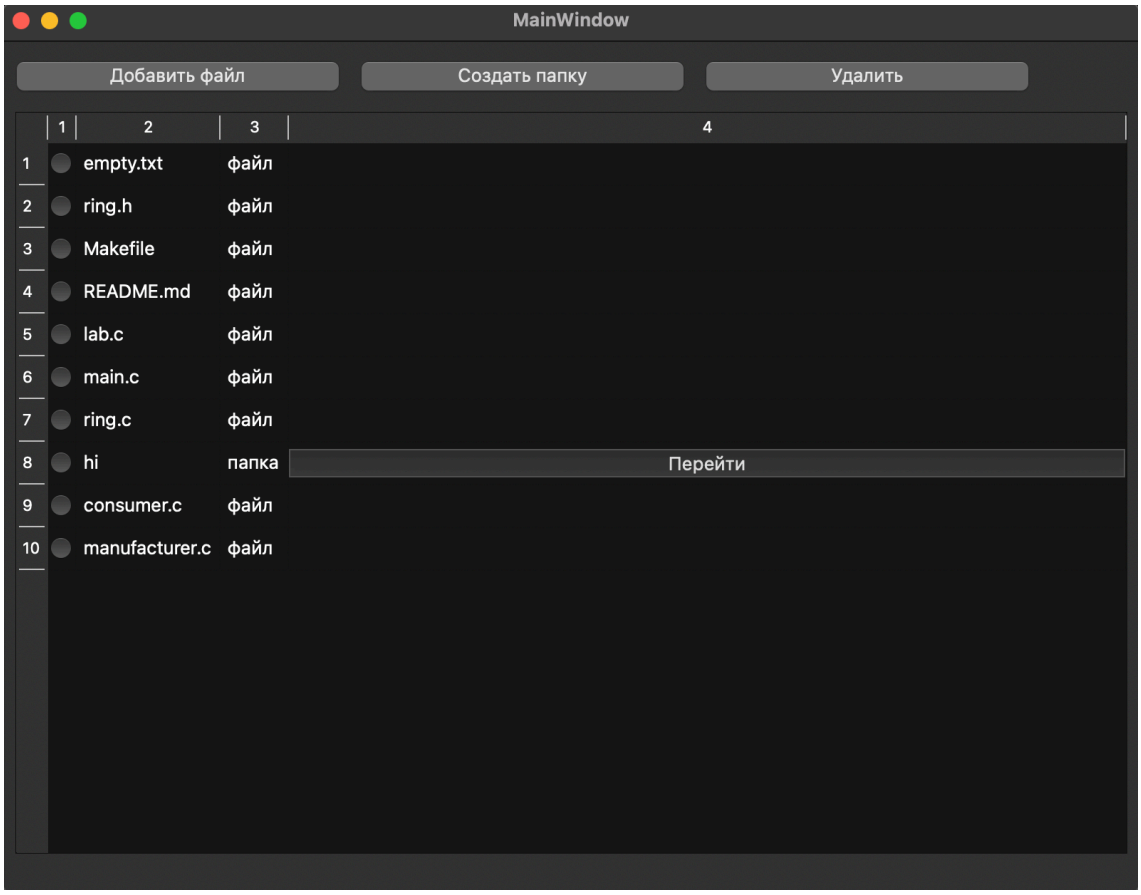

Рисунок 5.1.2 — Отоброжение добавленного файла на сервере

Создание папки представлено на рисунке 5.1.3

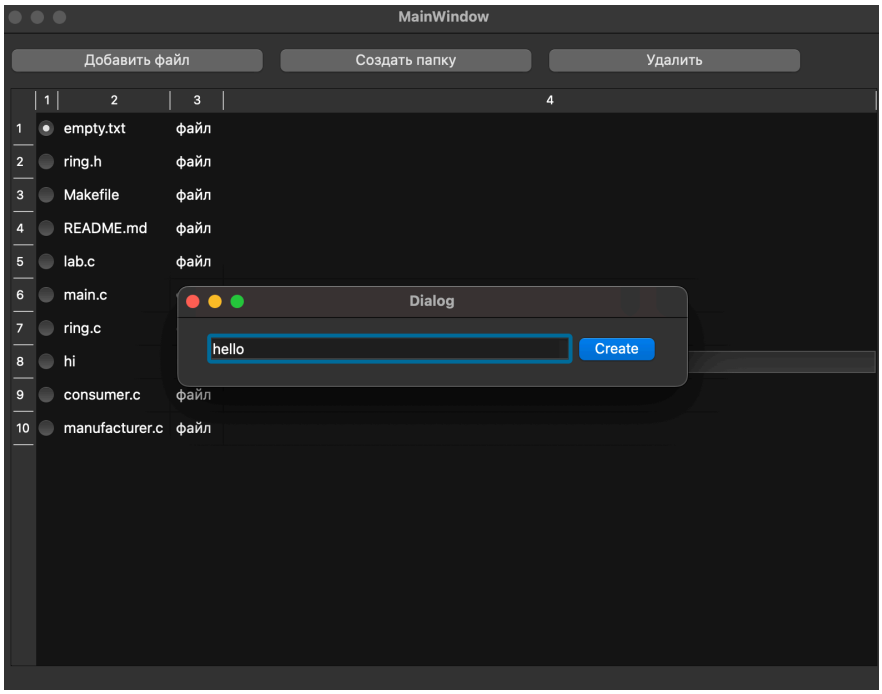

Рисунок 5.1.3 — Создание папки на сервере

5.2 Тестирование приложения со стороны клиента.

Пример подключения к серверу и просмотра каталога представлен на рисунке 5.2.1

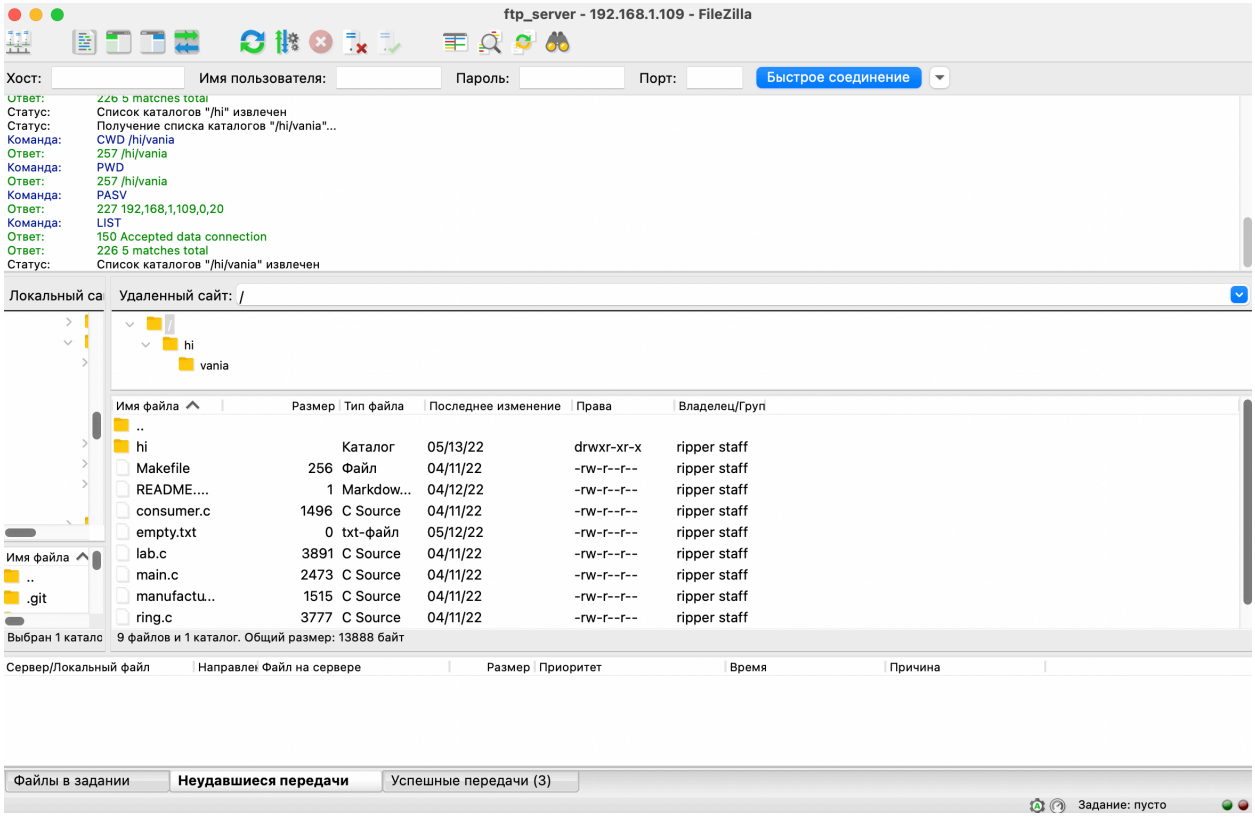

Рисунок 5.2.1 — Подключение и просмотр файлов на сервере

Скачивание файла с сервера представлено на рисунке 5.2.2

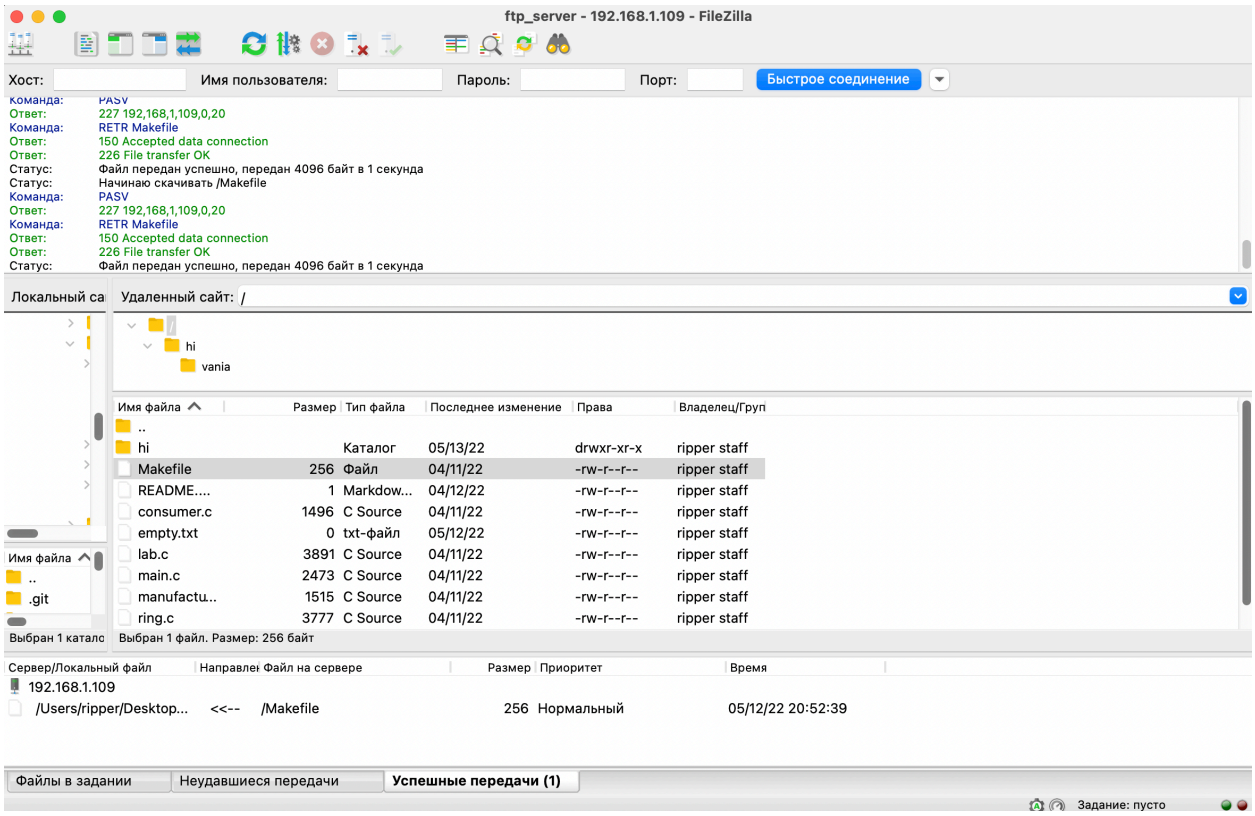

Рисунок 5.2.2 — Скачивание файла с сервера

Получение каталога в архивном виде представлено на рисунках 5.2.3, 5.2.4

```
227 192,168,1,109,0,20
150 Accepted data connection
-rw-r--r--  1 ripper staff 0 May 12 2022  empty.txt
-rw-r--r--  1 ripper staff 479 Apr 11 2022  ring.h
-rw-r--r--  1 ripper staff 256 Apr 11 2022  Makefile
-rw-r--r--  1 ripper staff 1 Apr 12 2022  README.md
-rw-r--r--  1 ripper staff 3891 Apr 11 2022  lab.c
-rw-r--r--  1 ripper staff 2473 Apr 11 2022  main.c
-rw-r--r--  1 ripper staff 3777 Apr 11 2022  ring.c
drwxr-xr-x  5 ripper staff 160 May 13:29:40 2022  hi
-rw-r--r--  1 ripper staff 1496 Apr 11 2022  consumer.c
-rw-r--r--  1 ripper staff 1515 Apr 11 2022  manufacturer.c
WARNING! 10 bare linefeeds received in ASCII mode
File may not have transferred correctly.
226 5 matches total
ftp> get hi hi.tar
227 192,168,1,109,0,20
150 Accepted data connection
226 File transfer OK
```

Рисунок 5.2.3 — Скачивание каталога с сервера

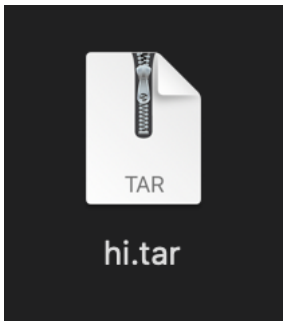

Рисунок 5.2.4 — Полученный архив

6 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

ЗАКЛЮЧЕНИЕ

В результате работы над данным курсовым проектом было разработано работоспособное приложение со своим набором функций и графическим интерфейсом. Данный курсовой проект был разработан в соответствии с поставленными задачами, весь функционал был реализован в полном объеме.

В ходе разработки были углублены знания языка программирования C++ и в области объектно-ориентированного программирования, а также получен опыт работы с Фреймворком Qt и с графическим интерфейсом.

Работа была разделена на такие этапы, как анализ существующих аналогов, литературных источников, постановка требований к проектируемому программному продукту, системное и функциональное проектирование, конструирование программного продукта, разработка программных модулей и тестирование проекта. После последовательного выполнения вышеперечисленных этапов разработки было получено исправно работающее приложение.

В дальнейшем планируется усовершенствование текущего функционала приложения, путем улучшения графического интерфейса, добавления новых функций и модулей, а также добавления возможности работы под разными системами.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Керниган Б. Язык программирования С/ 4-е издание М.:Питер, 2004. – 923 с.
2. Лав Р. Системное программирование на Linux/ 2-е издание 2014.
3. Таненбаум Э. Компьютерные сети 5-е издание 2019.

ПРИЛОЖЕНИЕ А
(обязательное)

Схема структурная

ПРИЛОЖЕНИЕ Б
(обязательное)

Диаграмма классов

ПРИЛОЖЕНИЕ В

(обязательное)

Ведомость документов