

CSE 344
System Programming
Project Report

Yavuz Selim İkizler
1901042617

Code Design

General Structure:

The program consists of two parts: client.c and server.c. On the server side, client connection requests are received within a while loop and added to pre-created threads. These threads invoke the client handler function for each client. If a client connects to the server for the first time, it calls the sync_directories function, which downloads the server's content to the newly connected client. After that, the server continuously receives commands from that client within a while loop and sends responses.

There are two different commands. One is called for client synchronization. If the client receives the sync command, it calls the sync_directories function to perform the necessary operations. The other command is used to modify files on the client side and synchronize them with the server. It calls the handle_change function to perform the necessary operations. In order to ensure synchronization for each received command, mutex lock and unlock calls are placed at the beginning and end of the respective functions. This allows for synchronization between multiple clients.

On the client side, the program is initially executed by sending the sync command to the server. Then, within a while loop, the client checks if there are any changes in its folder. If changes are detected, the client sends the change command to the server. If no changes are found, the synchronization command is sent again. After that, the client is put to sleep for a certain period (10 second) using the sleep call. Then, the same process is repeated.

Signal handlers are available for both the client and the server.

Some global struct variables are used for operations.

```
typedef struct{
    char file_name[MAX_FILENAME_LENGTH];
    char f_name[MAX_FILENAME_LENGTH];
    char status[MAX_FILENAME_LENGTH];
}file_info;
```

The file_info struct stores the properties of each modified file sent from the client. It contains the following fields:

file_name: This field holds the path of the file.

f_name: This field stores the name of the file without the path.

status: This field indicates the action to be taken for the file. It holds information on whether the file will be modified, added, or removed.

```
typedef struct {
    char client_directory[256];
    char command[256];
    file_info send_file[BUFFER_SIZE];
    int file_number;
} Command;
```

The Command struct is used for each command sent by the client. The data from the client arrives in this data type. It includes the following fields:

client_directory: This field is a char array that holds the directory name of the client sending the command.

command_variable: This field stores the type of command, either "sync" or "change".

file_info: This field is an array of file_info structs. It holds the information about the files to be modified. Each element of the array represents a file and its associated information.

file_number: This field holds the count of the files sent by the client.

```
typedef struct {
    char source[MAX_FILENAME_LENGTH];
    char destination[MAX_FILENAME_LENGTH];
    int source_descriptor;
    int destination_descriptor;
} FileCopyInfo;
```

FileCopyInfo stores the file paths and descriptors of the files to be copied.

Server.c:

The server.c main function is responsible for creating and managing threads that utilize the client_handler function. Within the client_handler function, there are handle_sync and handle_change functions to handle the commands. Additionally, there are helper functions such as read_dir, file_copy, create_path, and create_file.

Main:

The code declares several variables, including server_sock, client_sock, and port_no, which are used to manage the server socket and client connections. max_thread is initialized with the value passed as the second argument (argv[2]). This specifies the maximum number of threads to be used in the thread pool. The program checks the number of command-line arguments (argc). If it's less than 4, an error message is printed, indicating the correct usage of the program, and the program exits. A signal handler for the SIGINT signal is registered using signal(SIGINT, handle_sigint). This allows the program to handle interruptions such as pressing Ctrl+C gracefully. The port_no variable is initialized with the value passed as the

third argument (argv[3]). The program checks if the specified directory (argv[1]) exists. If it does not exist, the code creates the directory using mkdir with the appropriate permissions (0700).

A server socket is created using socket(AF_INET, SOCK_STREAM, 0).

The server address structure (server_addr) is initialized and bound to the server socket using bind. The server socket starts listening for incoming connections using listen(server_sock, 5).

Another SIGINT signal handler is registered, likely to handle interruption during the server loop. A mutex (commandMutex) is initialized using pthread_mutex_init to synchronize access to shared resources. The program enters a while loop that continuously accepts client connections using accept. For each accepted connection, a new thread is created to handle the client request. A ThreadArgs struct is allocated and populated with the client socket and the directory name (argv[1]) to be passed to the thread function.

The new thread is created using pthread_create, passing the thread ID, thread attributes, the thread function (client_handler), and the ThreadArgs struct as arguments.

If the thread creation fails, an error message is printed, and the error should be handled appropriately. The j variable is incremented to keep track of the number of threads created.

After the thread creation loop, another loop for (int i = 0; i < j; i++) is used to join all the threads using pthread_join. This ensures that the main program waits for all threads to finish before exiting. Some cleanup is performed, including freeing dynamically allocated memory, destroying the mutex, and returning 0 to indicate successful execution.

Client_handler:

```
struct ThreadArgs* threadArgs = (struct ThreadArgs*) args;
int sock = threadArgs->client_sock;
char* directory = threadArgs->arg;
```

When sending a thread, the structure taken as a parameter is assigned. The "sock" parameter assigns the client socket number, and the "directory" parameter assigns the server directory.

```
memset(buffer, 0, sizeof(buffer));
read(sock, buffer, sizeof(buffer) - 1);

printf("The client connected with id: %s\n", buffer);

snprintf(buffer, sizeof(buffer), "%s", directory);
```

It retrieves the ID of the connected client and writes it, and as feedback, it sends the server directory.

```
while(1){

    Command client_data;
    // Lock the mutex before reading commands
```

```

    ssize_t bytes_received = recv(sock, &client_data, sizeof(Command), 0);
    if (bytes_received < 0) {
        fprintf(stderr, "Error occurred while receiving data.\n");
        // exit(1);
    }

```

It continuously receives commands from the client and assigns the command to the Command structure.

```

// printf("%s",client_data.client_directory);
    if(strncmp(client_data.command, "SYNC", 4) == 0){
        handle_sync(directory,client_data.client_directory);
        snprintf(buffer, sizeof(buffer), "%s", "update");
        write(sock, buffer, strlen(buffer) + 1);
    }
    if(strncmp(client_data.command, "change", 6) == 0){
        // printf("\n %s \n",client_data.send_file[0].file_name);
        strcpy(change_file_list[list_size],client_data.client_directory);
        list_size++;
        handle_change(client_data.send_file,client_data.file_number,directory)
    }
;

    snprintf(buffer, sizeof(buffer), "%s", "changed");
    write(sock, buffer, strlen(buffer) + 1);
}

```

It checks the commands here, calls the appropriate functions, and after the functions finish the operation, it sends the appropriate feedback. Sends the functions appropriate variables includes file list of the changed files, size of the list, the source directory.

```

// Close the client socket
close(sock);
pthread_exit(NULL);

```

It frees the finished sockets and threads.

Handle_sync:

```

pthread_mutex_lock(&commandMutex);
DIR* dir;
DIR * dir2;

```

```

    dir = opendir(sourceDir);
    dir2=opendir(destDir);
    if (dir == NULL) {
        perror("Error opening source directory");
        // pthread_exit(NULL);
    }
    buffer = ( FileCopyInfo*)malloc(MAX_BUFFER_SIZE * sizeof( FileCopyInfo));

```

It opens the directory of the files to be copied and sends them to the read_dir function to read the files. It allocates the buffer to hold the files.

```

read_dir(dir,dir2,sourceDir,destDir);

file_copy();
free(buffer);

```

It calls the read_dir function to read the files, and then calls the copy_file function to write the read files to the client's directory. It frees the buffer after the operation is completed.

Read_dir:

```

int j=0;
while ((entry = readdir(dir)) != NULL) {

    // Skip "." and ".." directories
    if (strcmp(entry->d_name, ".") == 0 || strcmp(entry->d_name, "..") ==
0) {
        continue;
    }

    // Create source and destination file paths
    snprintf(sourcePath, MAX_FILENAME_LENGTH, "%s/%s", sourceDir, entry-
>d_name);
    snprintf(destPath, MAX_FILENAME_LENGTH, "%s/%s", destDir, entry-
>d_name);

```

It reads the files one by one using the dirent structure, and combines the paths for the source and destination directories using snprintf.

```

if(entry->d_type == DT_DIR){

    struct stat st;
    if (stat(destPath, &st) == 0) {
        if (S_ISDIR(st.st_mode)) {
            // printf("Directory already exists.\n");

```

```

        // return ;
    else{
        // Create the directory if it doesn't exist
        if (mkdir(destPath, 0700) == 0) {
            // printf("Directory created successfully.\n");

        } else {
            printf("Failed to create the directory.\n");
        }
    }
    dir2_second=opendir(destPath);
    dir2=opendir(sourcePath);
    read_dir(dir2,dir2_second,sourcePath,destPath);
    closedir(dir2);
    closedir(dir2_second);

```

If the read file is a directory, it checks if the corresponding destination directory exists in the client's directory. If it doesn't exist, it creates a subdirectory in the client's directory. It opens the subdirectory to read files in it and recursively calls itself to read files in that subdirectory as well.

```

// Copy file information into the buffer
strcpy(buffer[bufferIndex].source, sourcePath);
strcpy(buffer[bufferIndex].destination, destPath);
// Open source file for reading
buffer[bufferIndex].source_descriptor = open(buffer[bufferIndex].source,
O_RDONLY);
if (buffer[bufferIndex].source_descriptor == -1) {
    perror("Error opening source file");
    return ;
}

// Create or truncate destination file for writing
buffer[bufferIndex].destination_descriptor=
open(buffer[bufferIndex].destination, O_WRONLY | O_CREAT | O_TRUNC, 0666);
if (buffer[bufferIndex].destination_descriptor== -1) {
    perror("Error opening destination file");
    close(buffer[bufferIndex].source_descriptor);
    return ;
}

bufferIndex++;

```

It saves the source and destination paths to the buffer. It also opens as a read mode for source (server) for destination(client) and saves the descriptors of the files. The buffer index is then incremented by 1.

Then it reads the client directory and compares it with the files on the server. If there is a different file in the client directory, it deletes it.

File_copy:

It reads the files in the buffer one by one and calls the copy_file function, passing the source and destination directories, to perform the copy operations. When all the files have been read from the buffer and the buffer index reaches 0, it exits the loop.

Copy_file:

It writes the files read from the source to the destination using the descriptors in the buffer.

Handle_change:

It takes the list of files to be modified as a parameter and the server directory.

It runs a for loop based on the number of files in the list. It removes the part of the path that includes the client directory and replaces it with the server directory. It calls appropriate helper methods based on whether the files in the list need to be removed, added, or modified. These file operations are performed by extracting the status of the files and comparing them.

CreatePath:

The createPath function creates a directory based on the provided string.

CreateFile:

The CreateFile function reads and writes files based on the provided string path.

Client.c:

The main function includes the commands to perform, such as change_directory and sync_directories. It also includes the check_difference function to check for directory changes, and the read_folder and read_dir functions for file reading operations.

Main:

It defines an integer variable `file_number` and initializes it to 0. It checks if the command line argument count (`argc`) is less than 4. If it is, it prints an error message indicating the correct usage of the program and exits with an error code. It sets up a signal handler for the `SIGINT` signal (interrupt signal). It converts the second command line argument (`argv[2]`) to an integer using `atoi()` and assigns it to the variable `port_no`. It creates a socket using the `socket()` system call. If the socket creation fails, it prints an error message and exits. It uses the `gethostbyname()` function to get the server's host information based on the third command line argument (`argv[3]`). If the host information is not found (server is `NULL`), it prints an error message and exits. It initializes the `server_addr` structure with zeros using `memset()` and sets the address family to `AF_INET`. It copies the server address obtained from `gethostbyname()` to `server_addr.sin_addr.s_addr` and sets the port number to `port_no`. It attempts to establish a connection to the server using the `connect()` system call. If the connection fails, it prints an error message and exits.

First, it sends the IP address to connect to the server. It receives the server's directory name as a notification from the server. It checks if the client directory name received from `argv` exists or not. If it doesn't exist, it creates a directory with that name.

Afterward, it calls the `sync_directories` function to retrieve the files from the server. Then, it allocates the `before` variable and sends it to the `read_folder` function. This function saves the descriptors of the existing files, and `before` serves as a list to hold those descriptors.

Then, a while loop is initiated. Within this while loop, first, the current files in the client directory are read by calling the `read_folder` function, with `after` being passed as a parameter, which serves as a list to hold file descriptors. Then, the `check_difference` function is called, passing `before` and `after` variables. This function compares these lists, and if there is a difference, it returns 1 and the necessary parameters are passed to call the `change_server` function to send the change command to the server. If there is no difference, the `sync_directories` function is called to send the synchronization command. Finally, the loop is paused using `sleep`, and the same set of operations continues continuously.

Sync_directories:

It creates a sync command to send to the server and initiates the synchronization process on the server side.

Read_folder:

It opens the client directory, calls the `read_dir` function, and passes it to read the directory.

Read_dir:

It reads the files in the current directory within the loop, saves their descriptors to the `FileSave` list, and writes them. To handle subdirectories, it is recursively called.

Chaneg_server:

It adds the files to be modified to the appropriate locations in the command structure, creates a change command with that structure, sends it to the server, and waits for a response.

Check_Difference:

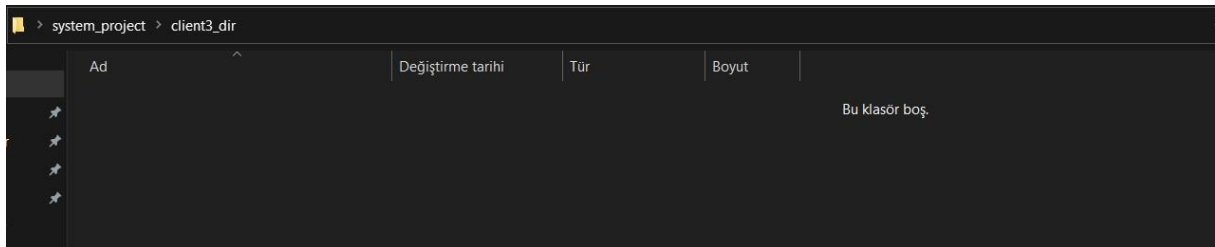
It compares the two file save lists, one being before and the other being after, to detect the differences. If there are more files in after compared to before, it takes those additional files, sets their status as "add," and adds them to the file_info struct array. If a file is missing in after compared to before, it sets its status as "remove" and adds it to the file_info array. If a file is different, it sets its status as "change." Finally, it closes the descriptors in before and sets before equal to after.

Test Outputs

Connect:

```
server_dir
update
update
update
update
update
tting down server
rmiles001@LAPTOP-TCV809A5:/mnt/c/Users/yavuz/Desktop/system_project$ sudo ./BibakBoxServer server_dir 5 128
[sudo] password for rmiles001:
The client connected with id: 467
```

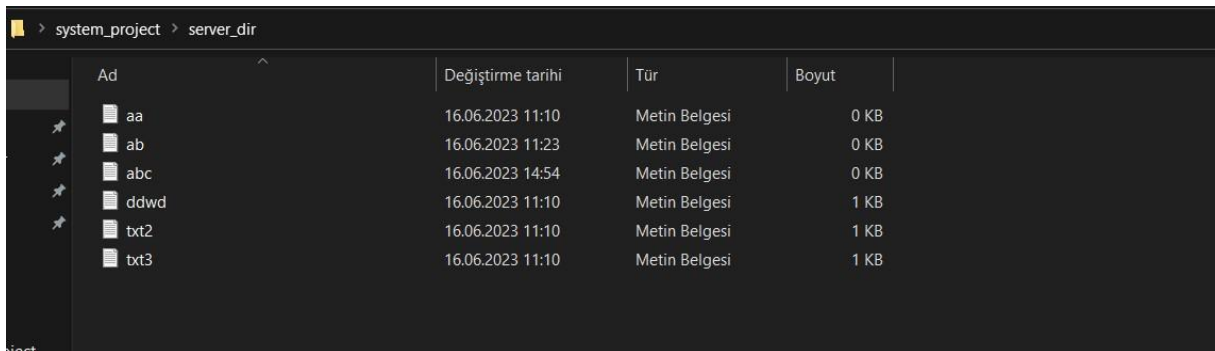
Client3_dir(before):



The screenshot shows a file explorer window with the path > system_project > client3_dir. The table below represents the content of this directory.

| Ad | Değiştirme tarihi | Tür | Boyut |
|----------------|-------------------|-----|-------|
| Bu klasör boş. | | | |

Server_dir:



The screenshot shows a file explorer window with the path > system_project > server_dir. The table below represents the content of this directory.

| Ad | Değiştirme tarihi | Tür | Boyut |
|------|-------------------|---------------|-------|
| aa | 16.06.2023 11:10 | Metin Belgesi | 0 KB |
| ab | 16.06.2023 11:23 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 14:54 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 11:10 | Metin Belgesi | 1 KB |
| txt2 | 16.06.2023 11:10 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 11:10 | Metin Belgesi | 1 KB |

Client3_dir(after):

> system_project > client3_dir

| Ad | Değiştirme tarihi | Tür | Boyut |
|------|-------------------|---------------|-------|
| aa | 16.06.2023 17:54 | Metin Belgesi | 0 KB |
| ab | 16.06.2023 17:54 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 17:54 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 17:54 | Metin Belgesi | 1 KB |
| txt2 | 16.06.2023 17:54 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 17:54 | Metin Belgesi | 1 KB |

Sig Handler:

Server, Client:

```
^Cserver_dir client3_dir server_dir client3_dir server_dir client3_dir server_dir client3_dir Shutting down server  
^CShutting down client
```

Server Operations:

Server: server_dir

| Ad | Değiştirme tarihi | Tür | Boyut |
|------|-------------------|---------------|-------|
| aa | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| ab | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt2 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |

Client: client1_dir, client2_dir, client3_dir, client4_dir, client5_dir

| Ad | Değiştirme tarihi | Tür | Boyut |
|------|-------------------|---------------|-------|
| aa | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| ab | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt2 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |

> system_project > client2_dir

| Ad | Değiştirme tarihi | Tür | Boyut |
|------|-------------------|---------------|-------|
| aa | 16.06.2023 18:28 | Metin Belgesi | 1 KB |
| ab | 16.06.2023 18:28 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 18:28 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 18:28 | Metin Belgesi | 1 KB |
| txt2 | 16.06.2023 18:28 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 18:28 | Metin Belgesi | 1 KB |

> system_project > client3_dir

| Ad | Değiştirme tarihi | Tür | Boyut |
|------|-------------------|---------------|-------|
| aa | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| ab | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt2 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |

> system_project > client4_dir

Ad

Değiştirme tarihi

Tür

Boyut

aa

16.06.2023 18:29

Metin Belgesi

1 KB

ab

16.06.2023 18:29

Metin Belgesi

0 KB

abc

16.06.2023 18:29

Metin Belgesi

0 KB

ddwd

16.06.2023 18:29

Metin Belgesi

1 KB

txt2

16.06.2023 18:29

Metin Belgesi

1 KB

txt3

16.06.2023 18:29

Metin Belgesi

1 KB

Personal

system_project > client5_dir

| Ad | Değiştirme tarihi | Tür | Boyut |
|------|-------------------|---------------|-------|
| aa | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| ab | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt2 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |

Add:

Server

system_project > server_dir

| Ad | Değiştirme tarihi | Tür | Boyut |
|---------|-------------------|---------------|-------|
| aa | 16.06.2023 11:10 | Metin Belgesi | 0 KB |
| ab | 16.06.2023 11:23 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 14:54 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 11:10 | Metin Belgesi | 1 KB |
| new_txt | 16.06.2023 18:20 | Metin Belgesi | 0 KB |
| txt2 | 16.06.2023 11:10 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 11:10 | Metin Belgesi | 1 KB |

Client1

system_project > client1_dir

| Ad | Değiştirme tarihi | Tür | Boyut |
|---------|-------------------|---------------|-------|
| aa | 16.06.2023 18:21 | Metin Belgesi | 0 KB |
| ab | 16.06.2023 18:21 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 18:21 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 18:21 | Metin Belgesi | 1 KB |
| new_txt | 16.06.2023 18:21 | Metin Belgesi | 0 KB |
| txt2 | 16.06.2023 18:21 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 18:21 | Metin Belgesi | 1 KB |

Client2

> system_project > client2_dir

| Ad | Değiştirme tarihi | Tür | Boyut |
|---------|-------------------|---------------|-------|
| aa | 16.06.2023 18:21 | Metin Belgesi | 0 KB |
| ab | 16.06.2023 18:21 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 18:21 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 18:21 | Metin Belgesi | 1 KB |
| new_txt | 16.06.2023 18:21 | Metin Belgesi | 0 KB |
| txt2 | 16.06.2023 18:21 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 18:21 | Metin Belgesi | 1 KB |

Client3

> system_project > client3_dir

| Ad | Değiştirme tarihi | Tür | Boyut |
|---------|-------------------|---------------|-------|
| aa | 16.06.2023 18:21 | Metin Belgesi | 0 KB |
| ab | 16.06.2023 18:21 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 18:21 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 18:21 | Metin Belgesi | 1 KB |
| new_txt | 16.06.2023 18:21 | Metin Belgesi | 0 KB |
| txt2 | 16.06.2023 18:21 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 18:21 | Metin Belgesi | 1 KB |

Client4

> system_project > client4_dir

| Ad | Değiştirme tarihi | Tür | Boyut |
|---------|-------------------|---------------|-------|
| aa | 16.06.2023 18:22 | Metin Belgesi | 0 KB |
| ab | 16.06.2023 18:22 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 18:22 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 18:22 | Metin Belgesi | 1 KB |
| new_txt | 16.06.2023 18:22 | Metin Belgesi | 0 KB |
| txt2 | 16.06.2023 18:22 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 18:22 | Metin Belgesi | 1 KB |

Client5

> system_project > client5_dir

| Ad | Değiştirme tarihi | Tür | Boyut |
|---------|-------------------|---------------|-------|
| aa | 16.06.2023 18:22 | Metin Belgesi | 0 KB |
| ab | 16.06.2023 18:22 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 18:22 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 18:22 | Metin Belgesi | 1 KB |
| new_txt | 16.06.2023 18:22 | Metin Belgesi | 0 KB |
| txt2 | 16.06.2023 18:22 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 18:22 | Metin Belgesi | 1 KB |

Change:

*aa - Not Defteri

| Dosya | Düzen | Biçim | Görünüm | Yardım |
|-------|-------|-------|---------|--------|
| | | | | |

Server

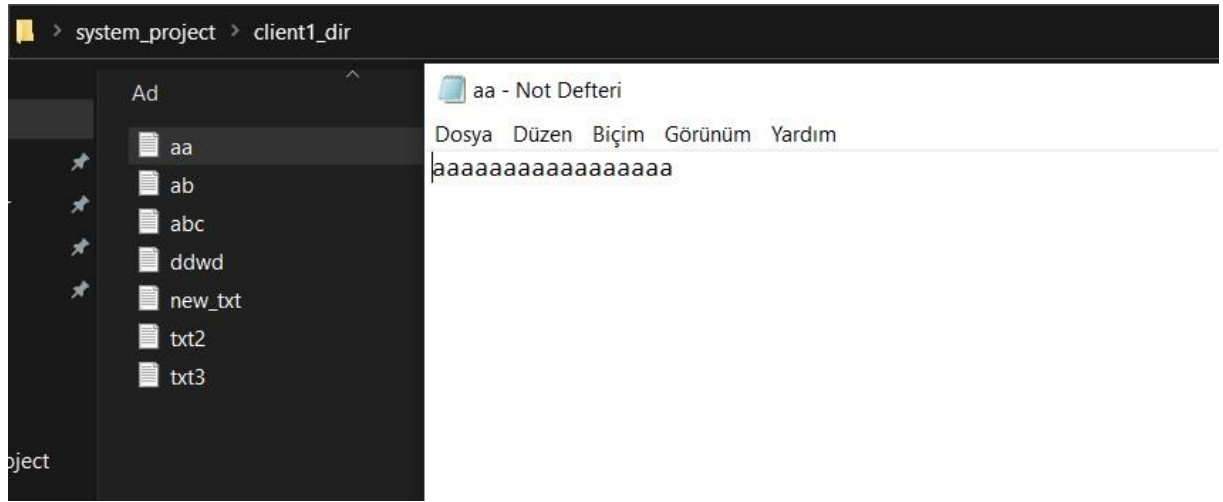
> system_project > server_dir

| Ad |
|------|
| aa |
| ab |
| abc |
| ddwd |
| txt2 |
| txt3 |

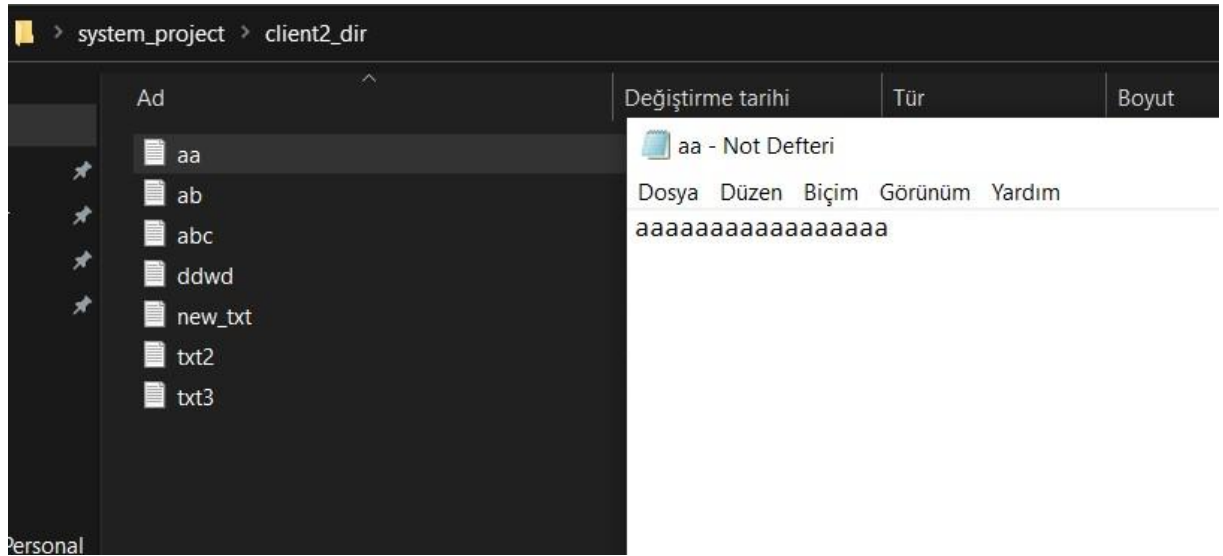
aa - Not Defteri

| Dosya | Düzen | Biçim | Görünüm | Yardım |
|----------------------|-------|-------|---------|--------|
| aaaaaaaaaaaaaaaaaaaa | | | | |

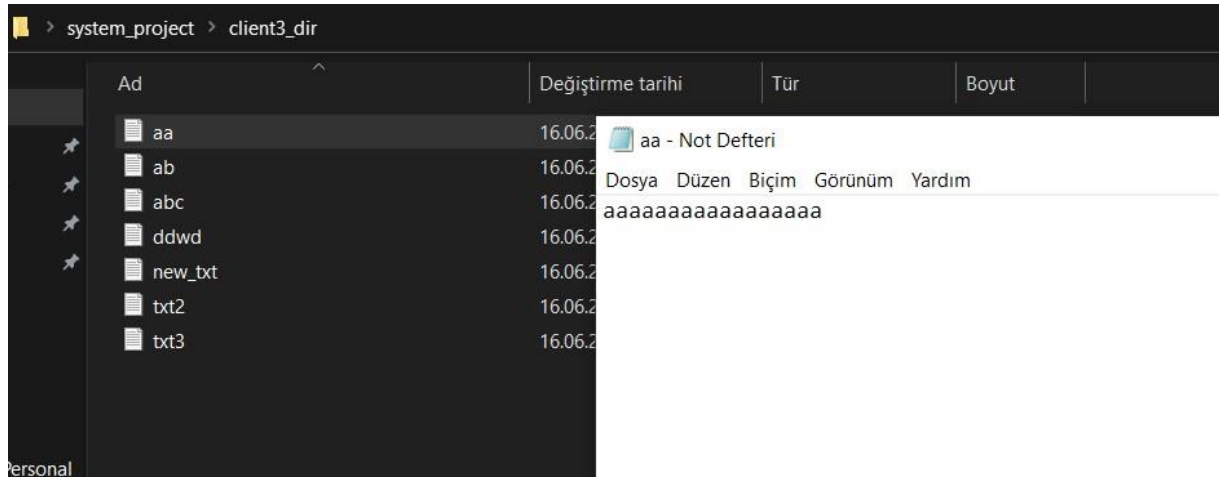
Client1



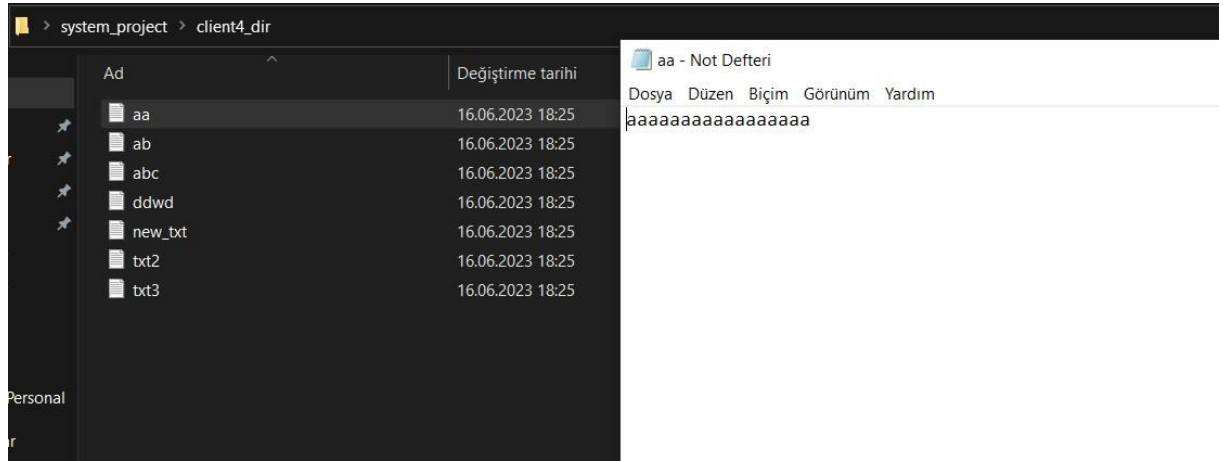
Client2



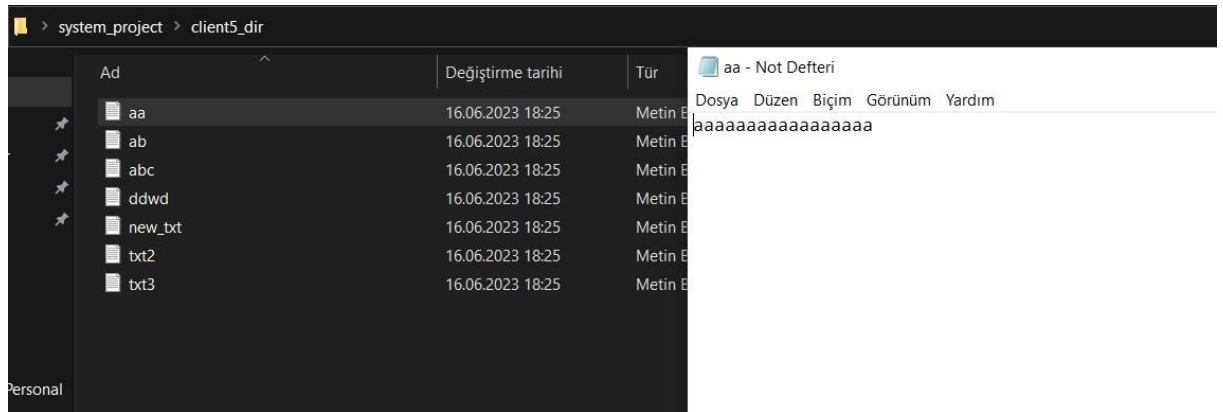
Client3



Client4



Client5



Remove:

Server

| Ad | Değiştirme tarihi | Tür | Boyut |
|------|-------------------|---------------|-------|
| aa | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| ab | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt2 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |

Client1

| Ad | Değiştirme tarihi | Tür | Boyut |
|------|-------------------|---------------|-------|
| aa | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| ab | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt2 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |

Client2

| | | | | |
|------------------------------|------|-------------------|---------------|-------|
| system_project > client2_dir | Ad | Değiştirme tarihi | Tür | Boyut |
| ✳ | aa | 16.06.2023 18:28 | Metin Belgesi | 1 KB |
| ✳ | ab | 16.06.2023 18:28 | Metin Belgesi | 0 KB |
| ✳ | abc | 16.06.2023 18:28 | Metin Belgesi | 0 KB |
| ✳ | ddwd | 16.06.2023 18:28 | Metin Belgesi | 1 KB |
| | txt2 | 16.06.2023 18:28 | Metin Belgesi | 1 KB |
| | txt3 | 16.06.2023 18:28 | Metin Belgesi | 1 KB |

Client3

system_project > client3_dir

| Ad | Değiştirme tarihi | Tür | Boyut |
|------|-------------------|---------------|-------|
| aa | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| ab | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt2 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |

Client4

system_project > client4_dir

| Ad | Değiştirme tarihi | Tür | Boyut |
|------|-------------------|---------------|-------|
| aa | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| ab | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt2 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |

Personal

Client5

system_project > client5_dir

| Ad | Değiştirme tarihi | Tür | Boyut |
|------|-------------------|---------------|-------|
| aa | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| ab | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 18:29 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt2 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 18:29 | Metin Belgesi | 1 KB |

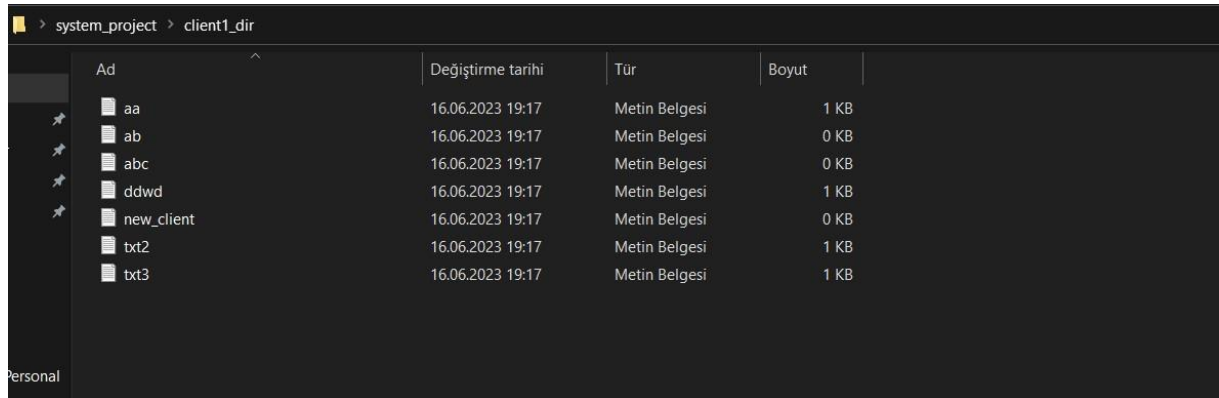
Personal

Client Operations:

Add:

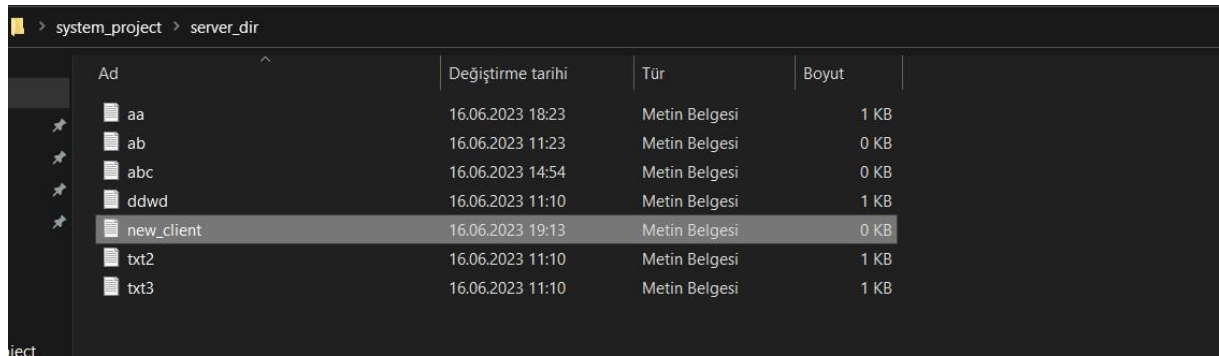
Add a file named new_client

Client1



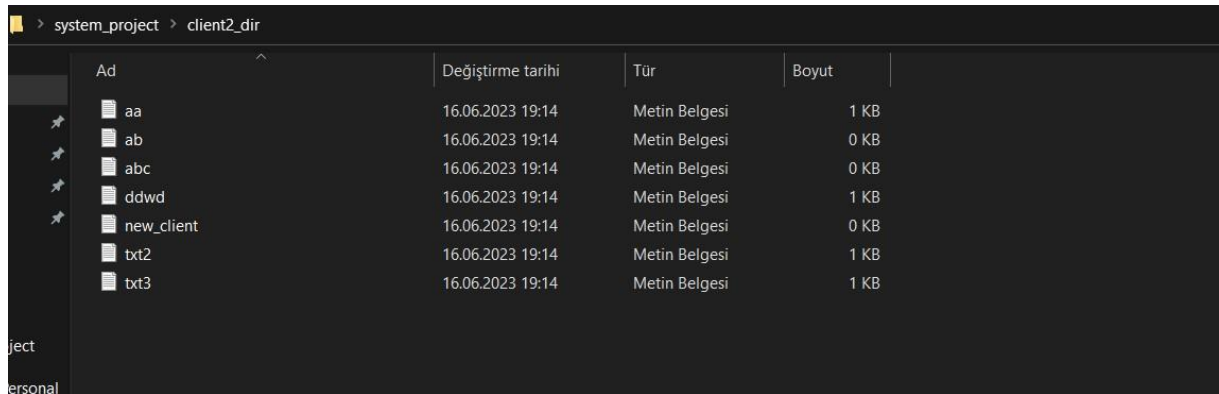
| Ad | Değiştirme tarihi | Tür | Boyut |
|------------|-------------------|---------------|-------|
| aa | 16.06.2023 19:17 | Metin Belgesi | 1 KB |
| ab | 16.06.2023 19:17 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 19:17 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 19:17 | Metin Belgesi | 1 KB |
| new_client | 16.06.2023 19:17 | Metin Belgesi | 0 KB |
| txt2 | 16.06.2023 19:17 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 19:17 | Metin Belgesi | 1 KB |

Server



| Ad | Değiştirme tarihi | Tür | Boyut |
|------------|-------------------|---------------|-------|
| aa | 16.06.2023 18:23 | Metin Belgesi | 1 KB |
| ab | 16.06.2023 11:23 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 14:54 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 11:10 | Metin Belgesi | 1 KB |
| new_client | 16.06.2023 19:13 | Metin Belgesi | 0 KB |
| txt2 | 16.06.2023 11:10 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 11:10 | Metin Belgesi | 1 KB |

Client2



| Ad | Değiştirme tarihi | Tür | Boyut |
|------------|-------------------|---------------|-------|
| aa | 16.06.2023 19:14 | Metin Belgesi | 1 KB |
| ab | 16.06.2023 19:14 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 19:14 | Metin Belgesi | 0 KB |
| ddwd | 16.06.2023 19:14 | Metin Belgesi | 1 KB |
| new_client | 16.06.2023 19:14 | Metin Belgesi | 0 KB |
| txt2 | 16.06.2023 19:14 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 19:14 | Metin Belgesi | 1 KB |

Change:

Remove:

Remove new_clien ü,ddwd from client1

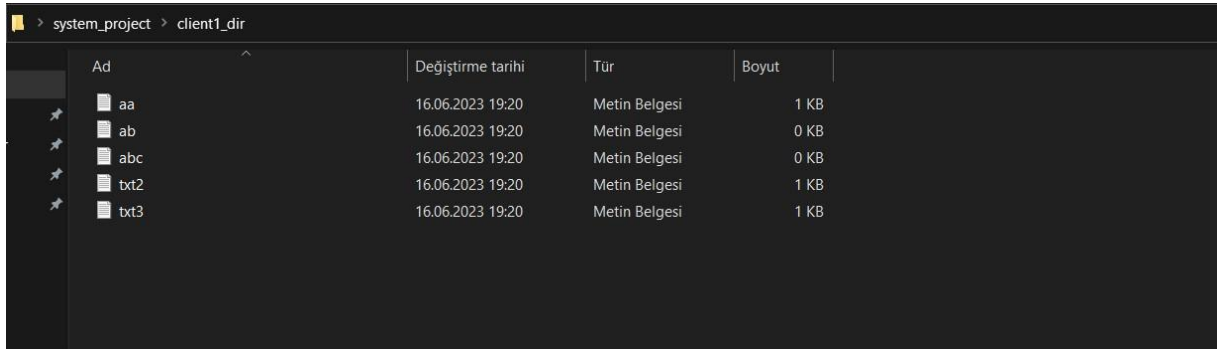
Server:

| > system_project > client2_dir | | | | | |
|--------------------------------|------|-------------------|---------------|-------|--|
| | Ad | Değiştirme tarihi | Tür | Boyut | |
| | aa | 16.06.2023 19:18 | Metin Belgesi | 1 KB | |
| | ab | 16.06.2023 19:18 | Metin Belgesi | 0 KB | |
| | abc | 16.06.2023 19:18 | Metin Belgesi | 0 KB | |
| | txt2 | 16.06.2023 19:18 | Metin Belgesi | 1 KB | |
| | txt3 | 16.06.2023 19:18 | Metin Belgesi | 1 KB | |

Client2

| > system_project > server_dir | | | | | |
|-------------------------------|------|-------------------|---------------|-------|--|
| | Ad | Değiştirme tarihi | Tür | Boyut | |
| | aa | 16.06.2023 18:23 | Metin Belgesi | 1 KB | |
| | ab | 16.06.2023 11:23 | Metin Belgesi | 0 KB | |
| | abc | 16.06.2023 14:54 | Metin Belgesi | 0 KB | |
| | txt2 | 16.06.2023 19:17 | Metin Belgesi | 1 KB | |
| | txt3 | 16.06.2023 19:18 | Metin Belgesi | 1 KB | |

Client1



| Ad | Değiştirme tarihi | Tür | Boyut |
|------|-------------------|---------------|-------|
| aa | 16.06.2023 19:20 | Metin Belgesi | 1 KB |
| ab | 16.06.2023 19:20 | Metin Belgesi | 0 KB |
| abc | 16.06.2023 19:20 | Metin Belgesi | 0 KB |
| txt2 | 16.06.2023 19:20 | Metin Belgesi | 1 KB |
| txt3 | 16.06.2023 19:20 | Metin Belgesi | 1 KB |

Note:

Running example

Server: `sudo ./BibakBoxServer server_dir [client_number]`
`[port number]`

Client: `./BibakBoxClient client_dir [port number] [IP adress]`