Assignment 4
Of
Network & Distributed System Lab (CS2051)
Masters of Technology in Computer Science And Engineering

submitted to
Dr Sujoy Saha
Assistant Professor
Dept. of CSE

National Institute of Technology, Durgapur

submitted by
Arghya Bandyopadhyay
RollNo. 20CS4103

1 Jul 2021

Objective: Sync is a communication protocol for peer-to-peer file sharing (P2P), enabling users to distribute data and electronic files over the Internet/offline in a decentralized manner.

Design and Implement a sync protocol using "nanohttpd" which will transfer multimodal files (Text, Image, Audio, and Video), and if the connection is intermittent during the synching, the process can restart the services and resume the cycle/process.

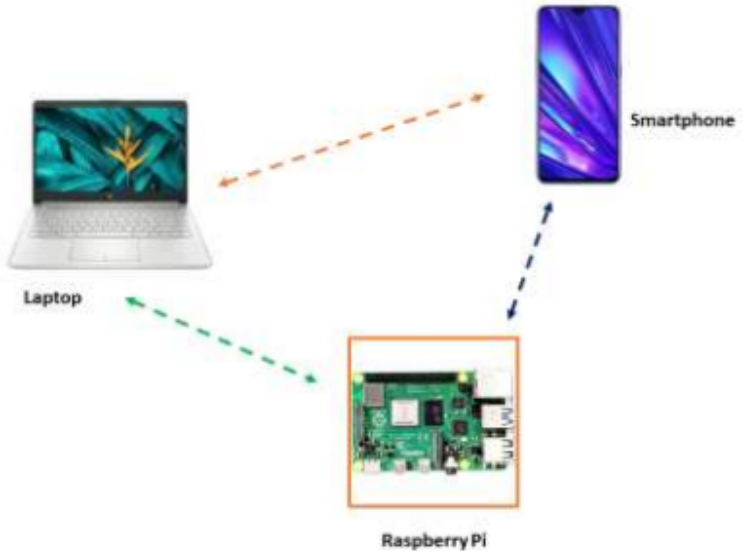


Figure 1: Information exchange between heterogeneous nodes

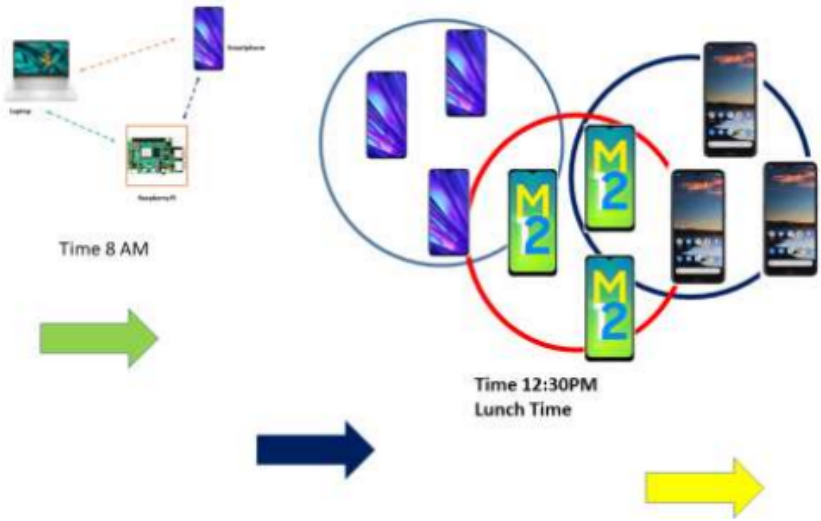


Figure 2: File synching (store and forwarding approach) between the nodes and file can move from one place another due to the movement of the nodes.

Answer.

build.gradle

```
plugins {
    id 'com.android.application'
}

android {
    compileSdkVersion 30
    buildToolsVersion "30.0.3"

    defaultConfig {
        applicationId "com.example.sync_protocol"
        minSdkVersion 16
        targetSdkVersion 30
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
}

dependencies {

    implementation 'androidx.appcompat:appcompat:1.3.0'
    implementation 'com.google.android.material:material:1.3.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.0.4'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.3'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'
    implementation 'com.android.support:support-annotations:28.0.0'
    implementation 'org.nanohttpd:nanohttpd:2.3.1'
    implementation group: 'javax.activation', name: 'activation', version: '1.1.1'
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/deviceIp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.328" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="56dp"
        android:text="Start Transfer"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/deviceIp" />

```

```
    <EditText
        android:id="@+id/ip_add"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="80dp"
        android:ems="10"
        android:hint="IP Address"
        android:inputType="textPersonName"
        app:layout_constraintBottom_toTopOf="@+id/deviceIp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.sync_protocol">

    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.MANAGE_EXTERNAL_STORAGE" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Sync_protocol"
        android:usesCleartextTraffic="true"
        android:requestLegacyExternalStorage="true">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

GetFileExtension.java

```
package com.example.sync_protocol;

import java.io.File;

public class GetFileExtension {

    public static String getFileExtension(File file) {
        String fileName = file.getName();
        if(fileName.lastIndexOf(".") != -1 && fileName.lastIndexOf(".") != 0)
            return fileName.substring(fileName.lastIndexOf(".")+1);
        else
            return "";
    }
}
```

Download.java

```
package com.example.sync_protocol;

import android.os.Build;
import android.widget.Toast;

import androidx.annotation.RequiresApi;

import java.io.*;
import java.net.HttpURLConnection;
import java.net.URL;

public class Download implements Runnable {
    String link;
```

```
    File out;

    public Download(String link, File out) {
        this.link = link;
        this.out = out;
    }

    @RequiresApi(api = Build.VERSION_CODES.N)
    @Override

    public void run() {
        try {
            URL url = new URL(link);
            HttpURLConnection http = (HttpURLConnection)url.openConnection();
            BufferedInputStream in = new BufferedInputStream(http.getInputStream());
            FileOutputStream fos;

            if(out.exists()) {
                fos = new FileOutputStream(out, true);
            } else {
                fos = new FileOutputStream(out);
            }

            in.skip(out.length());

            BufferedOutputStream bout = new BufferedOutputStream(fos, 1024);
            byte[] buffer = new byte[1024];
            double downloaded = 0.0;
            int read = 0;
            double percentDownloaded = 0.0;
            while((read = in.read(buffer, 0, 1024)) >= 0) {
                bout.write(buffer, 0, read);
                downloaded += read;
            }

            bout.close();
            in.close();
            System.out.println("Work Done");

        } catch(IOException ex) {
            ex.printStackTrace();
        }
    }
}
```

Ip.java

```
package com.example.sync_protocol;

import java.net.Inet4Address;
import java.net.InetAddress;
import java.net.NetworkInterface;
import java.net.SocketException;

import java.util.Enumeration;

public class Ip {
    public static String ipadd() {
        try {
            for(Enumeration<NetworkInterface> en = NetworkInterface.getNetworkInterfaces(); en.
    hasMoreElements();) {
                NetworkInterface intf = en.nextElement();
                for(Enumeration<InetAddress> enumIpAddr = intf.getInetAddresses(); enumIpAddr.
    hasMoreElements();) {
                    InetAddress inetAddress = enumIpAddr.nextElement();
                    if(!inetAddress.isLoopbackAddress() && inetAddress instanceof Inet4Address) {
                        return inetAddress.getHostAddress();
                    }
                }
            }
        } catch(SocketException ex) {
            ex.printStackTrace();
        }
        return null;
    }
}
```

Main.java

```java
package com.example.sync_protocol;

import android.Manifest;
import android.content.pm.PackageManager;
import android.os.Build;
import android.os.Environment;
import android.util.Log;

import androidx.core.content.ContextCompat;

import java.io.BufferedReader;
import java.io.File;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.URL;
import java.util.*;

public class Main {
    public static void invoke(String ip) throws IOException {
        //String ip = "10.0.2.16";
        String urlString = "http://"+ip+":8080";

        //Collection A = new ArrayList();
        //Collection B = new ArrayList();

        ArrayList<String> A = new ArrayList();
        ArrayList<String> B = new ArrayList();

        String fileName = "";
        String fileURL;
        String saveDir;
        String name = Environment.getExternalStorageDirectory().getAbsolutePath() + "/test/";
        File folder = new File(name);
        System.out.println(name);

        for(File file : Objects.requireNonNull(folder.listFiles())) {
            B.add(file.getName());
        }

        System.out.println("Files in My System\n"+B);

        URL url = new URL(urlString);
        Log.println(Log.INFO, String.valueOf(Log.INFO), String.valueOf(url) );

        BufferedReader reader = new BufferedReader(new InputStreamReader(url.openStream()));
        Log.println(Log.INFO, String.valueOf(Log.INFO), String.valueOf(reader) );

        String line;
        line = reader.readLine();
        Log.println(Log.INFO, String.valueOf(Log.INFO), String.valueOf(line) );

        int i;
        for(i = 0; i < line.length(); i++) {
            if(line.charAt(i) =='e' && line.charAt(i+1) == '=') {
                i = i + 2;
                while(line.charAt(i) != '"') {
                    fileName += line.charAt(i);
                    i++;
                }
                A.add(fileName);
                fileName = "";
            }
        }
        System.out.println("Files in Server\n"+A);

        ArrayList<String> diff = new ArrayList(A);
        diff.removeAll(B);
        System.out.println("File To Be Taken from Server\n"+diff);
        Iterator it = diff.iterator();
        while(it.hasNext()) {
            String x = (String)it.next();
            fileURL = "http://"+ip+":8080/get?name="+x;
            saveDir = Environment.getExternalStorageDirectory().getAbsolutePath()+"/test/"+x;
            File out = new File(saveDir);
            new Thread(new Download(fileURL, out)).start();
        }
        reader.close();
    }
}
```

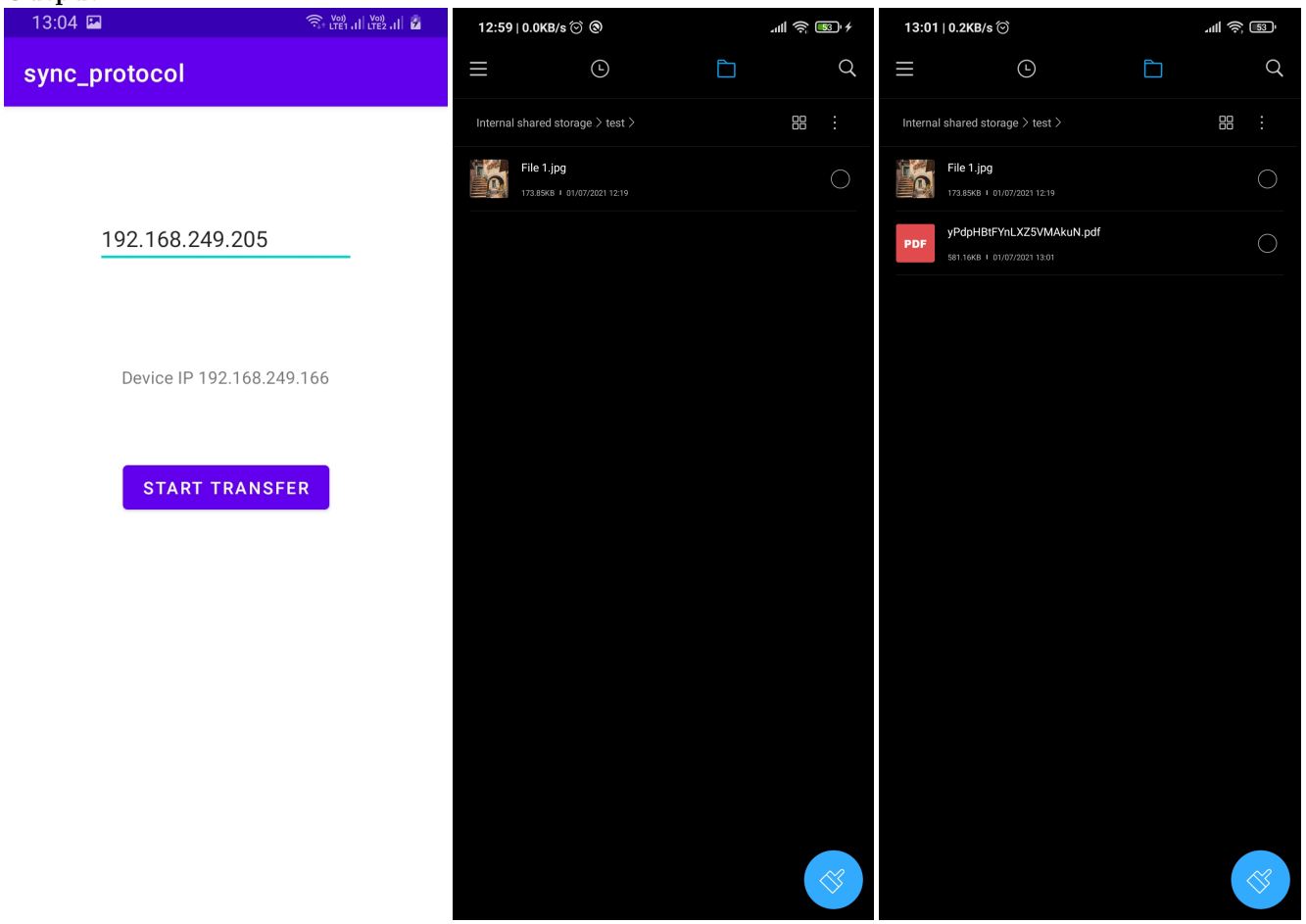
MainActivity.java

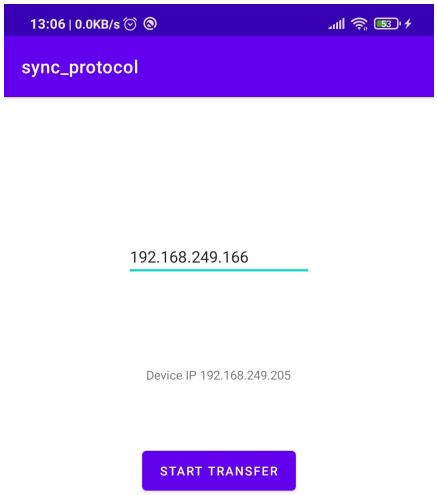
```java
package com.example.sync_protocol;

import androidx.appcompat.app.AppCompatActivity;
import androidx.core.content.ContextCompat;

import android.Manifest;
import android.content.pm.PackageManager;
import android.os.Build;
import android.os.Bundle;
import android.os.Environment;
import android.os.StrictMode;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.net.InetAddress;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.io.FileInputStream;

import javax.activation.MimetypesFileTypeMap;

import fi.iki.elonen.NanoHTTPD;

import static com.example.sync_protocol.Ip.ipadd;

public class MainActivity extends AppCompatActivity {
    private WebServer server;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        String i = "";
        i = ipadd();
        TextView deviceIp = findViewById(R.id.deviceIp);
        TextView serv_ip = findViewById(R.id.ip_add);
        deviceIp.setText("Device IP " + i);

        final Button button;
        button = findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                try {
                    StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll
().build();
                    StrictMode.setThreadPolicy(policy);
                    System.out.println(InetAddress.getLocalHost().getHostAddress());
                    server = new WebServer();
                    try {
                        server.start();
                    } catch(IOException ioe) {
                        Log.w("Httpd", "The Server could not start.");
                    }
                    Log.w("Httpd", "Web Server Initialized");
                    Main.invoke(serv_ip.getText().toString());
                    Toast.makeText(MainActivity.this,"Transfer Compelete", Toast.LENGTH_LONG);
                } catch (IOException e1) {
                    e1.printStackTrace();
                }
            }
        });
    }

    @Override
    public void onDestroy() {
        super.onDestroy();
        if(server != null) {
            server.stop();
        }
```

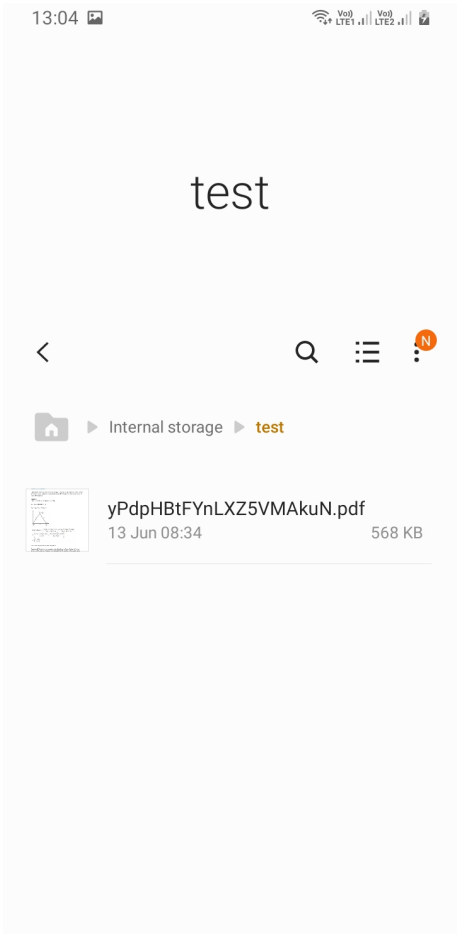
```
    }

    public class WebServer extends NanoHTTPD {
        public WebServer() {
            super(8080);
        }

        @Override
        public Response serve(String uri, Method method, Map<String, String> header,
                              Map<String, String> parameters, Map<String, String> files) {

            File folder = new File(Environment.getExternalStorageDirectory().getAbsolutePath() + "/test/");

            Map<Integer, List<String>> prio = new HashMap<>();
            List<String> list1 = new ArrayList<>();
            List<String> list2 = new ArrayList<>();
            List<String> list3 = new ArrayList<>();
            List<String> list4 = new ArrayList<>();
            List<String> list5 = new ArrayList<>();

            for(File file: folder.listFiles()) {
                if(GetFileExtension.getFileExtension(file).equals("txt")) {
                    list1.add(file.getName());
                    prio.put(new Integer(1), list1);
                }
                if(GetFileExtension.getFileExtension(file).equals("pdf")) {
                    list2.add(file.getName());
                    prio.put(new Integer(2), list2);
                }
                if(GetFileExtension.getFileExtension(file).equals("jpg") || GetFileExtension.getFileExtension(file).equals("png")) {
                    list3.add(file.getName());
                    prio.put(new Integer(3), list3);
                }
                if(GetFileExtension.getFileExtension(file).equals("mp3")) {
                    list4.add(file.getName());
                    prio.put(new Integer(4), list4);
                }
                if(GetFileExtension.getFileExtension(file).equals("mp4")) {
                    list5.add(file.getName());
                    prio.put(new Integer(5), list5);
                }
            }

            String fileName = "";

            if(uri.equals("/")) {
                System.out.println(uri);

                String st = "";
                String x = "";

                for(Map.Entry<Integer, List<String>> en : prio.entrySet()) {
                    for(String obj : en.getValue()) {
                        x = obj;
                        st = st + "<a href=\"/get?name=" + x + "\">" + x + "</a>";
                        st = st + "<br>";
                    }
                }

                return newFixedLengthResponse(Response.Status.OK, MIME_HTML, st);
            } else if(uri.equals("/get")) {
                FileInputStream fis = null;
                File f = null;
                try {
                    f = new File(Environment.getExternalStorageDirectory().getAbsolutePath() + "/test/" + parameters.get("name"));
                    System.out.println("GET : " + f);
                    fis = new FileInputStream(f);
                    System.out.println("GET FIS : " + fis);
                } catch(FileNotFoundException e) {
                    e.printStackTrace();
                }

                MimetypesFileTypeMap mimeTypesMap = new MimetypesFileTypeMap();

                String mimeType = mimeTypesMap.getContentType(fileName);
                return newChunkedResponse(Response.Status.OK,mimeType, fis);
```

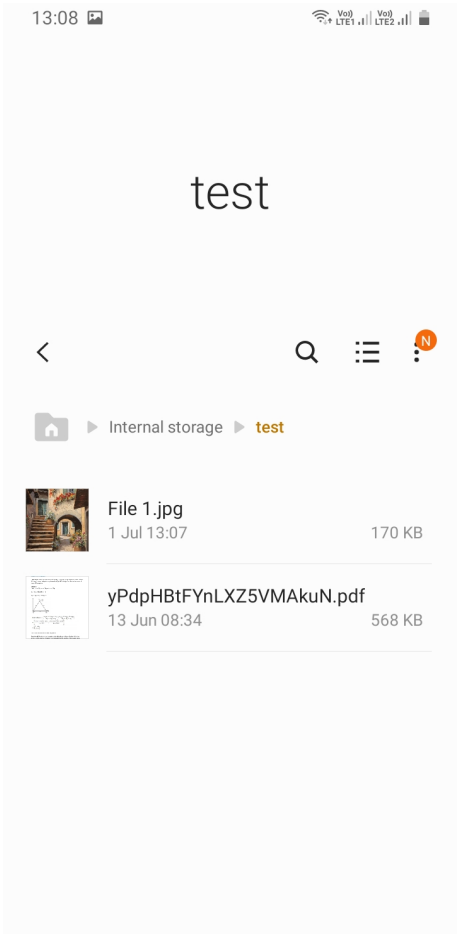
```
                } else {
                    return newFixedLengthResponse("404 File Not Found");
                }
            }
        }
}
```

Output:




13:06 | 0.0KB/s
sync_protocol
192.168.249.166
Device IP 192.168.249.205
START TRANSFER


13:04
test
Internal storage
test
yPdpHBtFYnLXZ5VMAkuN.pdf
13 Jun 08:34
568 KB


13:08
test
Internal storage
test
File 1.jpg
1 Jul 13:07
170 KB
yPdpHBtFYnLXZ5VMAkuN.pdf
13 Jun 08:34
568 KB