

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 syncing, the process can restart the services and resume the cycle/process.

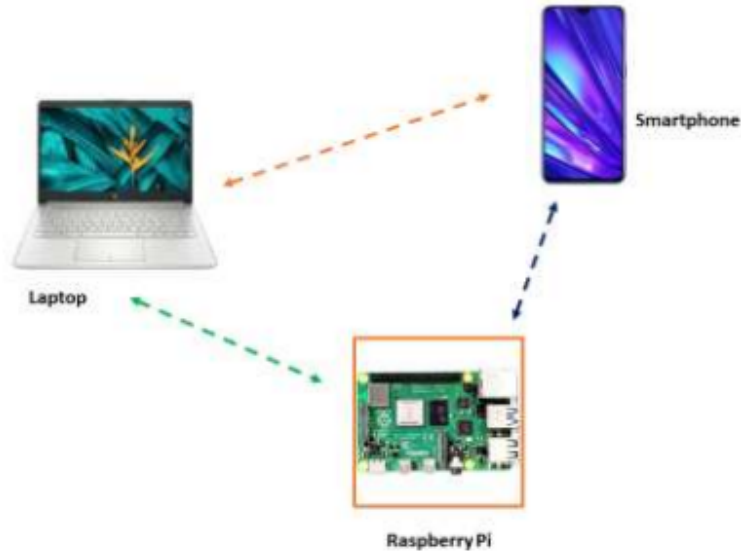


Figure 1: Information exchange between heterogeneous nodes

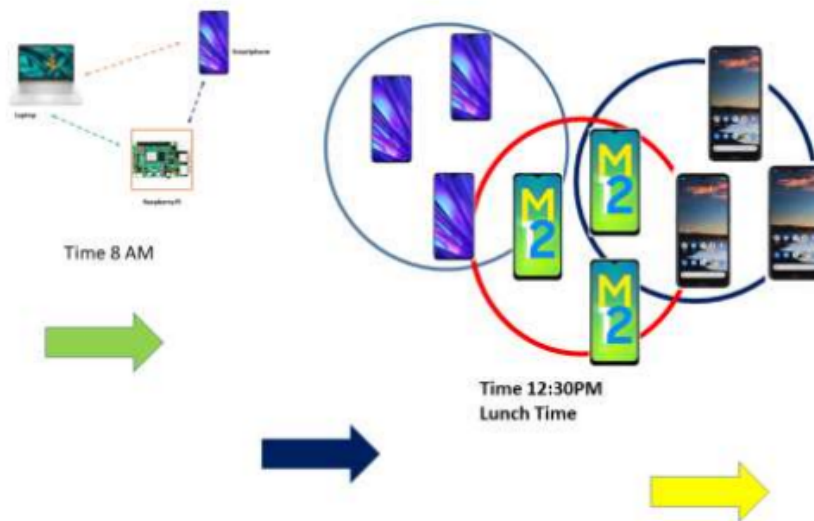


Figure 2: File syncing (store and forwarding approach) between the nodes and file can move from one place another due to the movement of the nodes.

Answer.

build.gradle

```
1 plugins {
2     id 'com.android.application'
3 }
4
5 android {
6     compileSdkVersion 30
7     buildToolsVersion "30.0.3"
8
9     defaultConfig {
10         applicationId "com.example.sync_protocol"
11         minSdkVersion 16
12         targetSdkVersion 30
13         versionCode 1
14         versionName "1.0"
15
16         testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
17     }
18
19     buildTypes {
20         release {
21             minifyEnabled false
22             proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.
23             pro'
24         }
25     }
26     compileOptions {
27         sourceCompatibility JavaVersion.VERSION_1_8
28         targetCompatibility JavaVersion.VERSION_1_8
29     }
30 }
31
32 dependencies {
33     implementation 'androidx.appcompat:appcompat:1.3.0'
34     implementation 'com.google.android.material:material:1.3.0'
35     implementation 'androidx.constraintlayout:constraintlayout:2.0.4'
36     testImplementation 'junit:junit:4.13.2'
37     androidTestImplementation 'androidx.test.ext:junit:1.1.3'
38     androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'
39     implementation 'com.android.support:support-annotations:28.0.0'
40     implementation 'org.nanohttpd:nanohttpd:2.3.1'
41     implementation group: 'javax.activation', name: 'activation', version: '1.1.1'
42 }
```

activity_main.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res
3 /android"
4     xmlns:app="http://schemas.android.com/apk/res-auto"
5     xmlns:tools="http://schemas.android.com/tools"
6     android:layout_width="match_parent"
7     android:layout_height="match_parent"
8     tools:context=".MainActivity">
9
10     <TextView
11         android:id="@+id/deviceIp"
12         android:layout_width="wrap_content"
13         android:layout_height="wrap_content"
14         android:text="Hello World!"
15         app:layout_constraintBottom_toBottomOf="parent"
16         app:layout_constraintHorizontal_bias="0.498"
17         app:layout_constraintLeft_toLeftOf="parent"
18         app:layout_constraintRight_toRightOf="parent"
19         app:layout_constraintTop_toTopOf="parent"
20         app:layout_constraintVertical_bias="0.328" />
21
22     <Button
23         android:id="@+id/button"
24         android:layout_width="wrap_content"
25         android:layout_height="wrap_content"
26         android:layout_marginTop="56dp"
27         android:text="Start Transfer"
28         app:layout_constraintEnd_toEndOf="parent"
29         app:layout_constraintStart_toStartOf="parent"
30         app:layout_constraintTop_toBottomOf="@+id/deviceIp" />
```

```

31 <EditText
32     android:id="@+id/ip_add"
33     android:layout_width="wrap_content"
34     android:layout_height="wrap_content"
35     android:layout_marginBottom="80dp"
36     android:ems="10"
37     android:hint="IP Address"
38     android:inputType="textPersonName"
39     app:layout_constraintBottom_toTopOf="@+id/deviceIp"
40     app:layout_constraintEnd_toEndOf="parent"
41     app:layout_constraintStart_toStartOf="parent" />
42
43 </androidx.constraintlayout.widget.ConstraintLayout>

```

AndroidManifest.xml

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3     package="com.example.sync_protocol">
4
5     <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
6     <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
7     <uses-permission android:name="android.permission.INTERNET" />
8     <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
9     <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
10    <uses-permission android:name="android.permission.MANAGE_EXTERNAL_STORAGE" />
11
12    <application
13        android:allowBackup="true"
14        android:icon="@mipmap/ic_launcher"
15        android:label="@string/app_name"
16        android:roundIcon="@mipmap/ic_launcher_round"
17        android:supportRtl="true"
18        android:theme="@style/Theme.Sync_protocol"
19        android:usesCleartextTraffic="true"
20        android:requestLegacyExternalStorage="true">
21        <activity android:name=".MainActivity">
22            <intent-filter>
23                <action android:name="android.intent.action.MAIN" />
24
25                <category android:name="android.intent.category.LAUNCHER" />
26            </intent-filter>
27        </activity>
28    </application>
29
30 </manifest>

```

GetFileExtension.java

```

1 package com.example.sync_protocol;
2
3 import java.io.File;
4
5 public class GetFileExtension {
6
7     public static String getFileExtension(File file) {
8         String fileName = file.getName();
9         if(fileName.lastIndexOf(".") != -1 && fileName.lastIndexOf(".") != 0)
10             return fileName.substring(fileName.lastIndexOf(".")+1);
11         else
12             return "";
13     }
14 }

```

Download.java

```

1 package com.example.sync_protocol;
2
3 import android.os.Build;
4 import android.widget.Toast;
5
6 import androidx.annotation.RequiresApi;
7
8 import java.io.*;
9 import java.net.HttpURLConnection;
10 import java.net.URL;
11
12 public class Download implements Runnable {
13     String link;

```

```

14     File out;
15
16     public Download(String link, File out) {
17         this.link = link;
18         this.out = out;
19     }
20
21     @RequiresApi(api = Build.VERSION_CODES.N)
22     @Override
23
24     public void run() {
25         try {
26             URL url = new URL(link);
27             HttpURLConnection http = (HttpURLConnection)url.openConnection();
28             BufferedInputStream in = new BufferedInputStream(http.getInputStream());
29             FileOutputStream fos;
30
31             if(out.exists()) {
32                 fos = new FileOutputStream(out, true);
33             } else {
34                 fos = new FileOutputStream(out);
35             }
36
37             in.skip(out.length());
38
39             BufferedOutputStream bout = new BufferedOutputStream(fos, 1024);
40             byte[] buffer = new byte[1024];
41             double downloaded = 0.0;
42             int read = 0;
43             double percentDownloaded = 0.0;
44             while((read = in.read(buffer, 0, 1024)) >= 0) {
45                 bout.write(buffer, 0, read);
46                 downloaded += read;
47             }
48
49             bout.close();
50             in.close();
51             System.out.println("Work Done");
52
53             } catch(IOException ex) {
54                 ex.printStackTrace();
55             }
56         }
57     }

```

Ip.java

```

1  package com.example.sync_protocol;
2
3  import java.net.Inet4Address;
4  import java.net.InetAddress;
5  import java.net.NetworkInterface;
6  import java.net.SocketException;
7
8  import java.util.Enumeration;
9
10 public class Ip {
11     public static String ipadd() {
12         try {
13             for(Enumeration<NetworkInterface> en = NetworkInterface.getNetworkInterfaces(); en.
hasMoreElements();) {
14                 NetworkInterface intf = en.nextElement();
15                 for(Enumeration<InetAddress> enumIpAddr = intf.getInetAddresses(); enumIpAddr.
hasMoreElements();) {
16                     InetAddress inetAddress = enumIpAddr.nextElement();
17                     if(!inetAddress.isLoopbackAddress() && inetAddress instanceof Inet4Address) {
18                         return inetAddress.getHostAddress();
19                     }
20                 }
21             }
22         } catch(SocketException ex) {
23             ex.printStackTrace();
24         }
25         return null;
26     }
27 }

```

Main.java

```

1 package com.example.sync_protocol;
2
3 import android.Manifest;
4 import android.content.pm.PackageManager;
5 import android.os.Build;
6 import android.os.Environment;
7 import android.util.Log;
8
9 import androidx.core.content.ContextCompat;
10
11 import java.io.BufferedReader;
12 import java.io.File;
13 import java.io.IOException;
14 import java.io.InputStreamReader;
15 import java.net.URL;
16 import java.util.*;
17
18 public class Main {
19     public static void invoke(String ip) throws IOException {
20         //String ip = "10.0.2.16";
21         String urlString = "http://" + ip + ":8080";
22
23         //Collection A = new ArrayList();
24         //Collection B = new ArrayList();
25
26         ArrayList<String> A = new ArrayList();
27         ArrayList<String> B = new ArrayList();
28
29         String fileName = "";
30         String fileURL;
31         String saveDir;
32         String name = Environment.getExternalStorageDirectory().getAbsolutePath() + "/test/";
33         File folder = new File(name);
34         System.out.println(name);
35
36         for(File file : Objects.requireNonNull(folder.listFiles())) {
37             B.add(file.getName());
38         }
39
40         System.out.println("Files in My System\n" + B);
41
42         URL url = new URL(urlString);
43         Log.println(Log.INFO, String.valueOf(Log.INFO), String.valueOf(url));
44
45         BufferedReader reader = new BufferedReader(new InputStreamReader(url.openStream()));
46         Log.println(Log.INFO, String.valueOf(Log.INFO), String.valueOf(reader));
47
48         String line;
49         line = reader.readLine();
50         Log.println(Log.INFO, String.valueOf(Log.INFO), String.valueOf(line));
51
52         int i;
53         for(i = 0; i < line.length(); i++) {
54             if(line.charAt(i) == 'e' && line.charAt(i+1) == '=') {
55                 i = i + 2;
56                 while(line.charAt(i) != '\n') {
57                     fileName += line.charAt(i);
58                     i++;
59                 }
60                 A.add(fileName);
61                 fileName = "";
62             }
63         }
64         System.out.println("Files in Server\n" + A);
65
66         ArrayList<String> diff = new ArrayList(A);
67         diff.removeAll(B);
68         System.out.println("File To Be Taken from Server\n" + diff);
69         Iterator it = diff.iterator();
70         while(it.hasNext()) {
71             String x = (String)it.next();
72             fileURL = "http://" + ip + ":8080/get?name=" + x;
73             saveDir = Environment.getExternalStorageDirectory().getAbsolutePath() + "/test/" + x;
74             File out = new File(saveDir);
75             new Thread(new Download(fileURL, out)).start();
76         }
77         reader.close();
78     }
79 }

```

MainActivity.java

```
1 package com.example.sync_protocol;
2
3 import androidx.appcompat.app.AppCompatActivity;
4 import androidx.core.content.ContextCompat;
5
6 import android.Manifest;
7 import android.content.pm.PackageManager;
8 import android.os.Build;
9 import android.os.Bundle;
10 import android.os.Environment;
11 import android.os.StrictMode;
12 import android.util.Log;
13 import android.view.View;
14 import android.widget.Button;
15 import android.widget.TextView;
16 import android.widget.Toast;
17
18 import java.io.File;
19 import java.io.FileNotFoundException;
20 import java.io.IOException;
21 import java.net.InetAddress;
22 import java.util.ArrayList;
23 import java.util.HashMap;
24 import java.util.List;
25 import java.util.Map;
26 import java.io.FileInputStream;
27
28 import javax.activation.MimetypesFileTypeMap;
29
30 import fi.iki.elonen.NanoHTTPD;
31
32 import static com.example.sync_protocol.Ip.ipadd;
33
34 public class MainActivity extends AppCompatActivity {
35     private WebServer server;
36
37     @Override
38     protected void onCreate(Bundle savedInstanceState) {
39         super.onCreate(savedInstanceState);
40         setContentView(R.layout.activity_main);
41         String i = "";
42         i = ipadd();
43         TextView deviceIp = findViewById(R.id.deviceIp);
44         TextView serv_ip = findViewById(R.id.ip_add);
45         deviceIp.setText("Device IP " + i);
46
47         final Button button;
48         button = findViewById(R.id.button);
49         button.setOnClickListener(new View.OnClickListener() {
50             @Override
51             public void onClick(View v) {
52                 try {
53                     StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll
54                     ().build();
55                     StrictMode.setThreadPolicy(policy);
56                     System.out.println(InetAddress.getLocalHost().getHostAddress());
57                     server = new WebServer();
58                     try {
59                         server.start();
60                     } catch (IOException ioe) {
61                         Log.w("Httpd", "The Server could not start.");
62                     }
63                     Log.w("Httpd", "Web Server Initialized");
64                     Main.invoke(serv_ip.getText().toString());
65                     Toast.makeText(MainActivity.this, "Transfer Complete", Toast.LENGTH_LONG);
66                 } catch (IOException e1) {
67                     e1.printStackTrace();
68                 }
69             }
70         });
71
72         @Override
73         public void onDestroy() {
74             super.onDestroy();
75             if (server != null) {
76                 server.stop();
77             }
78         }
79     }
80 }
```

```

78 }
79
80 public class WebServer extends NanoHTTPD {
81     public WebServer() {
82         super(8080);
83     }
84
85     @Override
86     public Response serve(String uri, Method method, Map<String, String> header,
87         Map<String, String> parameters, Map<String, String> files) {
88
89         File folder = new File(Environment.getExternalStorageDirectory().getAbsolutePath() + "/"
90 test/");
91
92         Map<Integer, List<String>> prio = new HashMap<>();
93         List<String> list1 = new ArrayList<>();
94         List<String> list2 = new ArrayList<>();
95         List<String> list3 = new ArrayList<>();
96         List<String> list4 = new ArrayList<>();
97         List<String> list5 = new ArrayList<>();
98
99         for(File file: folder.listFiles()) {
100             if(GetFileExtension.getFileExtension(file).equals("txt")) {
101                 list1.add(file.getName());
102                 prio.put(new Integer(1), list1);
103             }
104             if(GetFileExtension.getFileExtension(file).equals("pdf")) {
105                 list2.add(file.getName());
106                 prio.put(new Integer(2), list2);
107             }
108             if(GetFileExtension.getFileExtension(file).equals("jpg") || GetFileExtension.
109 getFileExtension(file).equals("png")) {
110                 list3.add(file.getName());
111                 prio.put(new Integer(3), list3);
112             }
113             if(GetFileExtension.getFileExtension(file).equals("mp3")) {
114                 list4.add(file.getName());
115                 prio.put(new Integer(4), list4);
116             }
117             if(GetFileExtension.getFileExtension(file).equals("mp4")) {
118                 list5.add(file.getName());
119                 prio.put(new Integer(5), list5);
120             }
121         }
122
123         String fileName = "";
124
125         if(uri.equals("/")) {
126             System.out.println(uri);
127
128             String st = "";
129             String x = "";
130
131             for(Map.Entry<Integer, List<String>> en : prio.entrySet()) {
132                 for(String obj : en.getValue()) {
133                     x = obj;
134                     st = st + "<a href=\"/" + get?name=" + x + "\">" + x + "</a>";
135                     st = st + "<br>";
136                 }
137             }
138
139             return newFixedLengthResponse(Response.Status.OK, MIME_HTML, st);
140         } else if(uri.equals("/get")) {
141             FileInputStream fis = null;
142             File f = null;
143             try {
144                 f = new File(Environment.getExternalStorageDirectory().getAbsolutePath() + "/"
145 test/" + parameters.get("name"));
146                 System.out.println("GET : " + f);
147                 fis = new FileInputStream(f);
148                 System.out.println("GET FIS : " + fis);
149             } catch(FileNotFoundException e) {
150                 e.printStackTrace();
151             }
152
153             MimetypeesFileTypeMap mimeTypeMap = new MimetypeesFileTypeMap();
154
155             String mimeType = mimeTypeMap.getContentType(fileName);
156             return newChunkedResponse(Response.Status.OK, mimeType, fis);

```

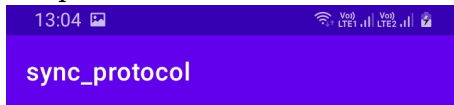


```

154         } else {
155             return newFixedLengthResponse("404 File Not Found");
156         }
157     }
158 }
159 }

```

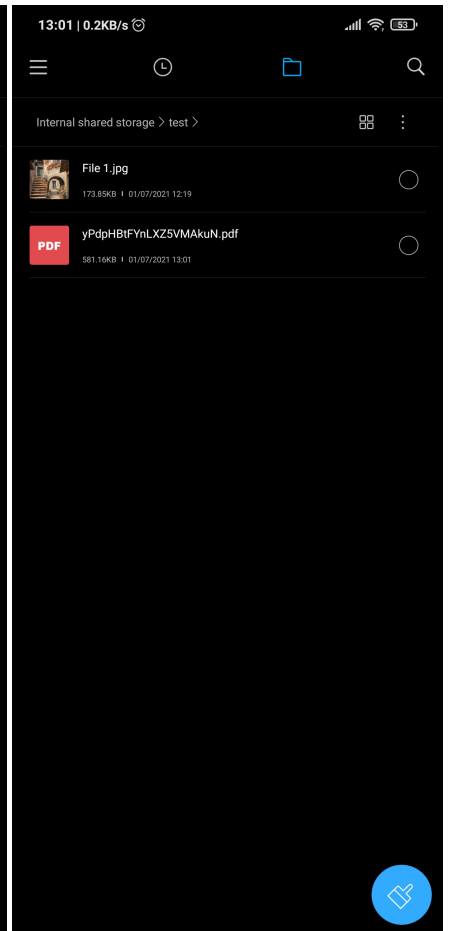
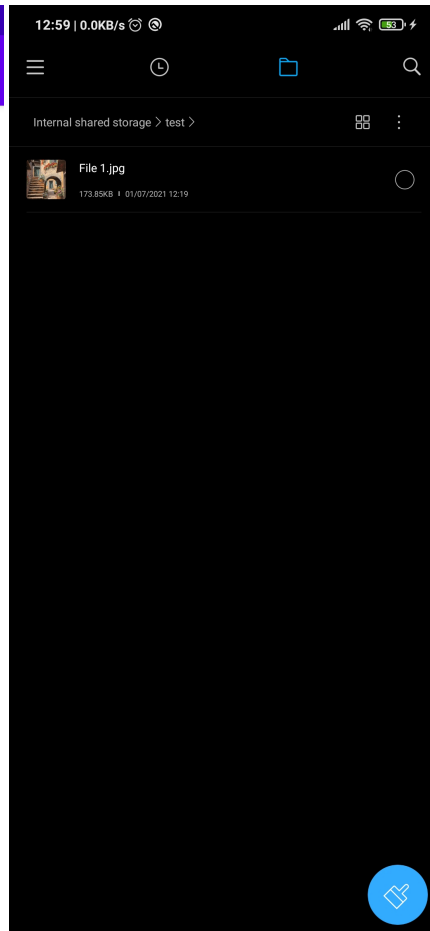
Output:



192.168.249.205

Device IP 192.168.249.166

START TRANSFER



13:06 | 0.0KB/s

sync_protocol

192.168.249.166

Device IP 192.168.249.205

START TRANSFER

13:04

test

<

Q

:

:

N

Internal storage

test

yPdpHBtFYnLXZ5VMAkuN.pdf

13 Jun 08:34

568 KB

13:08

test

<

Q

:

:

N

Internal storage

test

File 1.jpg

1 Jul 13:07

170 KB

yPdpHBtFYnLXZ5VMAkuN.pdf

13 Jun 08:34

568 KB