Work-let Name: Performance evaluation of DSCP values in various network

Worklet Details

1. Worklet ID: CP214MS
2. College Name: RAMAIAH INSTITUTE OF TECHOLOGY

KPIs achieved till now

Build an android application to setup the DSCP values

Any Challenges/ Issues faced

Facing slight issues with the API's

Next Steps

To finalize the build of the application and analize it over different service providers

Key Achievements/ Outcome till now

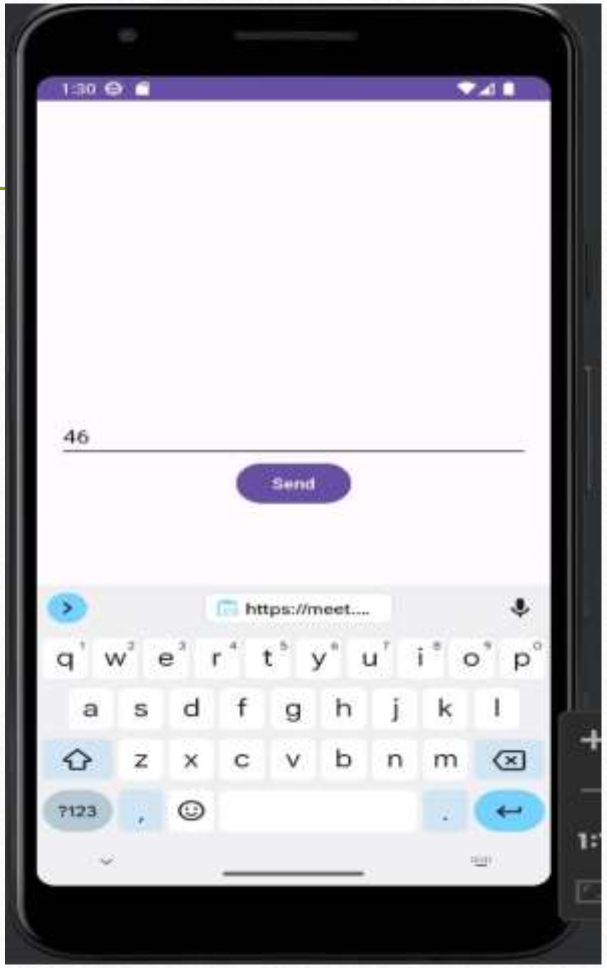
- Learned how to use android studio
- Created server using Java
- Able to setup the client server application which is also build on Java

Date: 20/06/23

Application setup in Android Studio

How it works

- Interface asks for the DSCP values which can be manually entered by the user
- Then there is a send option to send the DSCP values entered by the user to the server
- In the backend we have implemented auto generation of a 2mb file which is delivered according to the DSCP values entered by the user



Client side

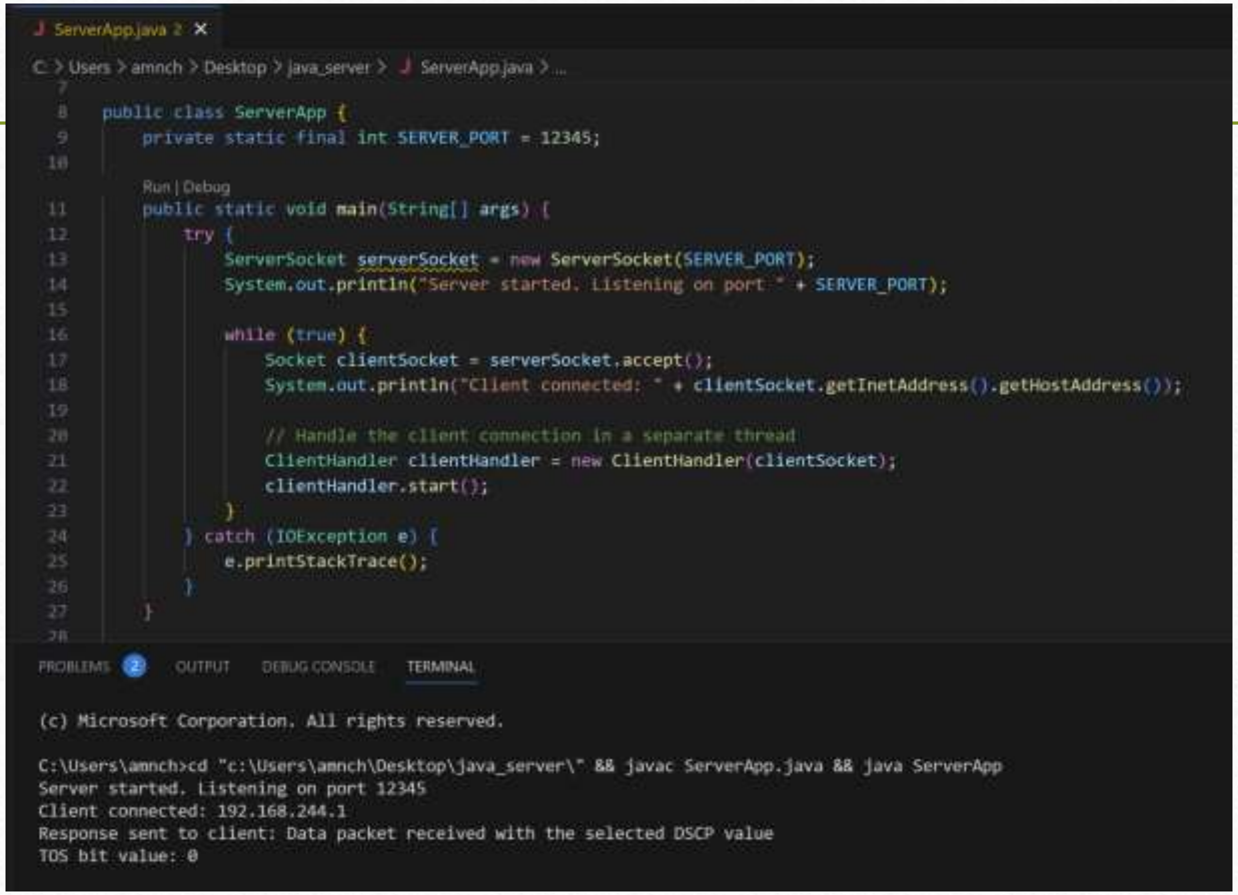
```
package com.example.prims_june;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.Socket;

public class MainActivity extends AppCompatActivity {

    private EditText dscpEditText;
    private Button sendButton;
    private TextView responseTextView;
```

```
@Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        dscpEditText = findViewById(R.id.dscpEditText);
        sendButton = findViewById(R.id.sendButton);
        responseTextView = findViewById(R.id.responseTextView);

        sendButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String dscpValue = dscpEditText.getText().toString();
                int dscp = Integer.parseInt(dscpValue);

                SendDataTask task = new SendDataTask(dscp);
                task.execute();
            }
        });
    }

    private class SendDataTask extends AsyncTask<Void, Void, String> {
        private int dscp;

        public SendDataTask(int dscp) {
            this.dscp = dscp;
        }
```

```java
@Override
    protected String doInBackground(Void... voids) {
        Socket socket = null;
        try {
            socket = new Socket("192.168.244.1", 12345);

            // Set DSCP value for the socket
            socket.setTrafficClass(dscp);

            // Send the data packet
            DataOutputStream outputStream = new DataOutputStream(socket.getOutputStream());
            byte[] dataPacket = generateDataPacket();
            outputStream.write(dataPacket);
            outputStream.flush();

            // Receive the response from the server
            DataInputStream inputStream = new DataInputStream(socket.getInputStream());
            String response = inputStream.readUTF();

            inputStream.close();
            outputStream.close();

            return response;
        }
```

```
catch (IOException e) {
                e.printStackTrace();
                return "Error: " + e.getMessage();
            } finally {
                if (socket != null) {
                    try {
                        socket.close();
                    } catch (IOException e) {
                        e.printStackTrace();
                    }
                }
            }
        }
@Override
        protected void onPostExecute(String response) {
            responseTextView.setText(response);
        }

        private byte[] generateDataPacket() {
            // Generate a 2MB data packet
            // Replace this with your own logic to generate the data packet
            byte[] dataPacket = new byte[2 * 1024 * 1024];
            // Fill the data packet with your data

            return dataPacket;
        }
    }
}
```

Output for the Server site



```java
public class ServerApp {
    private static final int SERVER_PORT = 12345;

    public static void main(String[] args) {
        try {
            ServerSocket serverSocket = new ServerSocket(SERVER_PORT);
            System.out.println("Server started. Listening on port " + SERVER_PORT);

            while (true) {
                Socket clientSocket = serverSocket.accept();
                System.out.println("Client connected: " + clientSocket.getInetAddress().getHostAddress());

                // Handle the client connection in a separate thread
                ClientHandler clientHandler = new ClientHandler(clientSocket);
                clientHandler.start();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```

```
(c) Microsoft Corporation. All rights reserved.

C:\Users\amnch>cd "c:\Users\amnch\Desktop\java_server\" && javac ServerApp.java && java ServerApp
Server started. Listening on port 12345
Client connected: 192.168.244.1
Response sent to client: Data packet received with the selected DSCP value
TOS bit value: 0
```