```
// main class
package com.example.salemalzubaidi.iotwm;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Switch;
import android.widget.TextView;
import android.widget.Toast;
import org.eclipse.paho.android.service.MqttAndroidClient;
import org.eclipse.paho.client.mqttv3.IMqttActionListener;
import org.eclipse.paho.client.mqttv3.IMqttToken;
import org.eclipse.paho.client.mqttv3.IMqttDeliveryToken;
import org.eclipse.paho.client.mqttv3.MqttCallbackExtended;
import org.eclipse.paho.client.mqttv3.MqttException;
import org.eclipse.paho.client.mqttv3.MqttMessage;
import java.util.Locale;
import helpers.MqttHelper;
public class MainActivity extends AppCompatActivity {
static int co=0; int flag=0; int s=0;
MqttAndroidClient mqttandroidclient;
MqttHelper mqttHelper;
TextView dataReceived,sliceOne,sliceTwo,overall,cons1,cons2;
Switch pumpSw;
int nc=0;int total1=0;double te1=0; double te2=0;double overall_total=0;double tot =0;int total2=0;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
pumpSw=(Switch)findViewById(R.id.sw);
dataReceived = (TextView) findViewById(R.id.dataReceived);
sliceOne = (TextView) findViewById(R.id.sl1);sliceOne.setText(String.format(Locale.ENGLISH,"%.2f",te1));
sliceTwo = (TextView) findViewById(R.id.sl2); sliceTwo.setText(String.format(Locale.ENGLISH,"%.2f",te2));
overall= (TextView) findViewById(R.id.over); overall.setText(String.format(Locale.ENGLISH,"%.2f",overall_total));
cons1= (TextView) findViewById(R.id.consume1);cons1.setText(String.valueOf(total1));
cons2= (TextView) findViewById(R.id.consume2);cons2.setText(String.valueOf(total2));
}
protected void con (View v){
if(s==1) Toast.makeText(MainActivity.this, " already connected", Toast.LENGTH_LONG).show();
else {
s=1;
startMqtt(); Toast.makeText(MainActivity.this, "connected", Toast.LENGTH_LONG).show();
flag=1;
}
}
private void startMqtt() {
mqttHelper = new MqttHelper(getApplicationContext());
mqttHelper.setCallback(new MqttCallbackExtended() {
@Override
public void connectComplete(boolean b, String s) {
}
```

```
@Override
public void connectionLost(Throwable throwable) {
}
@Override
public void messageArrived(String topic, MqttMessage mqttMessage) throws Exception {
Log.w("Debug", mqttMessage.toString());
dataReceived.setText(mqttMessage.toString());
int nc =Integer.parseInt(mqttMessage.toString());
total1+=nc;
if(total1<=2500){
cons1.setText(String.valueOf(total1));te1+=nc*0.1;
sliceOne.setText(String.format(Locale.ENGLISH,"%.2f",te1));
overall_total=te1; tot=te1;
overall.setText(String.format(Locale.ENGLISH,"%.2f",overall_total));}
else{ total2+=nc; cons2.setText(String.valueOf(total2)); te2+=(nc*0.3);
sliceTwo.setText(String.format(Locale.ENGLISH,"%.2f",te2));
overall_total=te2+tot;
overall.setText(String.format(Locale.ENGLISH,"%.2f",overall_total));}
}
@Override
public void deliveryComplete(IMqttDeliveryToken iMqttDeliveryToken) {
}
});
}
public void discon(View v){
//status.setText("disconnected");
if(s==0) Toast.makeText(MainActivity.this, " already disconnected", Toast.LENGTH_LONG).show();
else{
s=0;
try{ if(s==0) Toast.makeText(MainActivity.this, " already disconnected",
Toast.LENGTH_LONG).show();
else {
s = 0;
IMqttToken token = mqttandroidclient.disconnect();
token.setActionCallback(new IMqttActionListener() {
@Override
public void onSuccess(IMqttToken AsyncActionToken) {
Toast.makeText(MainActivity.this, "disconnected", Toast.LENGTH_LONG).show();
}
@Override
public void onFailure(IMqttToken iMqttToken, Throwable throwable) {
Toast.makeText(MainActivity.this, " could not disconnecte", Toast.LENGTH_LONG).show();
}
});
}}
catch (MqttException e){
e.printStackTrace();
}
flag=0; } }
public void pump(View view) {
String swtopic="sensor/myesp";
String ON = "1";
String OFF = "0";
if(pumpSw.isChecked()){
try{mqttHelper.mqttAndroidClient.publish(swtopic,ON.getBytes(),0,false);}
catch(Exception ex){
ex.printStackTrace();}}
else{try{mqttHelper.mqttAndroidClient.publish(swtopic,OFF.getBytes(),0,false);}
```

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


// Mqtt Helper class
package helpers;
import android.content.Context;
import android.support.annotation.NonNull;
import android.util.Log;
import android.view.View;
import android.widget.TextView;
import android.widget.Toast;
import org.eclipse.paho.android.service.MqttAndroidClient;
import org.eclipse.paho.client.mqttv3.DisconnectedBufferOptions;
import org.eclipse.paho.client.mqttv3.IMqttActionListener;
import org.eclipse.paho.client.mqttv3.IMqttDeliveryToken;
import org.eclipse.paho.client.mqttv3.IMqttToken;
import org.eclipse.paho.client.mqttv3.MqttCallbackExtended;
import org.eclipse.paho.client.mqttv3.MqttConnectOptions;
import org.eclipse.paho.client.mqttv3.MqttException;
import org.eclipse.paho.client.mqttv3.MqttMessage;
public class MqttHelper {
public MqttAndroidClient mqttAndroidClient;
final String serverUri = "tcp://m14.cloudmqtt.com:18338";
final String clientId = "ExampleAndroidClient";
final String subscriptionTopic = "sensor/+";
final String subscriptionnew = "data";
final String username = "ysthqekm";
final String password = "iprkKwQKvGf4";
public MqttHelper(Context context){
mqttAndroidClient = new MqttAndroidClient(context, serverUri, clientId);
mqttAndroidClient.setCallback(new MqttCallbackExtended() {
@Override
public void connectComplete(boolean b, String s) {
Log.w("mqtt", s);
}
@Override
public void connectionLost(Throwable throwable) {
}
@Override
public void messageArrived(String topic, MqttMessage mqttMessage) throws Exception {
Log.w("Mqtt", mqttMessage.toString());
}
@Override
public void deliveryComplete(IMqttDeliveryToken iMqttDeliveryToken) {
}
});
connect();
}
public void setCallback(MqttCallbackExtended callback) {
mqttAndroidClient.setCallback(callback);
}
private void connect(){
MqttConnectOptions mqttConnectOptions = new MqttConnectOptions();
mqttConnectOptions.setAutomaticReconnect(true);
mqttConnectOptions.setCleanSession(false);
```

```
mqttConnectOptions.setUserName(username);
mqttConnectOptions.setPassword(password.toCharArray());

try {
mqttAndroidClient.connect(mqttConnectOptions, null, new IMqttActionListener() {
@Override
public void onSuccess(IMqttToken asyncActionToken) {
DisconnectedBufferOptions disconnectedBufferOptions = new DisconnectedBufferOptions();
disconnectedBufferOptions.setBufferEnabled(true);
disconnectedBufferOptions.setBufferSize(100);
disconnectedBufferOptions.setPersistBuffer(false);
disconnectedBufferOptions.setDeleteOldestMessages(false);
mqttAndroidClient.setBufferOpts(disconnectedBufferOptions);
subscribeToTopicnew();}
@Override
public void onFailure(IMqttToken asyncActionToken, Throwable exception) {
Log.w("Mqtt", "Failed to connect to: " + serverUri + exception.toString());
}
});
} catch (MqttException ex){
ex.printStackTrace(); }
}
private void subscribeToTopicnew() {
try {
mqttAndroidClient.subscribe(subscriptionnew, 0, null, new IMqttActionListener() {
@Override
public void onSuccess(IMqttToken asyncActionToken) {
Log.w("Mqtt","Subscribed!");
}
@Override
public void onFailure(IMqttToken asyncActionToken, Throwable exception) {
Log.w("Mqtt", "Subscribed fail!");
}
});
} catch (MqttException ex) {
System.err.println("Exceptionst subscribing");
ex.printStackTrace(); }
}
public void disconnect(){
MqttConnectOptions mqttConnectOptions = new MqttConnectOptions();
mqttConnectOptions.setAutomaticReconnect(false);
mqttConnectOptions.setCleanSession(true);
mqttConnectOptions.setUserName(username);
mqttConnectOptions.setPassword(password.toCharArray());
try {
mqttAndroidClient.connect(mqttConnectOptions, null, new IMqttActionListener() {
@Override
public void onSuccess(IMqttToken asyncActionToken) {
DisconnectedBufferOptions disconnectedBufferOptions = new DisconnectedBufferOptions();
disconnectedBufferOptions.setBufferEnabled(false);
disconnectedBufferOptions.setBufferSize(100);
disconnectedBufferOptions.setPersistBuffer(false);
disconnectedBufferOptions.setDeleteOldestMessages(false);
mqttAndroidClient.setBufferOpts(disconnectedBufferOptions); }
@Override
public void onFailure(IMqttToken asyncActionToken, Throwable exception) {
Log.w("Mqtt", "Failed to disconnect to: " + serverUri + exception.toString());}
});
```

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