Appendix A : Android Studio Code For Android

Monitoring :

```
public Pubnub pubnub;
    public static final String PUBLISH_KEY  =  "pub-c-798bd0f6-540b-48af-9e98-
7d0028a5132a";
    public static final String SUBSCRIBE_KEY = "sub-c-29a57572-65e7-11e6-b99b-
02ee2ddab7fe";
    public static final String CHANNEL = "M25Pi";
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_monitor);
        final TextView insideTemp = (TextView) findViewById(R.id.inside_temp);
        final TextView outsideTemp = (TextView) findViewById(R.id.outside_temp);
        pubnub = new Pubnub(PUBLISH_KEY , SUBSCRIBE_KEY);
        try {
            pubnub.subscribe(CHANNEL, new Callback() {
                @Override
                public void successCallback(String s, final Object o) {
                    super.successCallback(s, o);
                    Log.d("Monitor" , o.toString());
                    runOnUiThread(new Runnable() {
                        @Override
                        public void run() {
                            String temp = o.toString().substring(4 , 8);
                            insideTemp.setText(temp);
                        }
                    });
                }
            });
        } catch (PubnubException e) {
            e.printStackTrace();
        }
    }
}
```

Main Program :

```
import com.pubnub.api.Callback;
import com.pubnub.api.PnGcmMessage;
import com.pubnub.api.PnMessage;
import com.pubnub.api.Pubnub;
import com.pubnub.api.PubnubError;
import com.pubnub.api.PubnubException;
import org.json.JSONException;
import org.json.JSONObject;
import java.util.ArrayList;
import java.util.List;
public class MainActivity extends AppCompatActivity {
    public static Pubnub pubnub ;
    public static final String PUBLISH_KEY = "pub-c-261cd3d6-b1db-4418-a66d-
fce7158b2b16";
    public static final String SUBSCRIBE_KEY = "sub-c-c605023e-68fd-11e6-a723-
0619f8945a4f";
    public static final String CHANNEL = "firstApp";
    static SharedPreferences preferences ;
    SharedPreferences.Editor prefEditor ;
    //boolean roomLight = false , gardenLight = false, frontDoor = false,
garageDoor = false, airConditioner = false, waterPumb = false;
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        initPubnub();
        startService(new Intent(this , RegistrationService.class));
        List<Appliance> appliances = new ArrayList<>();
        String names [] = {"Room Light" , "Garden Light" , "FrontDoor" ,"Garage
Door" , "Air Conditioner" , "Water Pump"};
        String statuses [] = {"Off" , "Off" , "Closed" , "Closed" , "Off" ,
"Off"};
        int images [] = {R.mipmap.light , R.mipmap.bulb , R.mipmap.door ,
R.mipmap.garage , R.mipmap.air , R.mipmap.water };
        for(int i=0 ; i<names.length ; i++){
            Appliance appliance = new Appliance(names[i] , statuses[i] ,
images[i]);
            appliances.add(appliance);
        }
        RecyclerView recyclerView = (RecyclerView)
findViewById(R.id.recyclerView);
        recyclerView.setLayoutManager(new LinearLayoutManager(this));
        SmartHomeAdapter smartHomeAdapter = new SmartHomeAdapter(appliances);
        recyclerView.setAdapter(smartHomeAdapter);
        preferences = PreferenceManager.getDefaultSharedPreferences(this);
        prefEditor = preferences.edit();
    }
    private void initPubnub() {
        this.pubnub = new Pubnub(PUBLISH_KEY , SUBSCRIBE_KEY);
        this.pubnub.setUUID("firstAppUUID");
        subscribe();
    }
    private void subscribe() {
        try {
                pubnub.subscribe(CHANNEL , new Callback(){
                @Override
                public void successCallback(String s, Object o) {
                    super.successCallback(s, o);
                    Log.d("PUBNUBsuccess","SUBSCRIBE : CONNECT on channel:" +
CHANNEL
                            + " : " + o.getClass() + " : "
                            + o.toString());
                }
                @Override
                public void errorCallback(String s, PubnubError pubnubError) {
                    super.errorCallback(s, pubnubError);
                }
                @Override
                public void connectCallback(String s, Object o) {
                    super.connectCallback(s, o);
                    Log.d("PUBNUBconnect","SUBSCRIBE : CONNECT on channel:" +
CHANNEL
                            + " : " + o.getClass() + " : "
                            + o.toString());
                }
                @Override
                public void reconnectCallback(String s, Object o) {
                    super.reconnectCallback(s, o);
                }
                @Override
                public void disconnectCallback(String s, Object o) {
                    super.disconnectCallback(s, o);
                }
            });
        } catch (Exception e){
            Log.d("ErrorInSubscribing" , " Error while subscribing");
            e.printStackTrace();
```

```
        }
    }
    public void LEDOff(View view) {
        pubnub.enablePushNotificationsOnChannel("firstApp" ,
preferences.getString("token" , ""));
        JSONObject json = new JSONObject();
        try {
            json.put("LED" , 0);
        }catch (Exception e) {
        }
        Callback callback = new Callback() {
            @Override
            public void successCallback(String s, Object o) {
                super.successCallback(s, o);
            }
            @Override
            public void errorCallback(String s, PubnubError pubnubError) {
                super.errorCallback(s, pubnubError);
            }
        };
        this.pubnub.publish(CHANNEL , json , callback  );
    }
    public void LEDOn(View view) {
        pubnub.enablePushNotificationsOnChannel("firstApp" ,
preferences.getString("token" , ""));
        JSONObject json = new JSONObject();
        try {
            json.put("LED" , 1);
        }catch (Exception e) {
        }
        Callback callback = new Callback() {
            @Override
            public void successCallback(String s, Object o) {
                super.successCallback(s, o);
                Log.d("PUBNUB","SUBSCRIBE : CONNECT on channel:" + CHANNEL
                        + " : " + o.getClass() + " : "
                        + o.toString());
            }
            @Override
            public void errorCallback(String s, PubnubError pubnubError) {
                super.errorCallback(s, pubnubError);
                Log.d("PUBNUB","SUBSCRIBE : CONNECT on channel:" + CHANNEL
                        + " : " + pubnubError.getClass() + " : "
                        + pubnubError.toString());
            }
        };
        this.pubnub.publish(CHANNEL , json , callback  );
    }
    public void sendNotification(View view) {
        pubnub.enablePushNotificationsOnChannel("firstApp" ,
preferences.getString("token" , ""));
        Toast.makeText(this , "Your Token is"+preferences.getString("token" ,
"No Token Found") , Toast.LENGTH_SHORT).show();
        PnGcmMessage gcmMessage = new PnGcmMessage();
        JSONObject json = new JSONObject();
        try {
            json.put("LED", 1);
        } catch (JSONException e) { }
        gcmMessage.setData(json);
        PnMessage message = new PnMessage(
                pubnub,
                "firstApp",
                callback,
                gcmMessage);
        try {
            message.publish();
```

```
        } catch (PubnubException e) {
            e.printStackTrace();
        }
    }
    public static Callback callback = new Callback() {
        @Override
        public void successCallback(String channel, Object message) {
            Log.d("Recieved", "Success on Channel " + CHANNEL + " : " +
message);
        }
        @Override
        public void errorCallback(String channel, PubnubError error) {
            Log.d("  ", "Error On Channel " + CHANNEL + " : " + error);
        }
    };
    public void navigate (View view) {
        Intent intent = new Intent(MainActivity.this , Monitor.class);
        startActivity(intent);
    }
}
```

Receiving Code :

```
import android.support.v7.widget.RecyclerView;
import android.text.Layout;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.Switch;
import android.widget.TextView;
import com.pubnub.api.Callback;
import com.pubnub.api.PubnubError;
import org.json.JSONObject;
import java.util.List;
public class SmartHomeAdapter  extends
RecyclerView.Adapter<SmartHomeAdapter.SmartHomeHolder> {
    List<Appliance> appliances;
    public SmartHomeAdapter(List<Appliance> appliances) {
        this.appliances = appliances;
    }
    @Override
    public SmartHomeAdapter.SmartHomeHolder onCreateViewHolder(ViewGroup parent,
int viewType) {
        View raw =
LayoutInflater.from(parent.getContext()).inflate(R.layout.home_raw , parent ,
false);
        SmartHomeHolder smartHomeHolder = new SmartHomeHolder(raw);
        return smartHomeHolder;
    }
    @Override
    public void onBindViewHolder(SmartHomeAdapter.SmartHomeHolder holder, int
position) {
        Appliance appliance = appliances.get(position);
        holder.appliance.setText(appliance.applianceName);
        holder.statusValue.setText(appliance.applianceStatus);
        holder.image.setImageResource(appliance.applianceImage);
    }
    @Override
    public int getItemCount() {
        return appliances.size();
    }
    class SmartHomeHolder extends RecyclerView.ViewHolder {
        TextView appliance , status , statusValue ;
```

```
        Switch aSwitch ;
        ImageView image;
        public SmartHomeHolder(View itemView) {
            super(itemView);
            appliance = (TextView) itemView.findViewById(R.id.appliance);
            status = (TextView) itemView.findViewById(R.id.status);
            statusValue = (TextView) itemView.findViewById(R.id.status_value);
            aSwitch = (Switch) itemView.findViewById(R.id.switch1);
            image = (ImageView) itemView.findViewById(R.id.imageView);
            aSwitch.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    Appliance appliance =
appliances.get(getAdapterPosition());
                    if(aSwitch.isChecked()) {
                        LEDOn(v , 2*getAdapterPosition()+1);
                        switch (getAdapterPosition()) {
                            case 2:
                            case 3:
                                statusValue.setText("Opened");
                                break;
                            default:
                                statusValue.setText("On");
                        }
                    } else {
                        LEDOff(v , 2*getAdapterPosition());
                        switch (getAdapterPosition()){
                            case 2:
                            case 3:
                                statusValue.setText("Closed");
                                break;
                            default:
                                statusValue.setText("Off");
                        }
                    }
                }
            });
        }
    }
    public void LEDOn(View view , int data) {
        MainActivity.pubnub.enablePushNotificationsOnChannel("firstApp" ,
MainActivity.preferences.getString("token" , ""));
        JSONObject json = new JSONObject();
        try {
            json.put("LED" , data);
        }catch (Exception e) {
        }
        Callback callback = new Callback() {
            @Override
            public void successCallback(String s, Object o) {
                super.successCallback(s, o);
                Log.d("PUBNUB","SUBSCRIBE : CONNECT on channel:" +
MainActivity.CHANNEL
                        + " : " + o.getClass() + " : "
                        + o.toString());
            }
            @Override
            public void errorCallback(String s, PubnubError pubnubError) {
                super.errorCallback(s, pubnubError);
                Log.d("PUBNUB","SUBSCRIBE : CONNECT on channel:" +
MainActivity.CHANNEL
                        + " : " + pubnubError.getClass() + " : "
                        + pubnubError.toString());
            }
        };
        MainActivity.pubnub.publish(MainActivity.CHANNEL , json , callback  );
```

```
    }
    public void LEDOff(View view , int data) {
        MainActivity.pubnub.enablePushNotificationsOnChannel("firstApp" ,
MainActivity.preferences.getString("token" , ""));
        JSONObject json = new JSONObject();
        try {
            json.put("LED" , data);
        }catch (Exception e) {
        }
        Callback callback = new Callback() {
            @Override
            public void successCallback(String s, Object o) {
                super.successCallback(s, o);
            }
            @Override
            public void errorCallback(String s, PubnubError pubnubError) {
                super.errorCallback(s, pubnubError);
            }
        };
        MainActivity.pubnub.publish(MainActivity.CHANNEL , json , callback  );
    }
}
```

Registration :

```
import android.app.IntentService;
import android.content.Intent;
import android.content.SharedPreferences;
import android.preference.PreferenceManager;
import android.util.Log;
import android.widget.Toast;
import com.google.android.gms.gcm.GoogleCloudMessaging;
import com.google.android.gms.iid.InstanceID;
import com.pubnub.api.Pubnub;
import java.io.IOException;
/**
 * Created by Mohammed on 10/9/2016.
 */
public class RegistrationService extends IntentService {
    SharedPreferences preferences ;
    SharedPreferences.Editor prefEditor ;
    static String staticToken ;
    private final Pubnub pubnub = new Pubnub("pub-c-261cd3d6-b1db-4418-a66d-
fce7158b2b16" ,
            "sub-c-c605023e-68fd-11e6-a723-0619f8945a4f");

    public RegistrationService(String name) {
        super(name);
    }
    public RegistrationService () {
        super("");
    }
    @Override
    protected void onHandleIntent(Intent intent) {
        preferences = PreferenceManager.getDefaultSharedPreferences(this);
        prefEditor = preferences.edit();
        InstanceID instanceID = InstanceID.getInstance(this);
        try {
              String token = instanceID.getToken("210258410243" ,
GoogleCloudMessaging.INSTANCE_ID_SCOPE , null);
              prefEditor.putString("token" , token).apply();
              Log.d("Token", "Done get Token "+token);
              if(!preferences.getBoolean("token_sent" , false)) {
                sendTokenToServer(token);
              }
```

```
        } catch (IOException e) {
            e.printStackTrace();
            Log.d("Registration Service", "Error :get Token Failed !");
        }
    }
    private void sendTokenToServer(String token) {
        pubnub.enablePushNotificationsOnChannel("firstApp" , token);
        prefEditor.putBoolean("token_sent" , true);
    }
}
```

Appendix B : Code For Raspberry Pi :

Receiving Code

```
import time
import RPi.GPIO as GPIO
from pubnub import Pubnub
from nanpy import (ArduinoApi , SerialManager)
from time import strftime


GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(18,GPIO.OUT)


pubnub = Pubnub(publish_key = 'pub-c-261cd3d6-b1db-4418-a66d-
fce7158b2b16' ,subscribe_key = 'sub-c-c605023e-68fd-11e6-a723-0619f8945a4f')
channel = 'firstApp'

try:
    connection = SerialManager()
    a = ArduinoApi(connection = connection)
except:
    print("failed to connect to the arduino ")

a.pinMode(2 , a.INPUT)
a.pinMode(13 , a.OUTPUT)

LEDStatus = False
```

```
def notify() :

     pubnub.publish ('firstApp' , {
       "message": "Your Smart Home" ,
       "pn_gcm": {

       "data": {
          "title": "At :"+ str(strftime("%H:%M")) ,
          "message": "Motion Detected"
       }
      },

       "pn_debug": "true"
    } , callback = _callback, error = _error )


def turnLEDOn() :
    GPIO.output(18,1)
    LEDStatus = True
    print("LED is ON")
    pubnub.publish(channel=channel , message = {"status":"LED is ON"} , error = _error)


def turnLEDOff() :
    GPIO.output(18,0)
    LEDstatus = False
    pubnub.publish(channel=channel , message = {"status":"LED is OFF"} , error = _error)


def _error(m):
    print(m)



def _callback(msg , n) :
    print(msg)

    if (msg['LED'] == 1) :

        print("LED is On")
        a.digitalWrite(13 , a.HIGH )

    elif (msg['LED'] == 0) :

        print("LED is Off")
        a.digitalWrite(13 , a.LOW)

        # pubnub.publish(channel=channel , message = {"title":"LED is ON"} , error =
_error)
```

```
    #pubnub.publish(channel=channel , message = {"status":" "+str(a.analogRead(0)/9.3)} ,
error = _error)




pubnub.subscribe(channels = channel , callback = _callback , error = _error)
```

Sending Code :

```
import time
import RPi.GPIO as GPIO
from pubnub import Pubnub
from nanpy import (ArduinoApi , SerialManager)
from time import strftime


GPIO.setwarnings(False)

pubnub = Pubnub(publish_key = 'pub-c-798bd0f6-540b-48af-9e98-
7d0028a5132a' ,subscribe_key = 'sub-c-29a57572-65e7-11e6-b99b-02ee2ddab7fe')
channel = 'M25Pi'


try:
    connection = SerialManager()
    a = ArduinoApi(connection = connection)
except:
    print("failed to connect to the arduino ")


def notify() :

     pubnub.publish ('firstApp' , {
       "message": "Your Smart Home" ,
       "pn_gcm": {

       "data": {
          "title": "At :"+ str(strftime("%H:%M")) ,
          "message": "Motion Detected"
       }
      },

       "pn_debug": "true"
    } , callback = _callback, error = _error )
```

```
def _error(m):
    print(m)

def _callback(msg) :
    print(msg)

while True :

    val = a.analogRead(0)
    print(val)
    temp = (val*0.4880)
    message = {'': temp , 'out': temp+4.6}
    print(temp)
    pubnub.publish (channel = channel , message = message , callback = _callback , error
= _error)
    if a.digitalRead(2) :
        print("Motion Detected")
        #notify()
    else :
        print("No Motion Detected")


    time.sleep(2)



pubnub.subscribe(channels = channel , callback = _callback , error = _error)
```