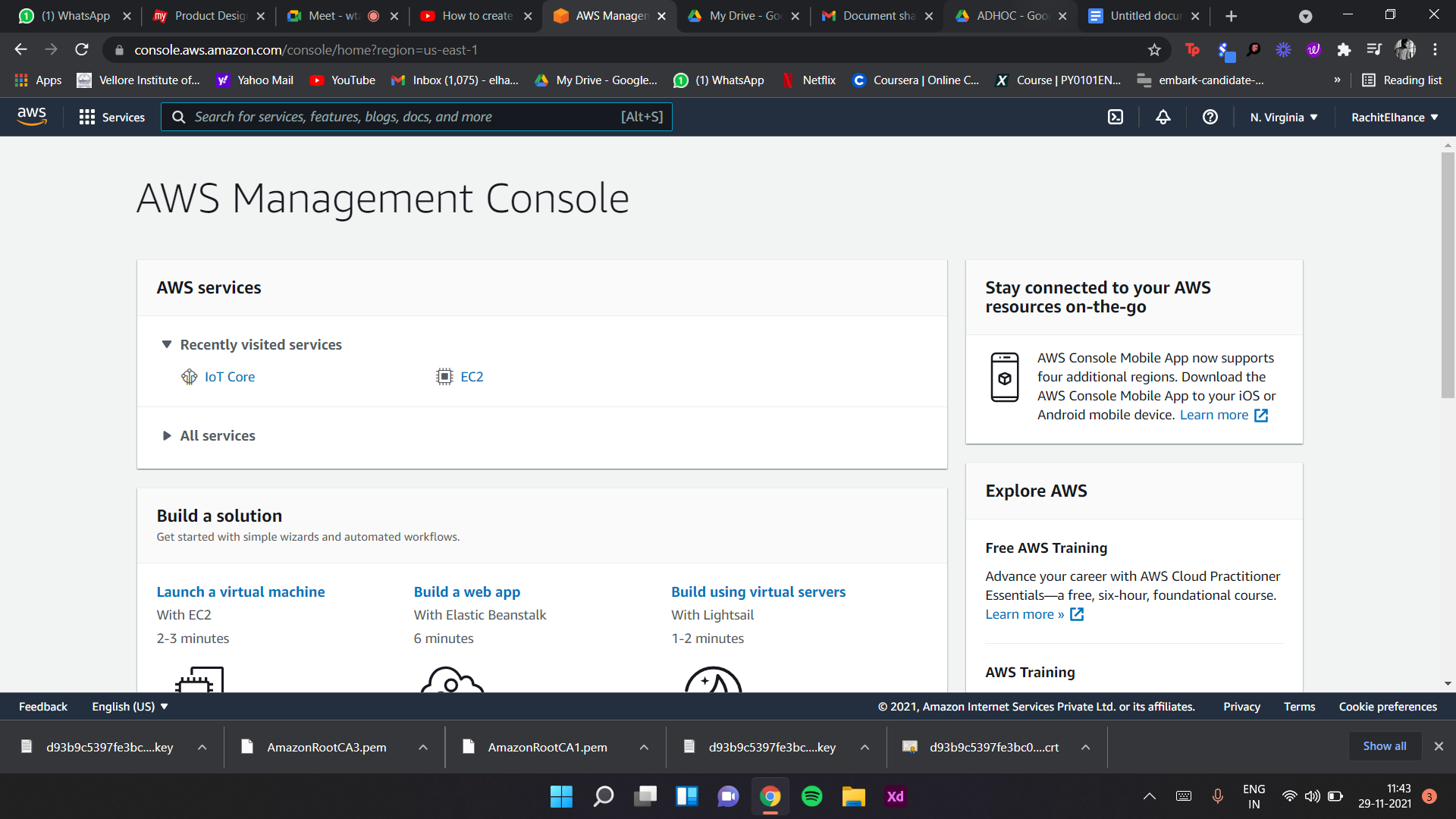
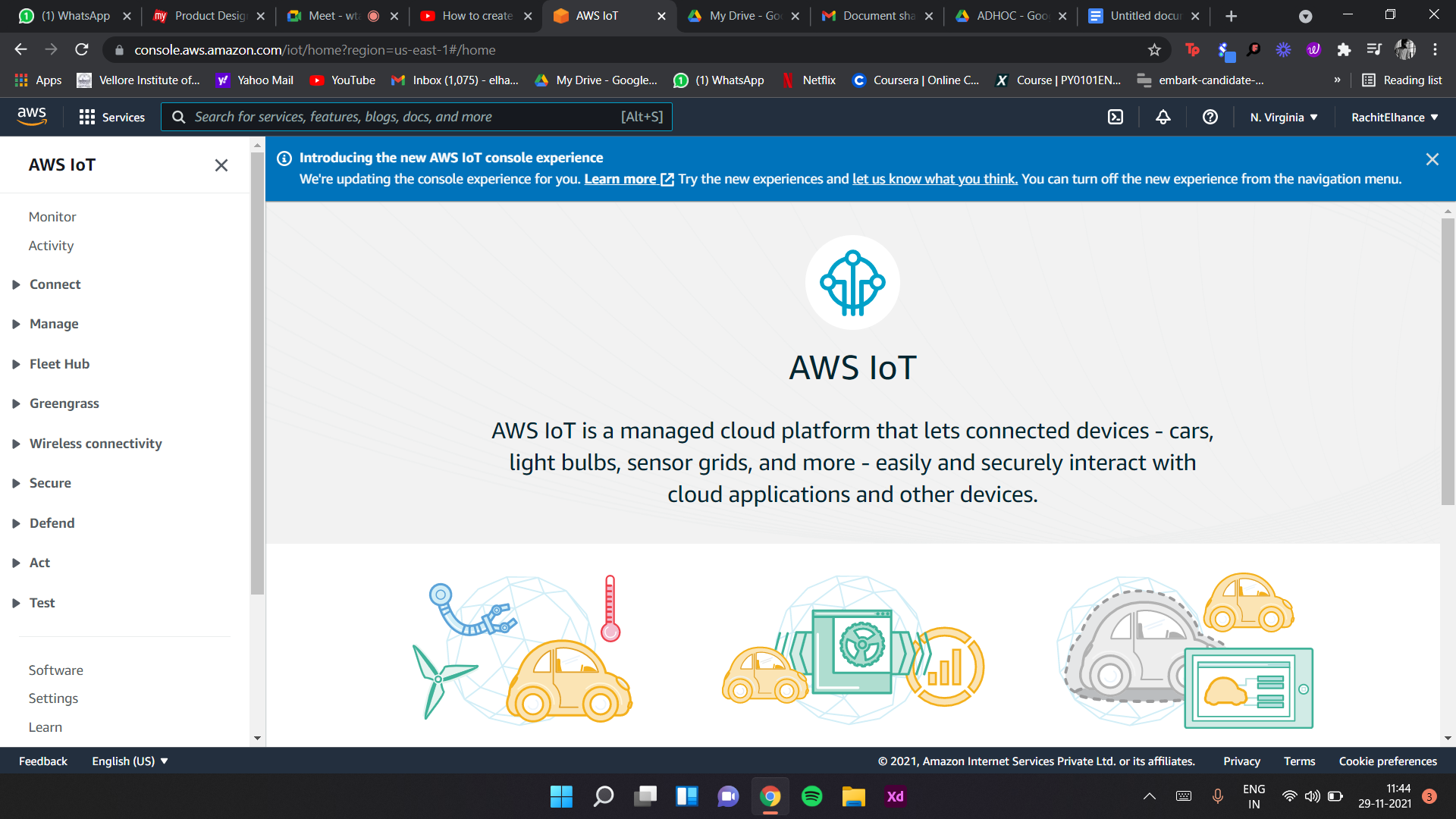
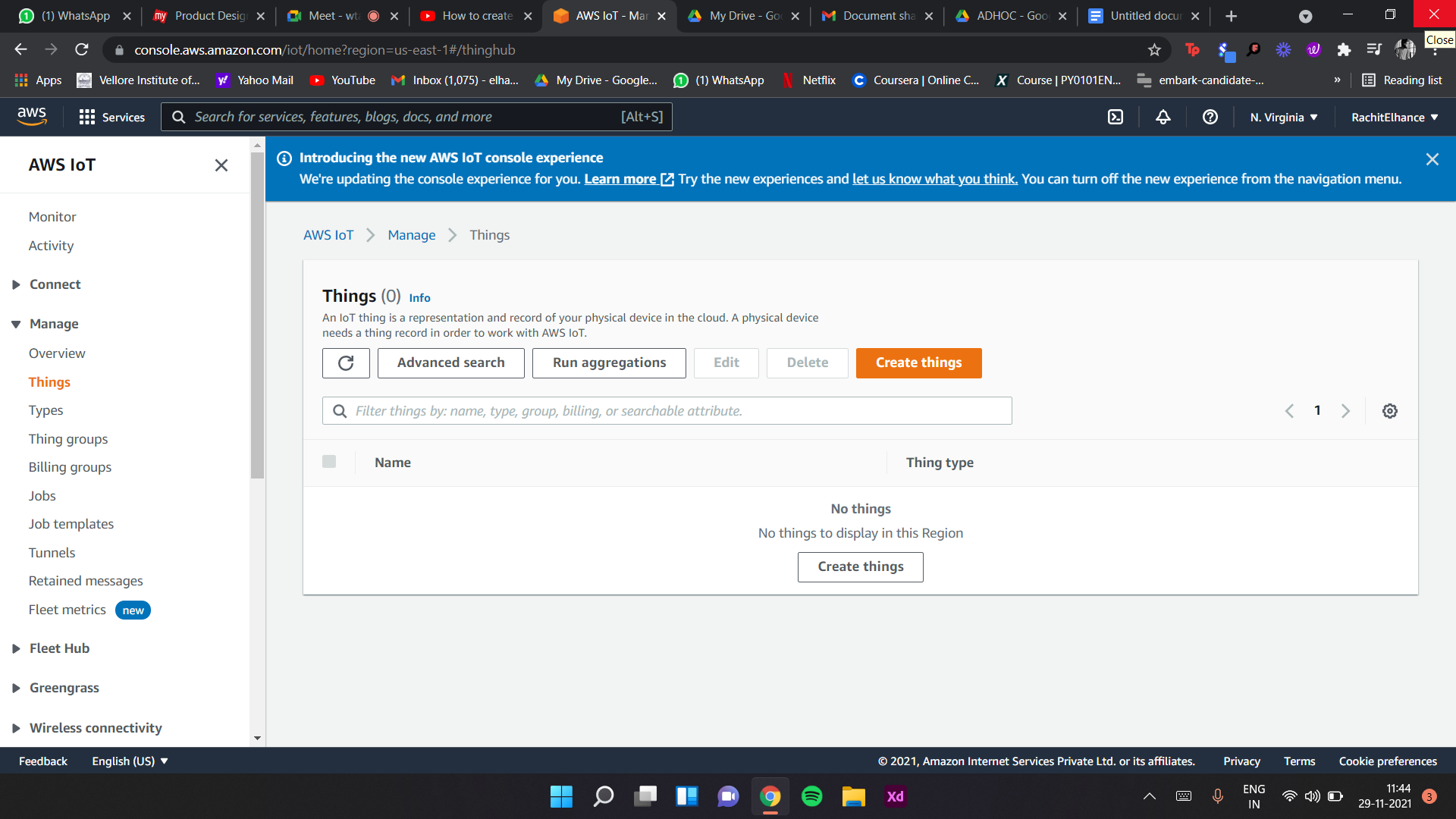
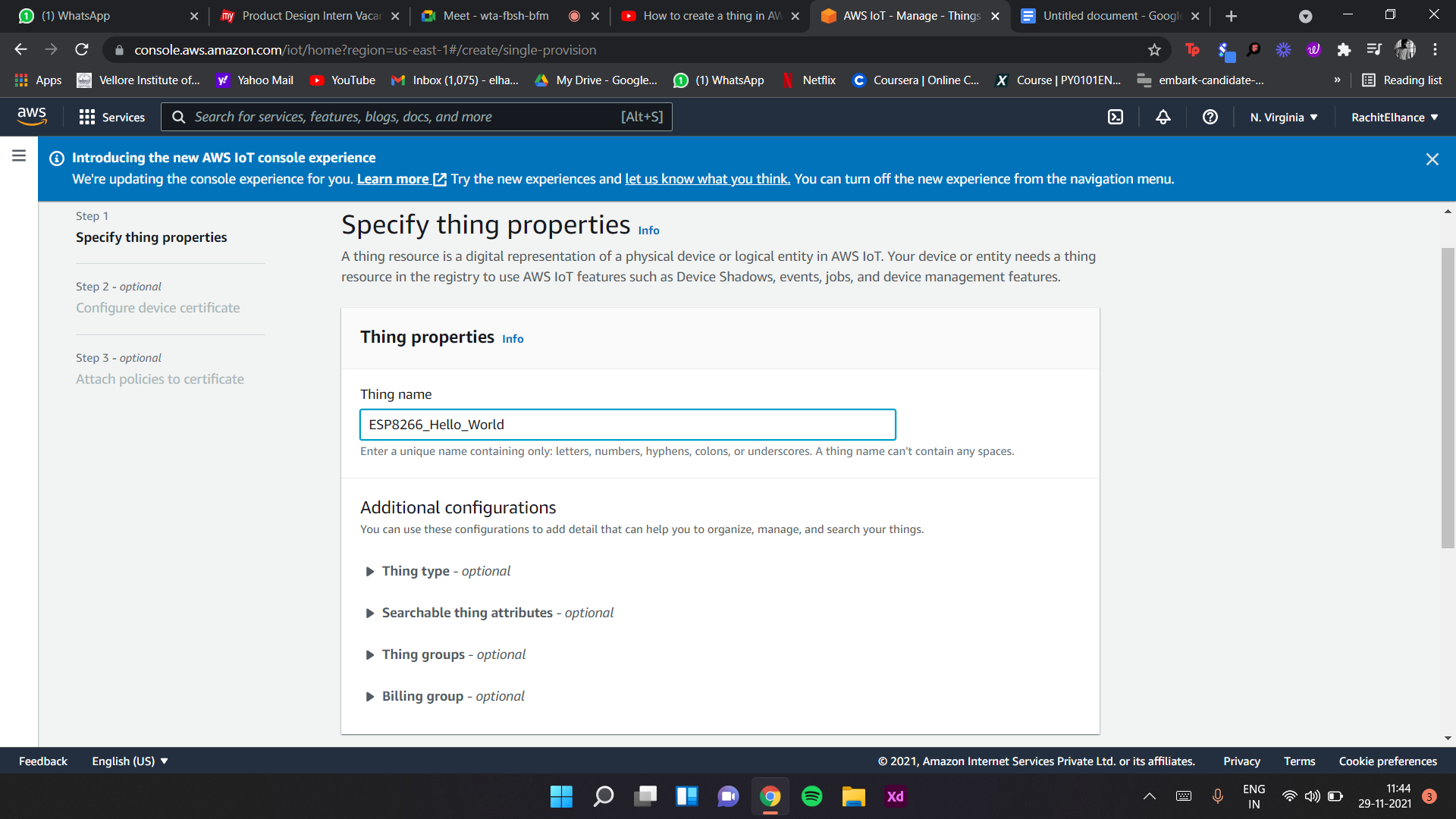
Amazon Console

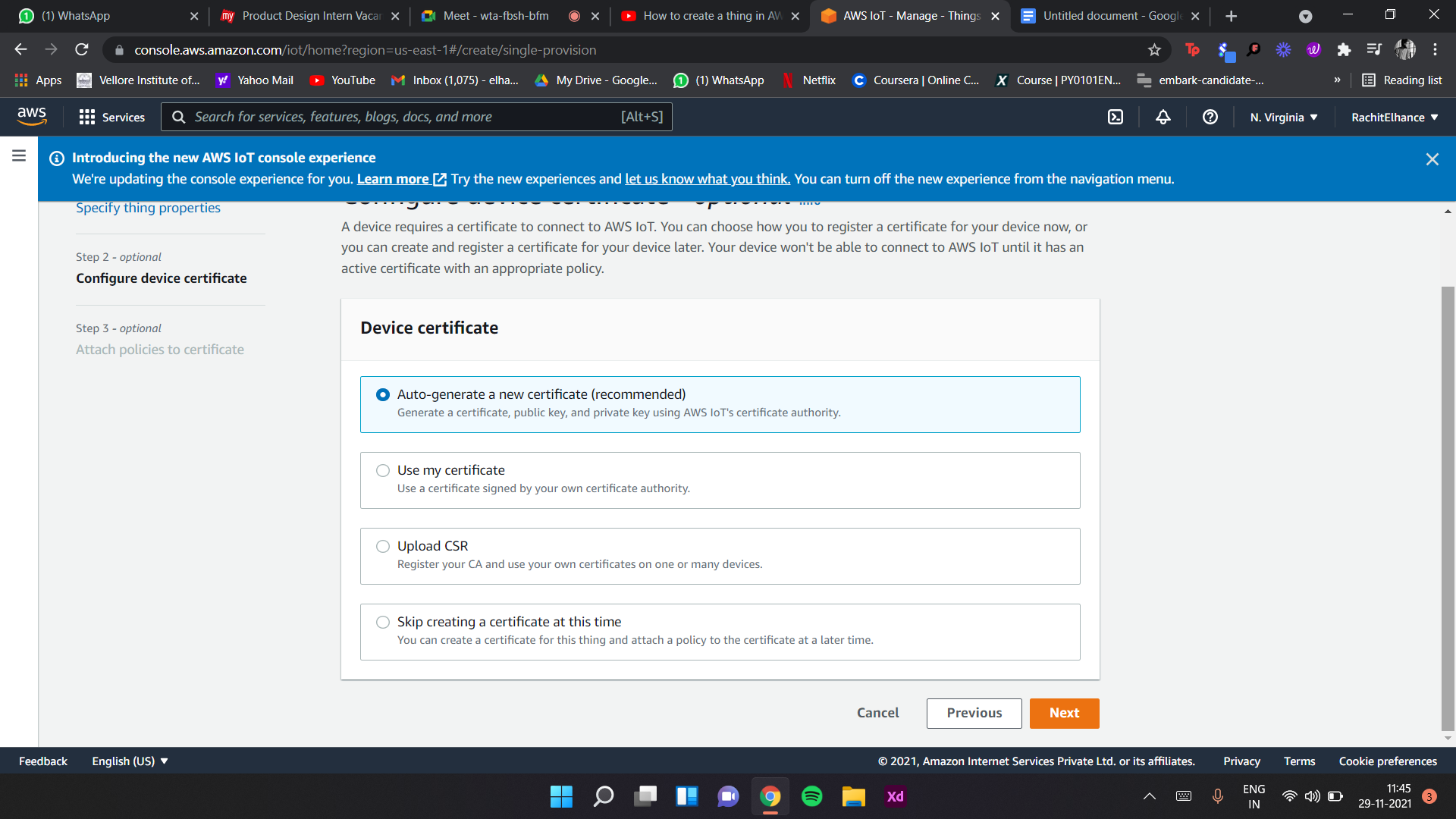


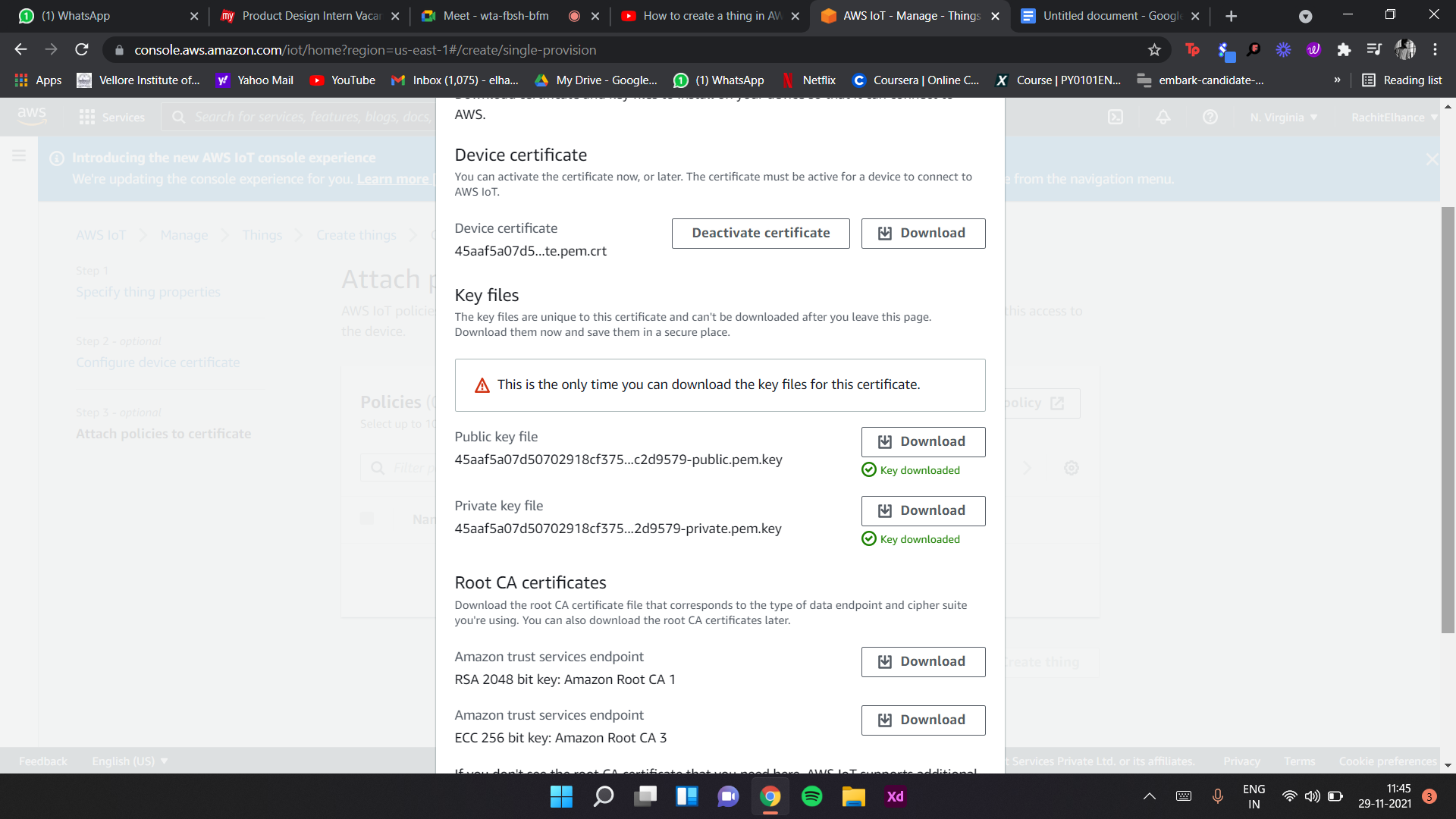
Selecting IoT core service

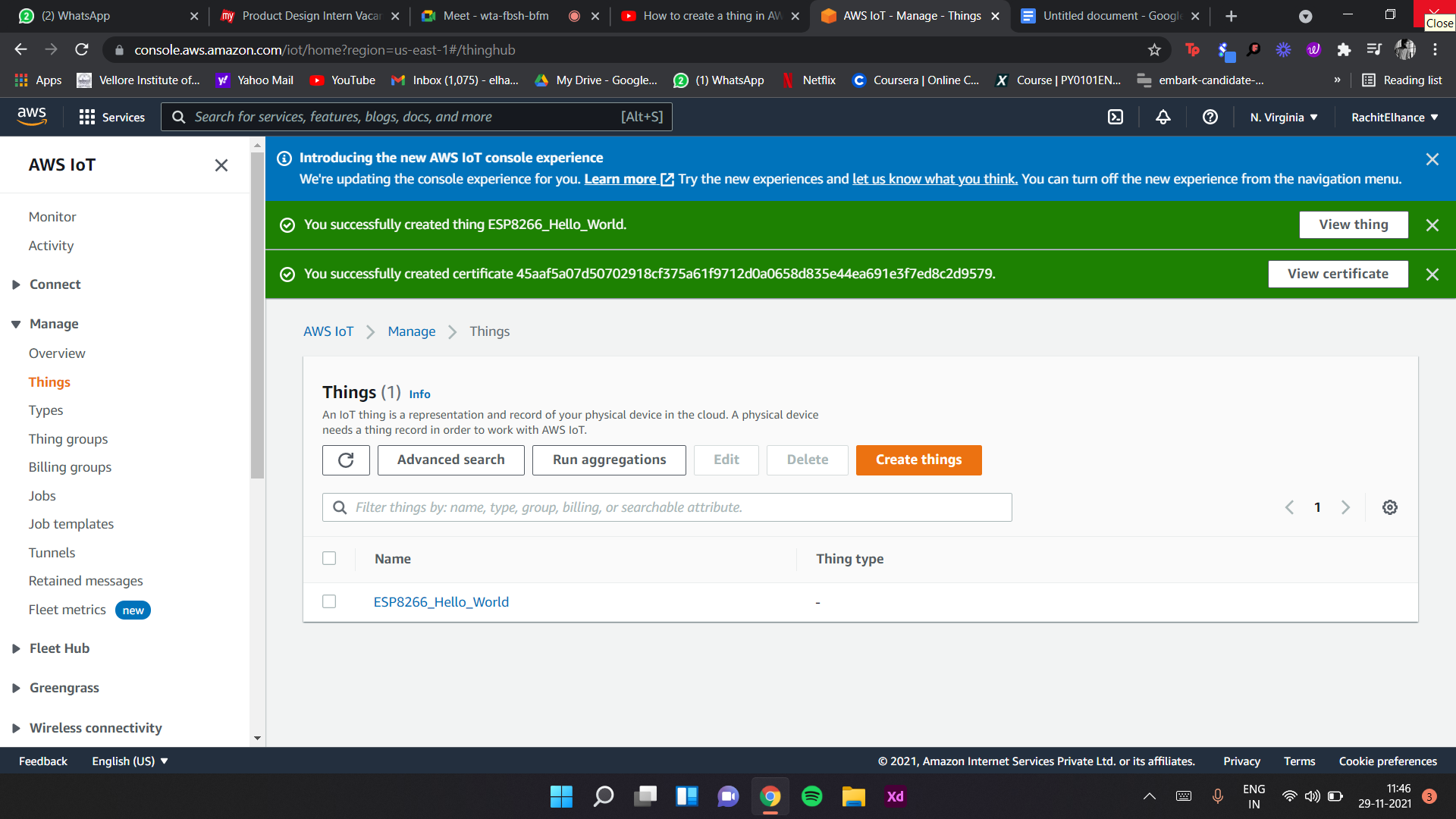


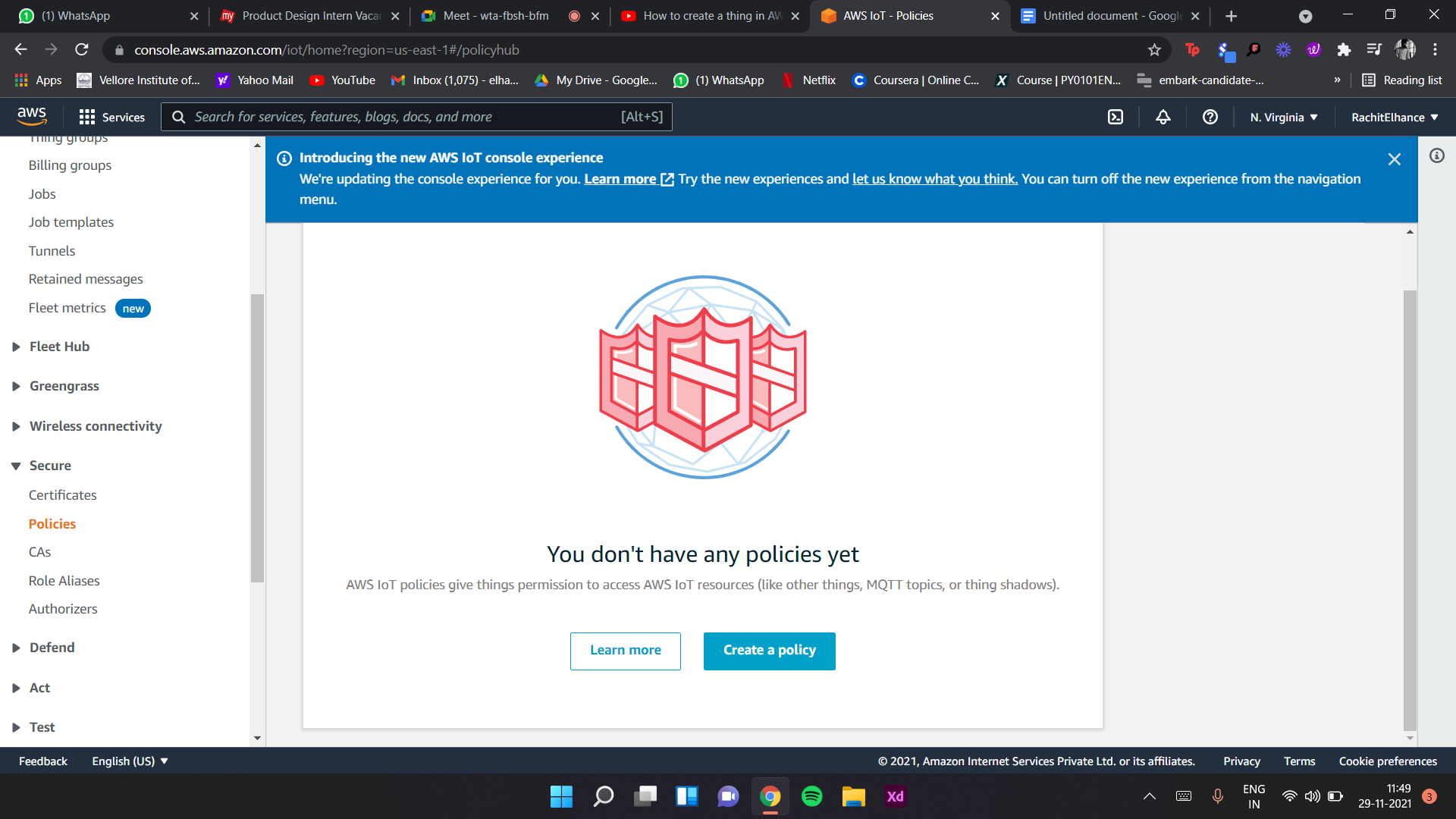




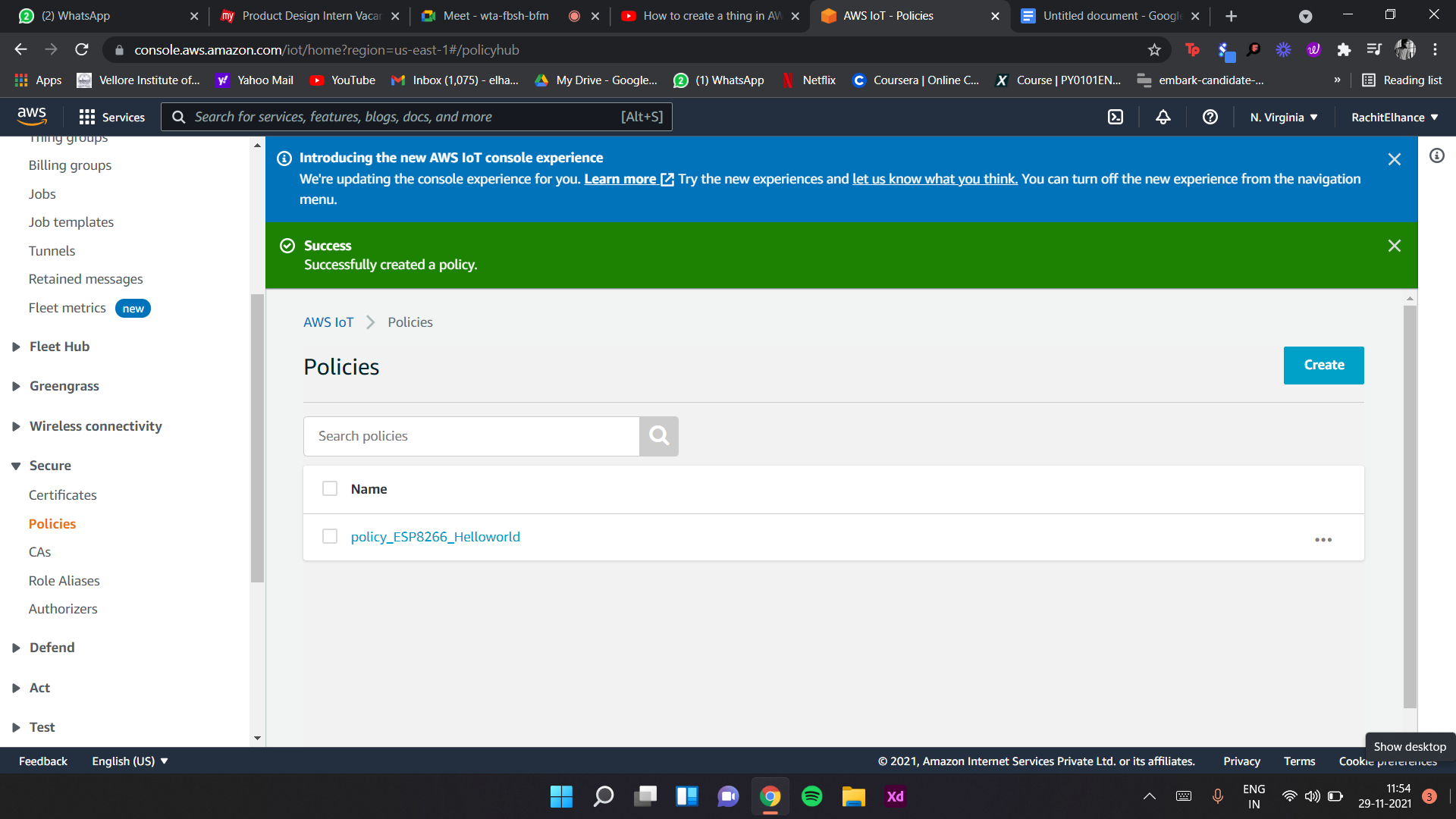


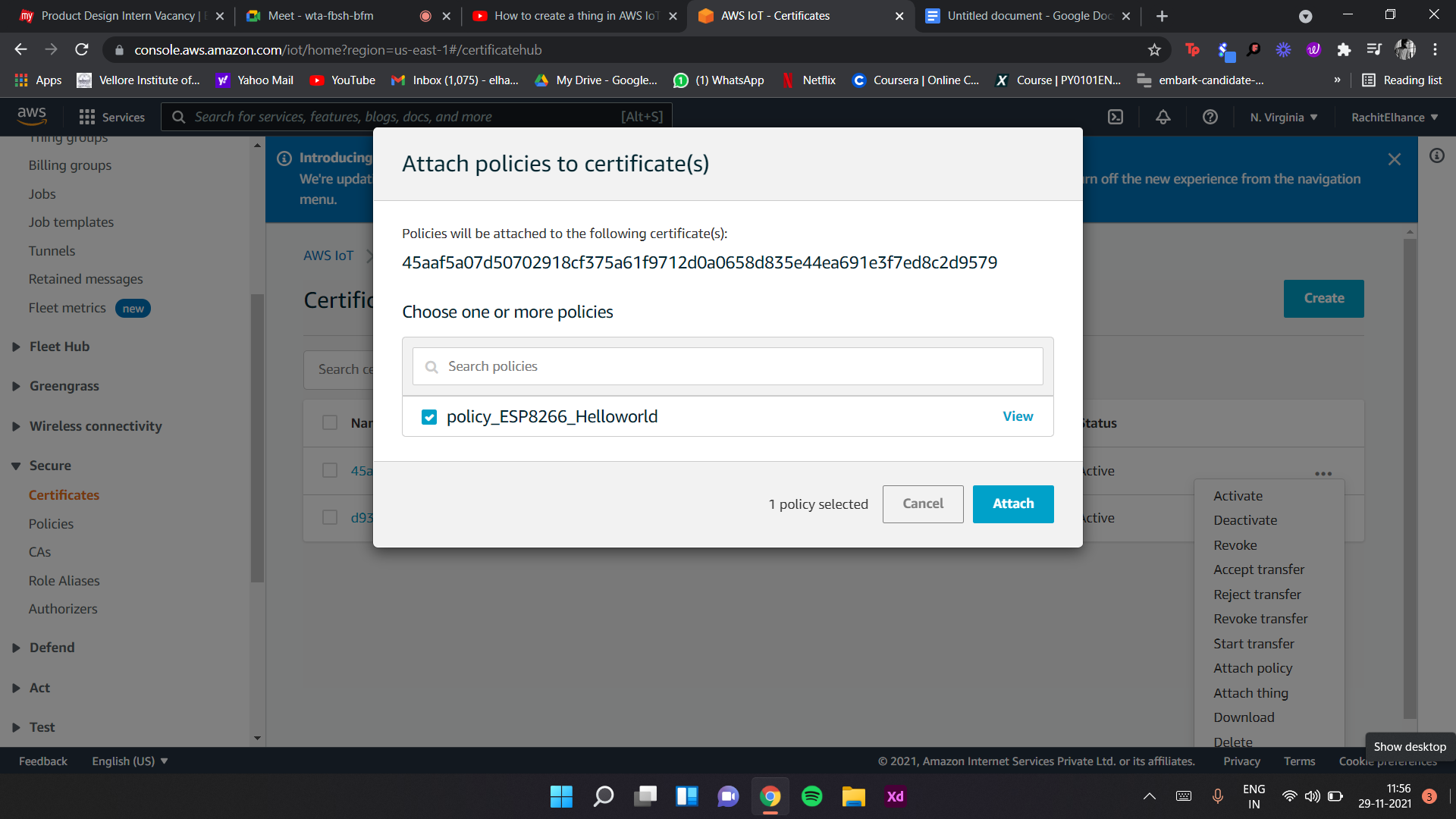


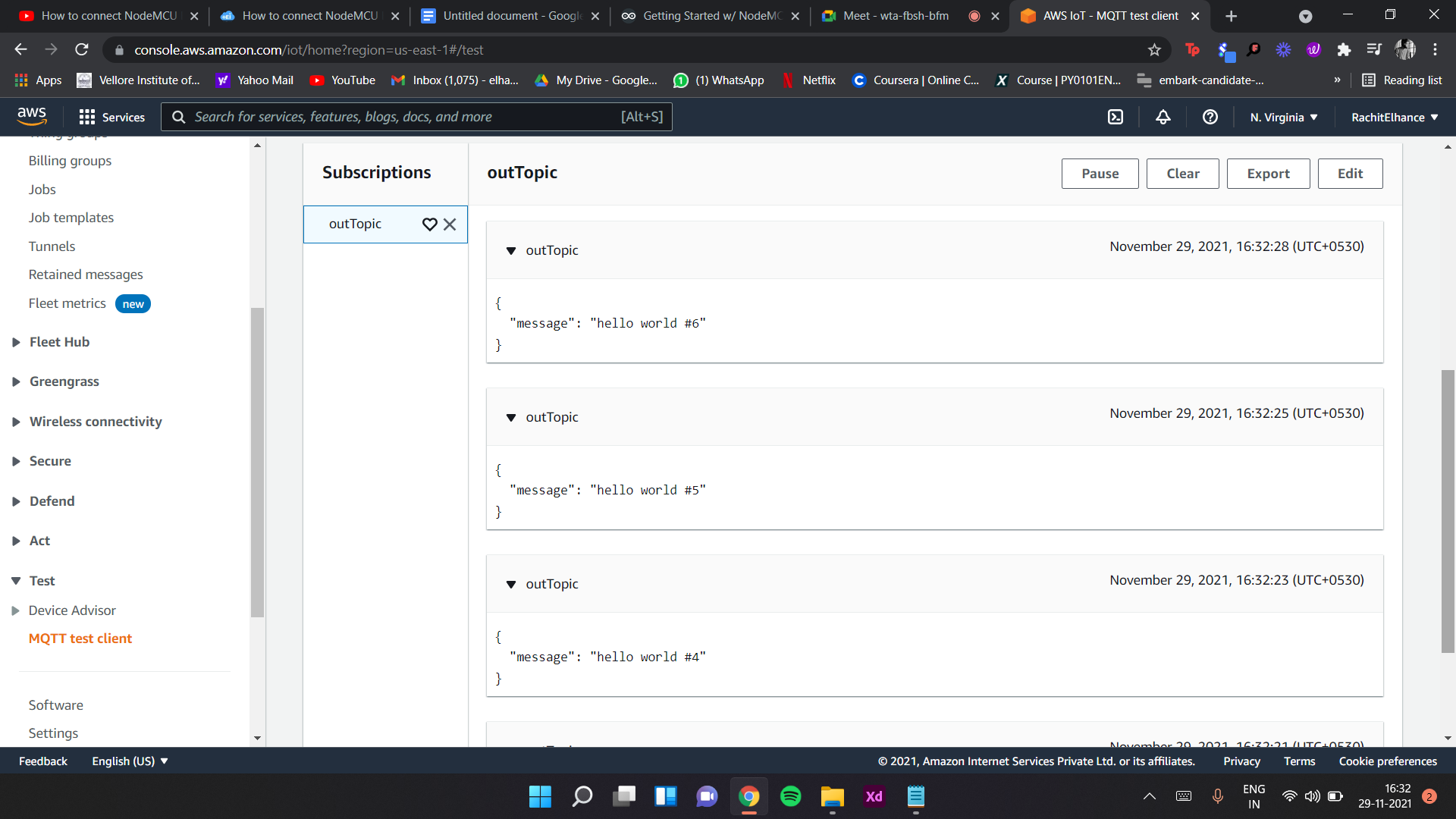


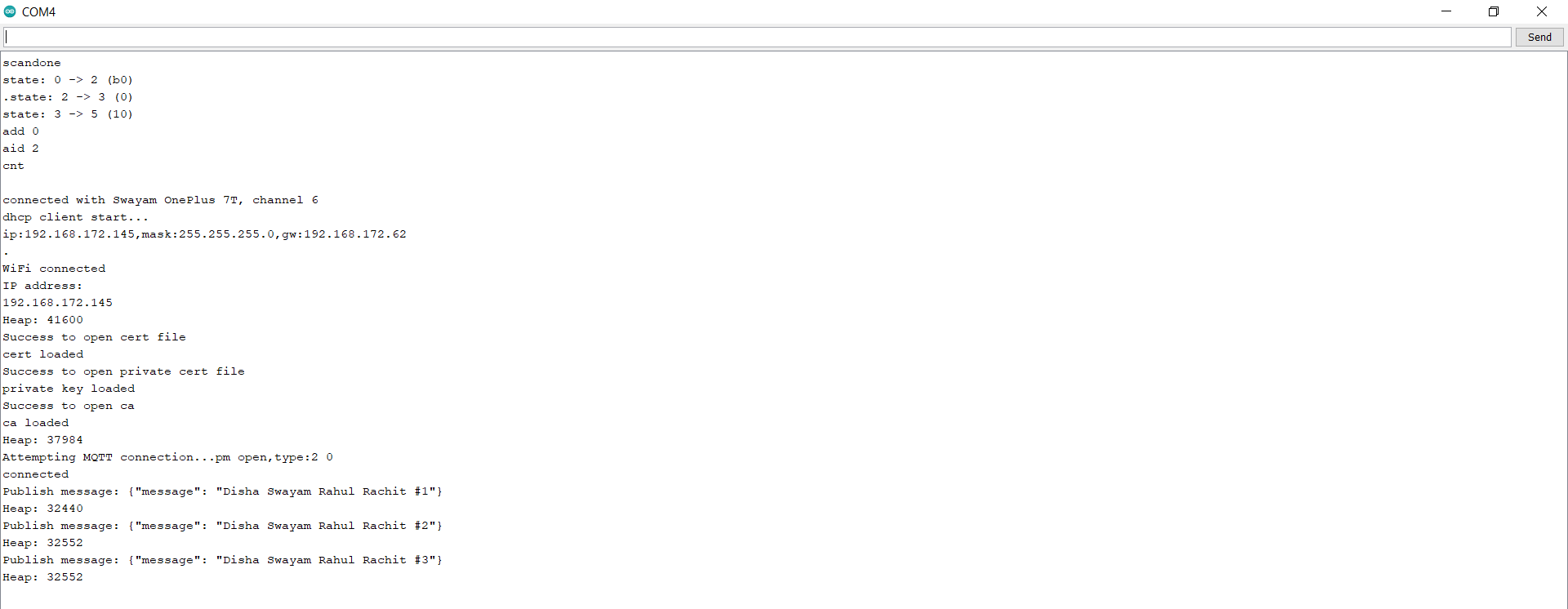


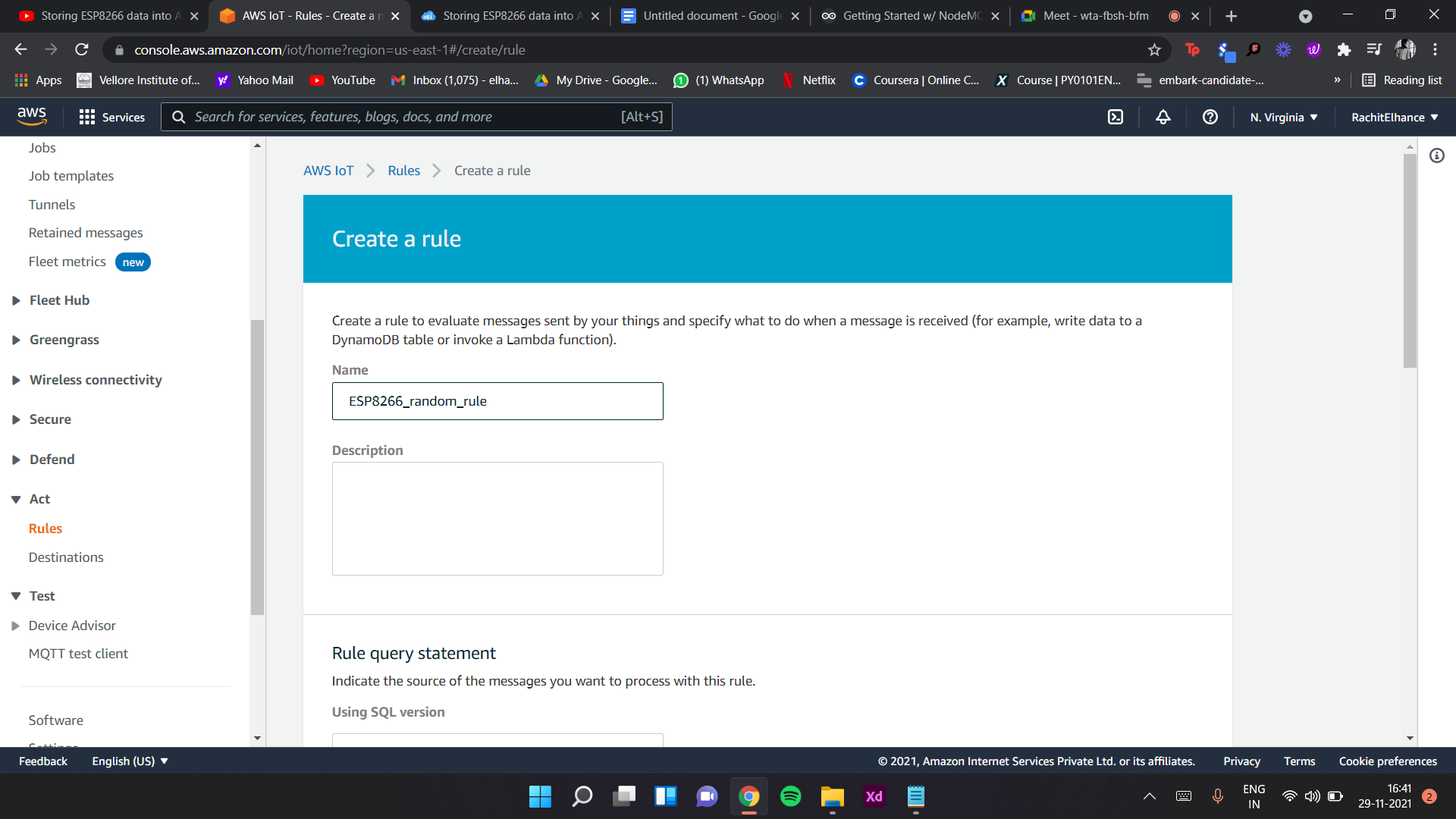


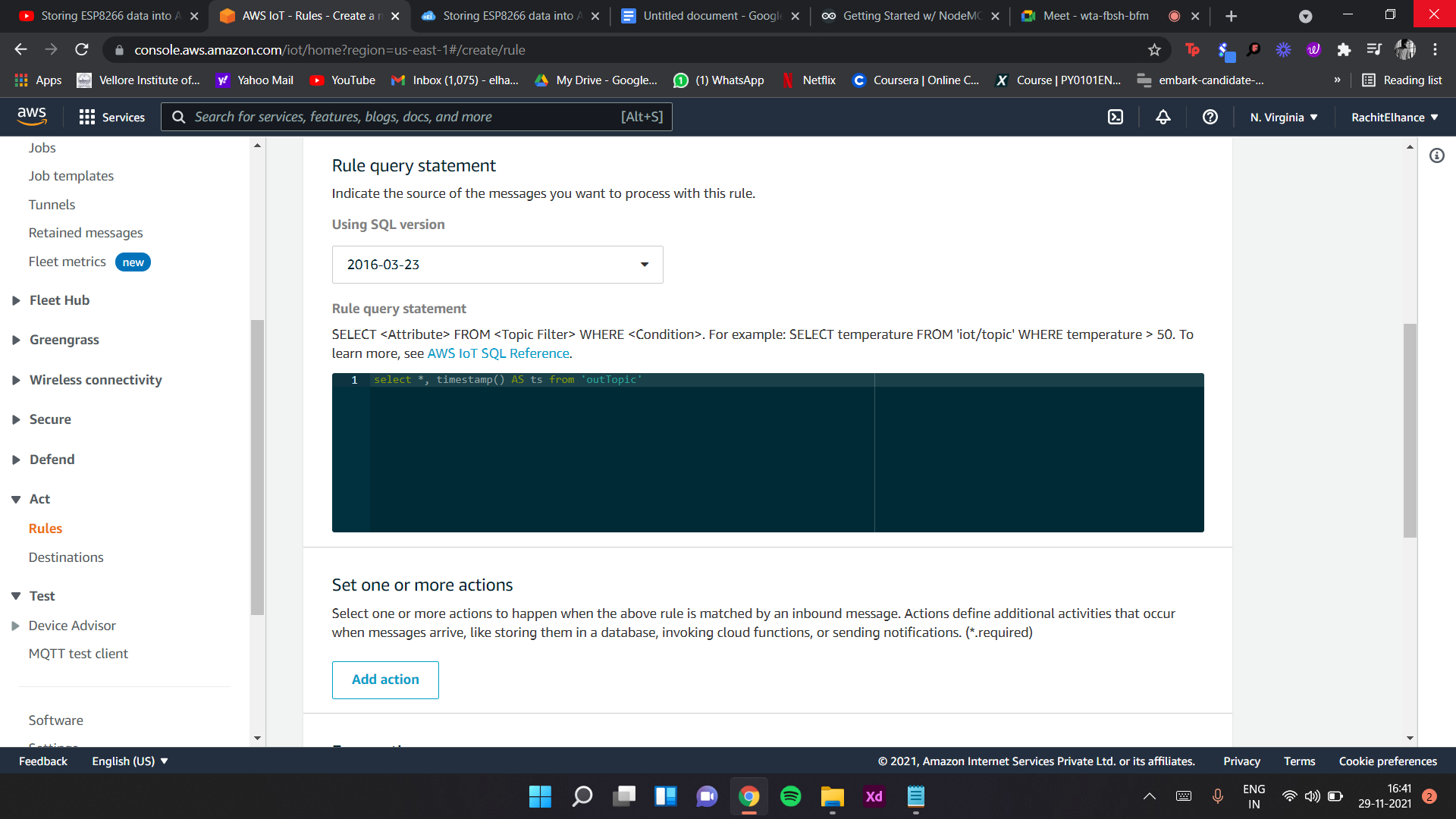


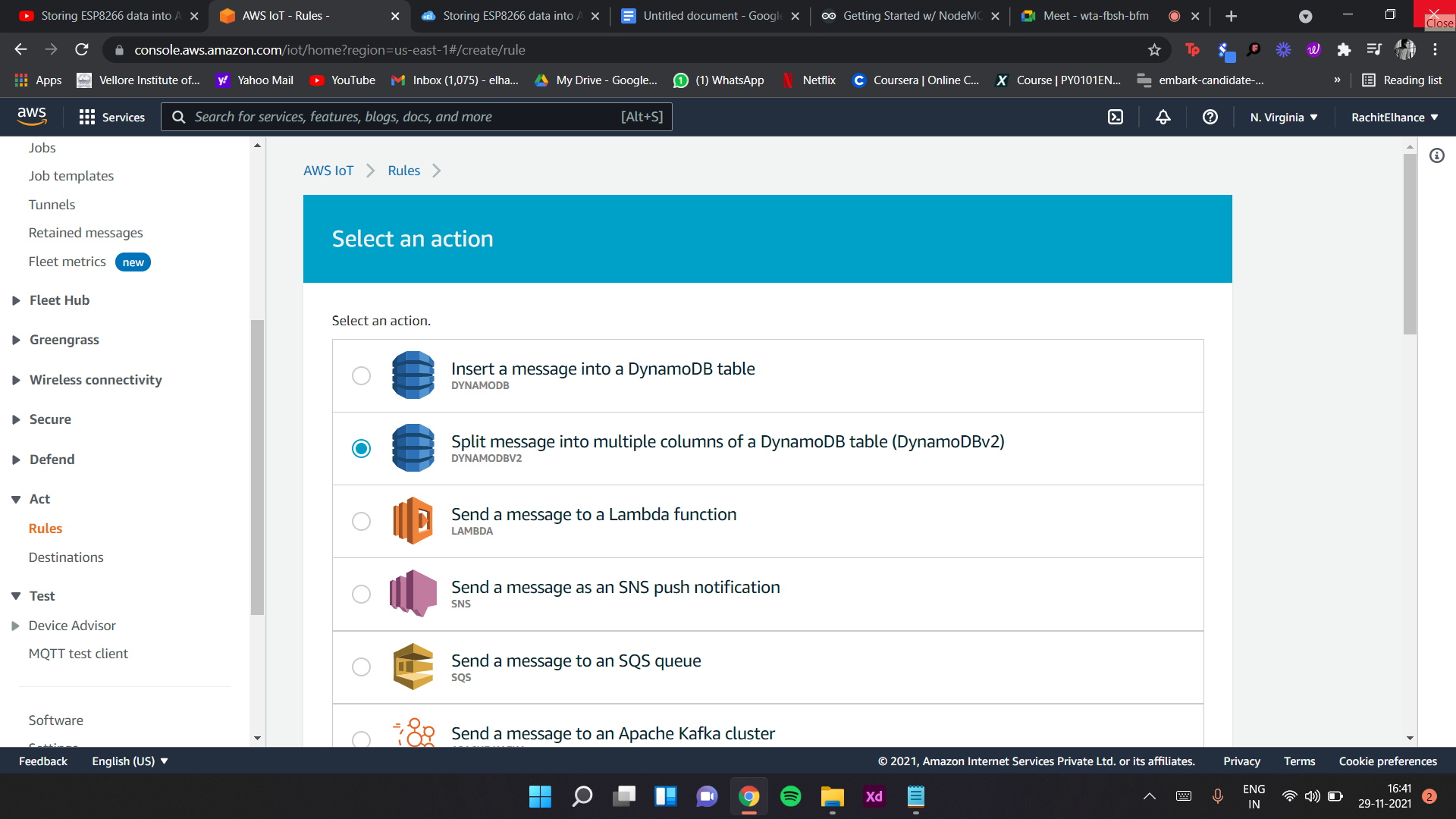


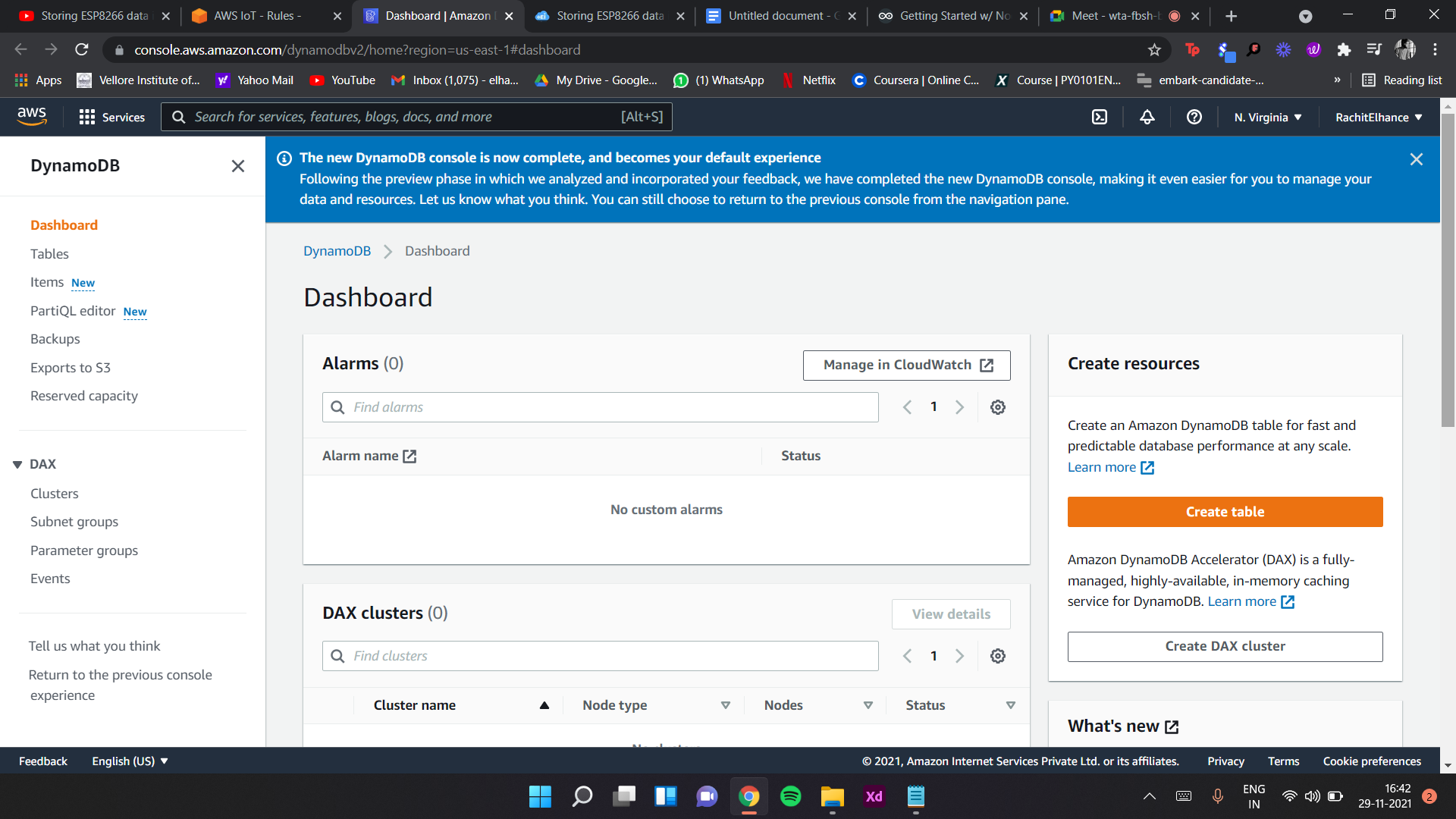


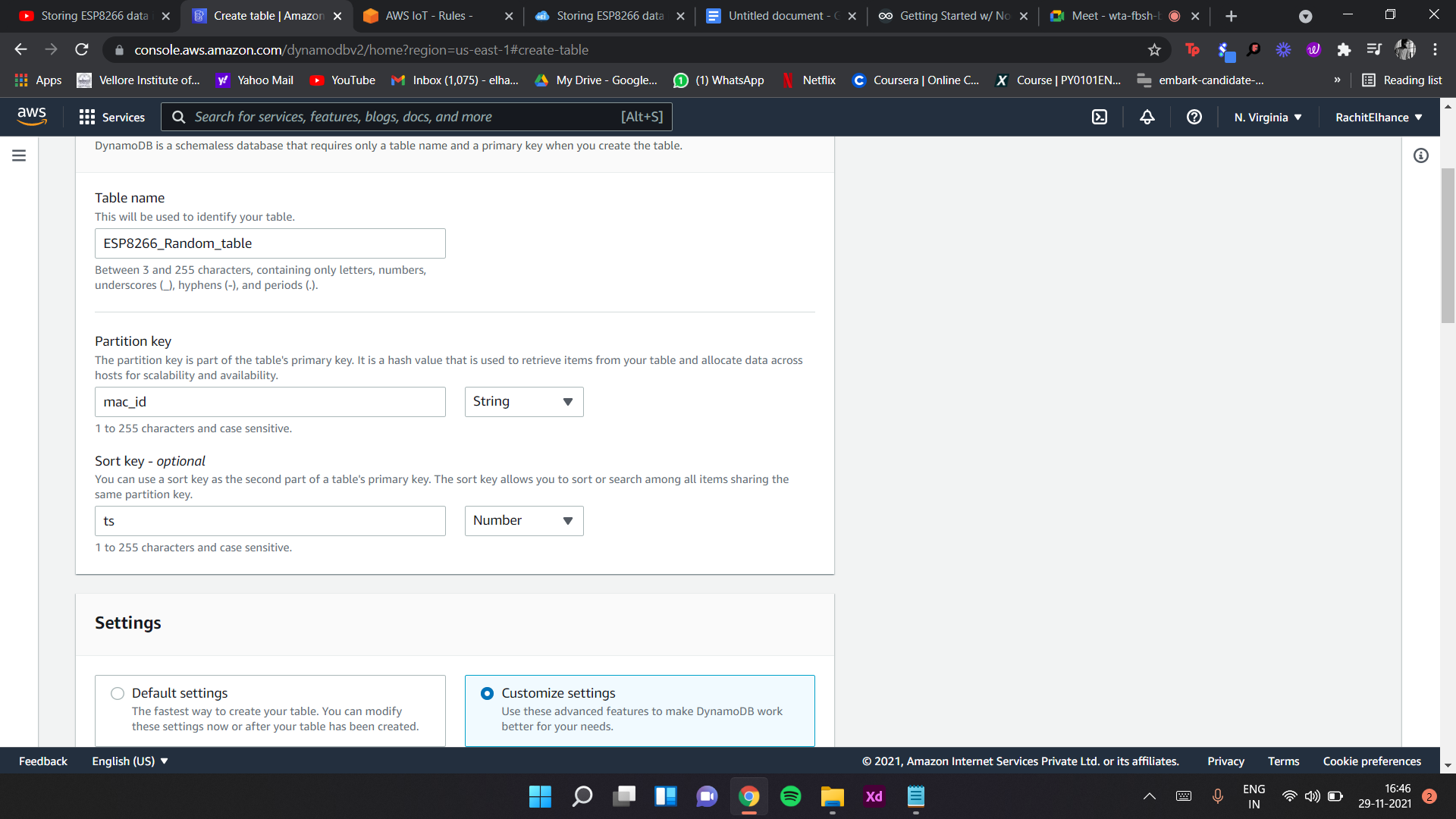


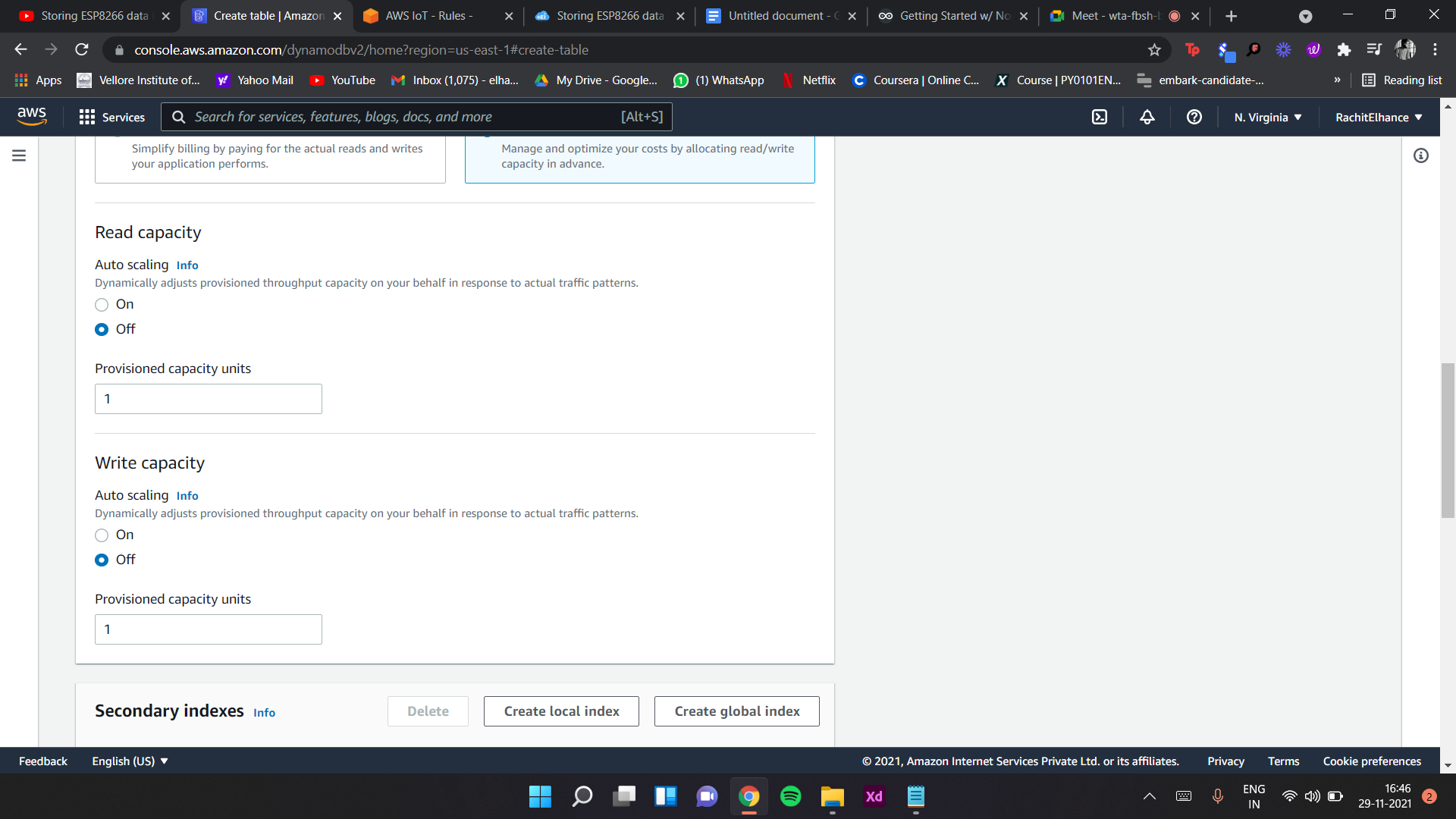


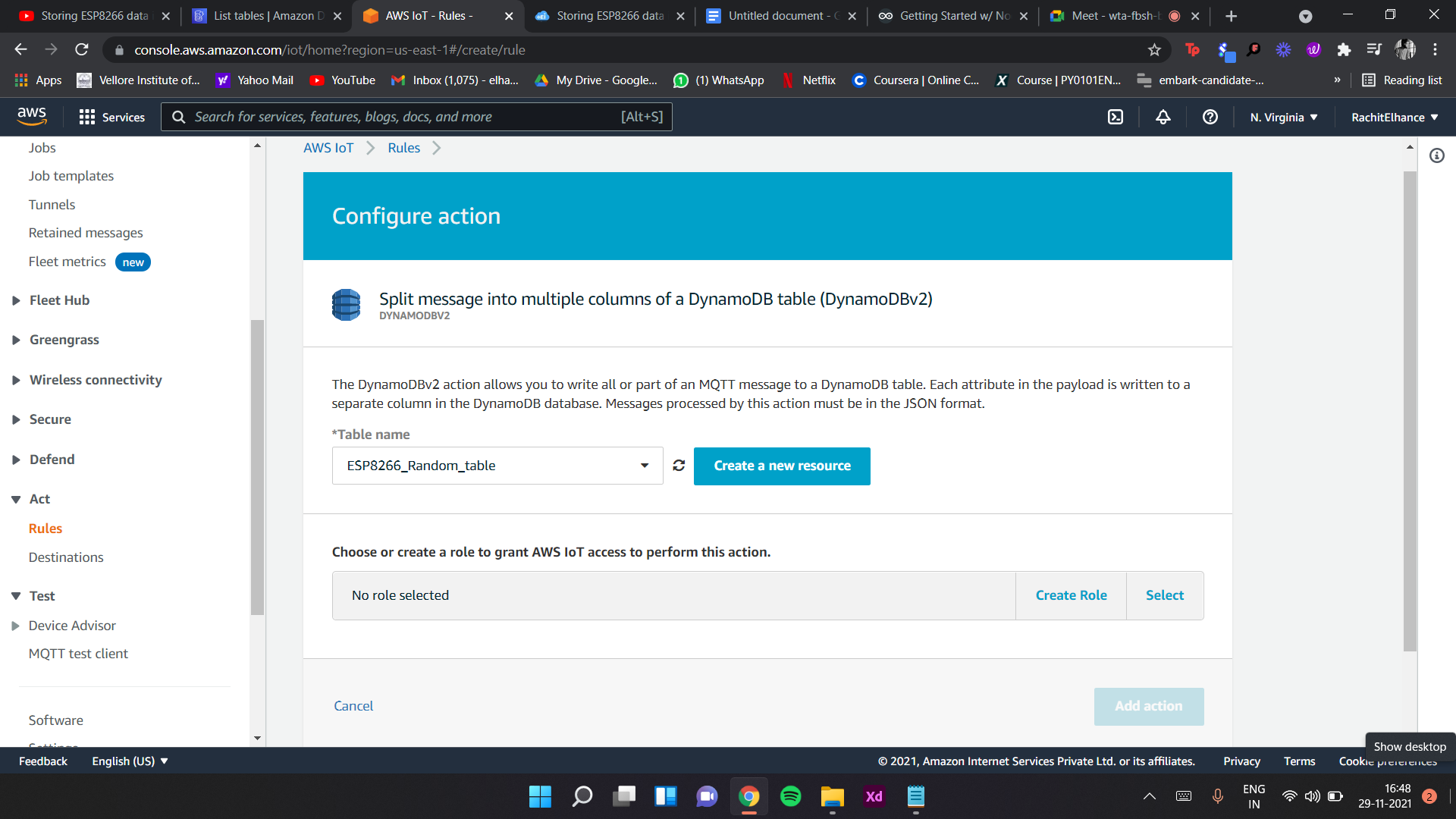


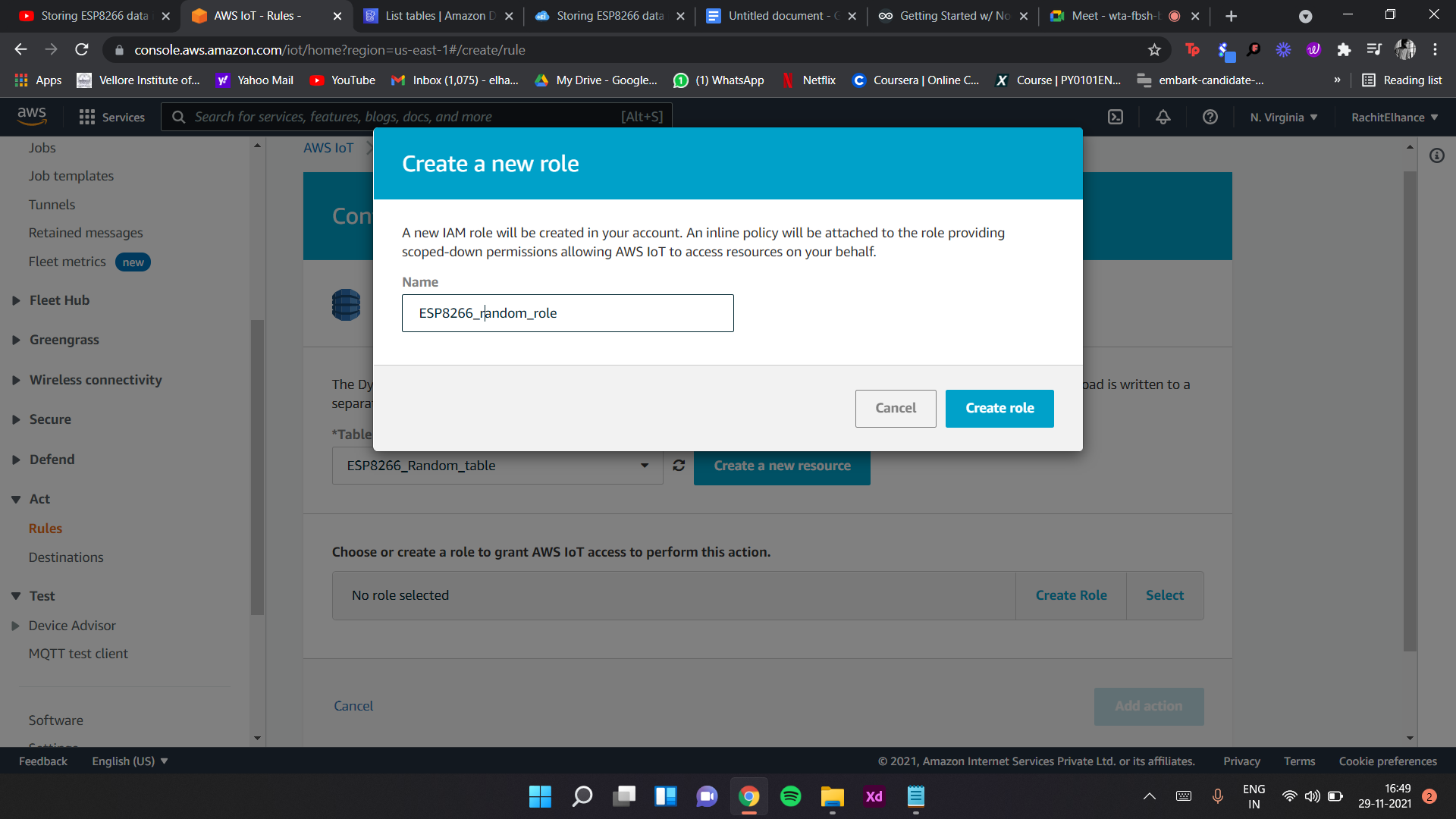












//Temperature

#include <DHT.h>

#include <DHT\_U.h>

#include <Hash.h>

#include <Adafruit\_Sensor.h>

//Gyroscope

#include <Wire.h>

//Air Quality Sensor

#include <MQ135.h>

#include <PubSubClient.h>

#include <FS.h>

#include <NTPClient.h>

#include <ESP8266WebServer.h>

#include <ESP8266WiFiGeneric.h>

#include <ESP8266WiFiGratuitous.h>

#include <ESP8266WiFiMulti.h>

#include <ESP8266WiFiScan.h>

#include <ESP8266WiFiSTA.h>

#include <ESP8266WiFiType.h>

#include <WiFiClient.h>

#include <WiFiClientSecure.h>

#include <WiFiClientSecureBearSSL.h>

#include <WiFiServer.h>

#include <WiFiServerSecure.h>

#include <WiFiServerSecureBearSSL.h>

#include <WiFiUdp.h>

String apiKey = "R2M4QUP18HXS0Q9P"; // Enter your Write API key from ThingSpeak

const char \*ssid = "Swayam OnePlus 7T"; // replace with your wifi ssid and wpa2 key

const char \*password = "swayam123";

const char\* things\_server = "api.thingspeak.com";

//Temperature and Humidity Sensor

const int dht\_pin = 16;

#define DHTTYPE DHT11 // DHT 11

// Initialize DHT sensor.

DHT dht(dht\_pin, DHTTYPE);

float temp;

float humid;

//Gyroscope

const int MPU\_addr = 0x68; // I2C address of the MPU-6050

int16\_t AcX, AcY, AcZ, Tmp, GyX, GyY, GyZ;

float ax = 0, ay = 0, az = 0, gx = 0, gy = 0, gz = 0;

boolean fall = false; //stores if a fall has occurred

boolean trigger1 = false; //stores if first trigger (lower threshold) has occurred

boolean trigger2 = false; //stores if second trigger (upper threshold) has occurred

boolean trigger3 = false; //stores if third trigger (orientation change) has occurred

byte trigger1count = 0; //stores the counts past since trigger 1 was set true

byte trigger2count = 0; //stores the counts past since trigger 2 was set true

byte trigger3count = 0; //stores the counts past since trigger 3 was set true

int angleChange = 0;

int Amp = 0;

//Ultrasonic Sensor

// defines pins numbers

const int trigPin = D5;

const int echoPin = D6;

// defines variables

long duration;

int distance;

//IR

const int ir\_pin = D7;

int ir\_val = 0;

//PIR

const int pir\_pin = D8;

long pir\_val = 0;

//Air Quality Sensor

const int air\_pin = A0; //z\_pin

float air\_quality = 0; //z\_acc

ESP8266WebServer server(80);

WiFiClient things\_client;

WiFiUDP ntpUDP;

NTPClient timeClient(ntpUDP, "pool.ntp.org");

const char\* AWS\_endpoint = "a1qdmh2g7a5jd8-ats.iot.us-east-1.amazonaws.com"; //MQTT broker ip

void callback(char\* topic, byte\* payload, unsigned int length) {

Serial.print("Message arrived [");

Serial.print(topic);

Serial.print("] ");

for (int i = 0; i < length; i++) {

Serial.print((char)payload[i]);

}

Serial.println();

}

WiFiClientSecure espClient;

PubSubClient client(AWS\_endpoint, 8883, callback, espClient); //set MQTT port number to 8883 as per //standard

long lastMsg = 0;

char msg[512];

int value = 0;

void setup\_wifi() {

delay(10);

// We start by connecting to a WiFi network

espClient.setBufferSizes(512, 512);

Serial.println();

Serial.print("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected");

Serial.println("IP address: ");

Serial.println(WiFi.localIP());

timeClient.begin();

while (!timeClient.update()) {

timeClient.forceUpdate();

}

espClient.setX509Time(timeClient.getEpochTime());

}

void reconnect() {

// Loop until we're reconnected

while (!client.connected()) {

Serial.print("Attempting MQTT connection...");

// Attempt to connect

if (client.connect("ESPthing")) {

Serial.println("connected");

// Once connected, publish an announcement...

client.publish("outTopic", "hello world");

// ... and resubscribe

client.subscribe("inTopic");

} else {

Serial.print("failed, rc=");

Serial.print(client.state());

Serial.println(" try again in 5 seconds");

char buf[256];

espClient.getLastSSLError(buf, 256);

Serial.print("WiFiClientSecure SSL error: ");

Serial.println(buf);

// Wait 5 seconds before retrying

delay(5000);

}

}

}

void setup() {

Serial.begin(115200);

dht.begin();

Wire.begin();

Serial.setDebugOutput(true);

// initialize digital pin LED\_BUILTIN as an output.

pinMode(LED\_BUILTIN, OUTPUT);

//Temperature and Humidity Sensor

pinMode(dht\_pin, INPUT);

//Gyroscope

Wire.beginTransmission(MPU\_addr);

Wire.write(0x6B); // PWR\_MGMT\_1 register

Wire.write(0); // set to zero (wakes up the MPU-6050)

Wire.endTransmission(true);

Serial.println("Wrote to IMU");

//Ultrasonic

pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output

pinMode(echoPin, INPUT); // Sets the echoPin as an Input

//IR

pinMode(ir\_pin, INPUT);

//PIR

pinMode(pir\_pin, INPUT);

setup\_wifi();

delay(1000);

if (!SPIFFS.begin()) {

Serial.println("Failed to mount file system");

return;

}

Serial.print("Heap: "); Serial.println(ESP.getFreeHeap());

// Load certificate file

File cert = SPIFFS.open("/cert.der", "r"); //replace cert.crt eith your uploaded file name

if (!cert) {

Serial.println("Failed to open cert file");

}

else

Serial.println("Success to open cert file");

delay(1000);

if (espClient.loadCertificate(cert))

Serial.println("cert loaded");

else

Serial.println("cert not loaded");

// Load private key file

File private\_key = SPIFFS.open("/private.der", "r"); //replace private eith your uploaded file name

if (!private\_key) {

Serial.println("Failed to open private cert file");

}

else

Serial.println("Success to open private cert file");

delay(1000);

if (espClient.loadPrivateKey(private\_key))

Serial.println("private key loaded");

else

Serial.println("private key not loaded");

// Load CA file

File ca = SPIFFS.open("/ca.der", "r"); //replace ca eith your uploaded file name

if (!ca) {

Serial.println("Failed to open ca ");

}

else

Serial.println("Success to open ca");

delay(1000);

if (espClient.loadCACert(ca))

Serial.println("ca loaded");

else

Serial.println("ca failed");

Serial.print("Heap: "); Serial.println(ESP.getFreeHeap());

server.on("/", handle\_OnConnect);

server.onNotFound(handle\_NotFound);

server.begin();

Serial.println("HTTP server started");

}

void loop() {

if(fall==true){

fall=false;

}

if (!client.connected()) {

reconnect();

}

client.loop();

long now = millis();

//Temperature and Humidity Sensor

float newT = dht.readTemperature();

if(!(isnan(newT))){

temp = newT;

}

float newH = dht.readHumidity();

if(!(isnan(newH))){

humid = newT;

}

// temp = dht.readTemperature(); // Gets the values of the temperature

// humid = dht.readHumidity(); // Gets the values of the humidity

//Ultrasonic

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 micro seconds

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds

duration = pulseIn(echoPin, HIGH);

// Calculating the distance

distance = duration \* 0.034 / 2;

//Gyroscope

mpu\_read();

ax = (AcX - 2050 - 0.88\*16384) / 16384.00;

ay = (AcY - 77 - 0.02\*16384) / 16384.00;

az = (AcZ - 1947 + 0.05\*16384) / 16384.00;

// Serial.print(ax);

// Serial.print("\t");

// Serial.print(ay);

// Serial.print("\t");

// Serial.print(az);

// Serial.print("\t");

gx = (GyX + 270) / 131.07;

gy = (GyY - 351) / 131.07;

gz = (GyZ + 136) / 131.07;

// calculating Amplitute vector for 3 axis

float Raw\_Amp = pow(pow(ax, 2) + pow(ay, 2) + pow(az, 2), 0.5);

Amp = Raw\_Amp \* 10; // Mulitiplied by 10 bcz values are between 0 to 1

// if (Amp <= 2 && trigger2 == false) { //if AM breaks lower threshold (0.4g)

// trigger1 = true;

//// Serial.println("TRIGGER 1 ACTIVATED");

// }

// if (trigger1 == true) {

// trigger1count++;

// if (Amp >= 12) { //if AM breaks upper threshold (3g)

// trigger2 = true;

//// Serial.println("TRIGGER 2 ACTIVATED");

// trigger1 = false; trigger1count = 0;

// }

// }

// if (trigger2 == true) {

// trigger2count++;

// angleChange = pow(pow(gx, 2) + pow(gy, 2) + pow(gz, 2), 0.5); Serial.println(angleChange);

// if (angleChange >= 30 && angleChange <= 400) { //if orientation changes by between 80-100 degrees

// trigger3 = true; trigger2 = false; trigger2count = 0;

//// Serial.println(angleChange);

//// Serial.println("TRIGGER 3 ACTIVATED");

// }

// }

// if (trigger3 == true) {

// trigger3count++;

// if (trigger3count >= 10) {

// angleChange = pow(pow(gx, 2) + pow(gy, 2) + pow(gz, 2), 0.5);

// //delay(10);

//// Serial.println(angleChange);

// if ((angleChange >= 0) && (angleChange <= 10)) { //if orientation changes remains between 0-10 degrees

// fall = true; trigger3 = false; trigger3count = 0;

//// Serial.println(angleChange);

// }

// else { //user regained normal orientation

// trigger3 = false; trigger3count = 0;

//// Serial.println("TRIGGER 3 DEACTIVATED");

// }

// }

// }

//// if (fall == true) { //in event of a fall detection

//// Serial.println("FALL DETECTED");

//// send\_event("fall\_detect");

//// fall = false;

//// }

// if (trigger2count >= 6) { //allow 0.5s for orientation change

// trigger2 = false; trigger2count = 0;

//// Serial.println("TRIGGER 2 DECACTIVATED");

// }

// if (trigger1count >= 6) { //allow 0.5s for AM to break upper threshold

// trigger1 = false; trigger1count = 0;

//// Serial.println("TRIGGER 1 DECACTIVATED");

// }

if(Amp > 2){

fall = 1;

}

//IR

ir\_val = digitalRead(ir\_pin);

if (ir\_val == 0) ir\_val = 1;

else ir\_val = 0;

//PIR

pir\_val = digitalRead(pir\_pin);

//Air Quality Sensor

MQ135 gasSensor = MQ135(air\_pin);

air\_quality = gasSensor.getPPM();

server.handleClient();

if (things\_client.connect(things\_server, 80)) // "184.106.153.149" or api.thingspeak.com

{

String postStr = apiKey;

postStr += "&field1=";

postStr += String(temp);

postStr += "&field2=";

postStr += String(humid);

postStr += "&field3=";

postStr += String(Amp);

postStr += "&field4=";

postStr += String(fall);

postStr += "&field5=";

postStr += String(distance);

postStr += "&field6=";

postStr += String(ir\_val);

postStr += "&field7=";

postStr += String(pir\_val);

postStr += "&field8=";

postStr += String(air\_quality);

postStr += "\r\n\r\n";

things\_client.print("POST /update HTTP/1.1\n");

things\_client.print("Host: api.thingspeak.com\n");

things\_client.print("Connection: close\n");

things\_client.print("X-THINGSPEAKAPIKEY: " + apiKey + "\n");

things\_client.print("Content-Type: application/x-www-form-urlencoded\n");

things\_client.print("Content-Length: ");

things\_client.print(postStr.length());

things\_client.print("\n\n");

things\_client.print(postStr);

Serial.print("\nTemperature: ");

Serial.print(temp);

Serial.print("\tHumidity: ");

Serial.print(humid);

Serial.print("\nAcceleration: ");

Serial.print(Amp);

Serial.print("\t\tFall: ");

Serial.print(fall);

Serial.print("\nDistance: ");

Serial.print(distance);

// Serial.print("\nZ Acceleration: ");

// Serial.print(air);

Serial.print("\nObject Detected: ");

Serial.print(ir\_val);

Serial.print("\tMotion Detected: ");

Serial.print(pir\_val);

Serial.print("\nAir Quality: ");

Serial.print(air\_quality);

Serial.println("\n\nSend to Thingspeak.");

}

things\_client.stop();

Serial.println("Waiting...\n");

// thingspeak needs minimum 15 sec delay between updates

delay(1000);

if (now - lastMsg > 2000) {

// Prints the distance on the Serial Monitor

lastMsg = now;

++value;

String s1 = "Hello Disha";

String s2 = "LA";

String falls = fall==1?"True":"False";

String objects = ir\_val>0?"True":"False";

String motions = pir\_val>0?"True":"False";

snprintf (msg, 512, "{\"Patient\_id\": \"%s\", \"ts\": %d, \"Temperature\": %f, \"Humidity\": %f, \"Amplitude\": %d, \"Fall\": \"%s\", \"Distance\": %d, \"Object Detected\": %d, \"Motion Detected\": %d, \"AirQuality\": %f}", s1.c\_str(), distance, temp, humid, Amp, falls.c\_str(), distance, objects.c\_str(), motions.c\_str(), air\_quality);

// snprintf (msg, 75, "{\"mac\_Id\" : \"%s\", \"random\_number\" : %d, \"random\_string\" : \"%s\"}", "macabc", distance, "LA");

Serial.print("Publish message: ");

Serial.println(msg);

client.publish("outTopic", msg);

// Serial.print("Heap: "); Serial.println(ESP.getFreeHeap()); //Low heap can cause problems

}

digitalWrite(LED\_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)

delay(100); // wait for a second

digitalWrite(LED\_BUILTIN, LOW); // turn the LED off by making the voltage LOW

delay(100); // wait for a second

}

void mpu\_read(){ //Gyroscope

Wire.beginTransmission(MPU\_addr);

Wire.write(0x3B); // starting with register 0x3B (ACCEL\_XOUT\_H)

Wire.endTransmission(false);

Wire.requestFrom(MPU\_addr,14,true); // request a total of 14 registers

AcX=Wire.read()<<8|Wire.read(); // 0x3B (ACCEL\_XOUT\_H) & 0x3C (ACCEL\_XOUT\_L)

AcY=Wire.read()<<8|Wire.read(); // 0x3D (ACCEL\_YOUT\_H) & 0x3E (ACCEL\_YOUT\_L)

AcZ=Wire.read()<<8|Wire.read(); // 0x3F (ACCEL\_ZOUT\_H) & 0x40 (ACCEL\_ZOUT\_L)

Tmp=Wire.read()<<8|Wire.read(); // 0x41 (TEMP\_OUT\_H) & 0x42 (TEMP\_OUT\_L)

GyX=Wire.read()<<8|Wire.read(); // 0x43 (GYRO\_XOUT\_H) & 0x44 (GYRO\_XOUT\_L)

GyY=Wire.read()<<8|Wire.read(); // 0x45 (GYRO\_YOUT\_H) & 0x46 (GYRO\_YOUT\_L)

GyZ=Wire.read()<<8|Wire.read(); // 0x47 (GYRO\_ZOUT\_H) & 0x48 (GYRO\_ZOUT\_L)

}

void handle\_OnConnect() {

server.send(200, "text/html", SendHTML(temp, humid, Amp, fall, distance, ir\_val, pir\_val, air\_quality));

}

void handle\_NotFound() {

server.send(404, "text/plain", "Not found");

}

String SendHTML(float temp, float humid, int Amp, boolean fall, int distance, int ir\_val, long pir\_val, float air\_quality) {

String ptr = "<!DOCTYPE html>";

ptr += "<html>";

ptr += "<head>";

ptr += "<title>ESP32 Patient Health Monitoring</title>";

ptr += "<meta name='viewport' content='width=device-width, initial-scale=1.0'>";

ptr += "<link href='https://fonts.googleapis.com/css?family=Open+Sans:300,400,600' rel='stylesheet'>";

ptr += "<style>";

ptr += "html { font-family: 'Open Sans', sans-serif; display: block; margin: 0px auto; text-align: center;color: #444444;}";

ptr += "body{margin: 0px;} ";

ptr += "h1 {margin: 50px auto 30px;} ";

ptr += ".side-by-side{display: table-cell;vertical-align: middle;position: relative;}";

ptr += ".text{font-weight: 600;font-size: 19px;width: 200px;}";

ptr += ".reading{font-weight: 300;font-size: 50px;padding-right: 25px;}";

ptr += ".tem .reading{color: #FF0000;}";

ptr += ".hum .reading{color: #0000FF;}";

ptr += ".dist .reading{color: #F29C1F;}";

ptr += ".acc .reading{color: #00FF00;}";

ptr += ".fal .reading{color: #FF0000;}";

ptr += ".irval .reading{color: #FF0000;}";

ptr += ".pirval .reading{color: #955BA5;}";

ptr += ".airq .reading{color: #777777;}";

/\*ptr +=".bodytemperature .reading{color: #F29C1F;}";\*/

ptr += ".superscript{font-size: 17px;font-weight: 600;position: absolute;top: 10px;}";

ptr += ".data{padding: 10px;}";

ptr += ".container{display: table;margin: 0 auto;}";

ptr += ".icon{width:65px}";

ptr += "</style>";

ptr += "</head>";

ptr += "<body>";

ptr += "<h1>ESP32 Patient Health Monitoring</h1>";

ptr += "<h3>Disha Swayam Rahul Rachit</h3>";

ptr += "<div class='container'>";

ptr += "<div class='data tem'>";

ptr += "<div class='side-by-side icon'>";

// ptr += "<svg enable-background='new 0 0 19.438 54.003'height=54.003px id=Layer\_1 version=1.1 viewBox='0 0 19.438 54.003'width=19.438px x=0px xml:space=preserve xmlns=http://www.w3.org/2000/svg xmlns:xlink=http://www.w3.org/1999/xlink y=0px><g><path d='M11.976,8.82v-2h4.084V6.063C16.06,2.715,13.345,0,9.996,0H9.313C5.965,0,3.252,2.715,3.252,6.063v30.982";

// ptr += "C1.261,38.825,0,41.403,0,44.286c0,5.367,4.351,9.718,9.719,9.718c5.368,0,9.719-4.351,9.719-9.718";

// ptr += "c0-2.943-1.312-5.574-3.378-7.355V18.436h-3.914v-2h3.914v-2.808h-4.084v-2h4.084V8.82H11.976z M15.302,44.833";

// ptr += "c0,3.083-2.5,5.583-5.583,5.583s-5.583-2.5-5.583-5.583c0-2.279,1.368-4.236,3.326-5.104V24.257C7.462,23.01,8.472,22,9.719,22";

// ptr += "s2.257,1.01,2.257,2.257V39.73C13.934,40.597,15.302,42.554,15.302,44.833z'fill=#F29C21 /></g></svg>";

ptr += "</div>";

ptr += "<div class='side-by-side text'>Temperature</div>";

ptr += "<div class='side-by-side reading'>";

ptr += (int)temp;

ptr += "<span class='superscript'>&deg;C</span></div>";

ptr += "</div>";

ptr += "<div class='data hum'>";

ptr += "<div class='side-by-side icon'>";

// ptr += "<svg enable-background='new 0 0 19.438 54.003'height=54.003px id=Layer\_1 version=1.1 viewBox='0 0 19.438 54.003'width=19.438px x=0px xml:space=preserve xmlns=http://www.w3.org/2000/svg xmlns:xlink=http://www.w3.org/1999/xlink y=0px><g><path d='M11.976,8.82v-2h4.084V6.063C16.06,2.715,13.345,0,9.996,0H9.313C5.965,0,3.252,2.715,3.252,6.063v30.982";

// ptr += "C1.261,38.825,0,41.403,0,44.286c0,5.367,4.351,9.718,9.719,9.718c5.368,0,9.719-4.351,9.719-9.718";

// ptr += "c0-2.943-1.312-5.574-3.378-7.355V18.436h-3.914v-2h3.914v-2.808h-4.084v-2h4.084V8.82H11.976z M15.302,44.833";

// ptr += "c0,3.083-2.5,5.583-5.583,5.583s-5.583-2.5-5.583-5.583c0-2.279,1.368-4.236,3.326-5.104V24.257C7.462,23.01,8.472,22,9.719,22";

// ptr += "s2.257,1.01,2.257,2.257V39.73C13.934,40.597,15.302,42.554,15.302,44.833z'fill=#F29C21 /></g></svg>";

ptr += "</div>";

ptr += "<div class='side-by-side text'>Humidity</div>";

ptr += "<div class='side-by-side reading'>";

ptr += (int)humid;

ptr += "<span>ppt</span></div>";

ptr += "</div>";

ptr += "<div class='data dist'>";

ptr += "<div class='side-by-side icon'>";

// ptr += "<svg enable-background='new 0 0 19.438 54.003'height=54.003px id=Layer\_1 version=1.1 viewBox='0 0 19.438 54.003'width=19.438px x=0px xml:space=preserve xmlns=http://www.w3.org/2000/svg xmlns:xlink=http://www.w3.org/1999/xlink y=0px><g><path d='M11.976,8.82v-2h4.084V6.063C16.06,2.715,13.345,0,9.996,0H9.313C5.965,0,3.252,2.715,3.252,6.063v30.982";

// ptr += "C1.261,38.825,0,41.403,0,44.286c0,5.367,4.351,9.718,9.719,9.718c5.368,0,9.719-4.351,9.719-9.718";

// ptr += "c0-2.943-1.312-5.574-3.378-7.355V18.436h-3.914v-2h3.914v-2.808h-4.084v-2h4.084V8.82H11.976z M15.302,44.833";

// ptr += "c0,3.083-2.5,5.583-5.583,5.583s-5.583-2.5-5.583-5.583c0-2.279,1.368-4.236,3.326-5.104V24.257C7.462,23.01,8.472,22,9.719,22";

// ptr += "s2.257,1.01,2.257,2.257V39.73C13.934,40.597,15.302,42.554,15.302,44.833z'fill=#F29C21 /></g></svg>";

ptr += "</div>";

ptr += "<div class='side-by-side text'>Distance</div>";

ptr += "<div class='side-by-side reading'>";

ptr += (int)distance;

ptr += "<span>cm</span></div>";

ptr += "</div>";

ptr += "<div class='data acc'>";

ptr += "<div class='side-by-side icon'>";

// ptr += "<svg enable-background='new 0 0 19.438 54.003'height=54.003px id=Layer\_1 version=1.1 viewBox='0 0 19.438 54.003'width=19.438px x=0px xml:space=preserve xmlns=http://www.w3.org/2000/svg xmlns:xlink=http://www.w3.org/1999/xlink y=0px><g><path d='M11.976,8.82v-2h4.084V6.063C16.06,2.715,13.345,0,9.996,0H9.313C5.965,0,3.252,2.715,3.252,6.063v30.982";

// ptr += "C1.261,38.825,0,41.403,0,44.286c0,5.367,4.351,9.718,9.719,9.718c5.368,0,9.719-4.351,9.719-9.718";

// ptr += "c0-2.943-1.312-5.574-3.378-7.355V18.436h-3.914v-2h3.914v-2.808h-4.084v-2h4.084V8.82H11.976z M15.302,44.833";

// ptr += "c0,3.083-2.5,5.583-5.583,5.583s-5.583-2.5-5.583-5.583c0-2.279,1.368-4.236,3.326-5.104V24.257C7.462,23.01,8.472,22,9.719,22";

// ptr += "s2.257,1.01,2.257,2.257V39.73C13.934,40.597,15.302,42.554,15.302,44.833z'fill=#F29C21 /></g></svg>";

ptr += "</div>";

ptr += "<div class='side-by-side text'>Acceleration Amplitude</div>";

ptr += "<div class='side-by-side reading'>";

ptr += (int)Amp;

ptr += "<span>m/s<span class='superscript'>2</span></span></div>";

ptr += "</div>";

ptr += "<div class='data fal'>";

ptr += "<div class='side-by-side icon'>";

// ptr += "<svg enable-background='new 0 0 19.438 54.003'height=54.003px id=Layer\_1 version=1.1 viewBox='0 0 19.438 54.003'width=19.438px x=0px xml:space=preserve xmlns=http://www.w3.org/2000/svg xmlns:xlink=http://www.w3.org/1999/xlink y=0px><g><path d='M11.976,8.82v-2h4.084V6.063C16.06,2.715,13.345,0,9.996,0H9.313C5.965,0,3.252,2.715,3.252,6.063v30.982";

// ptr += "C1.261,38.825,0,41.403,0,44.286c0,5.367,4.351,9.718,9.719,9.718c5.368,0,9.719-4.351,9.719-9.718";

// ptr += "c0-2.943-1.312-5.574-3.378-7.355V18.436h-3.914v-2h3.914v-2.808h-4.084v-2h4.084V8.82H11.976z M15.302,44.833";

// ptr += "c0,3.083-2.5,5.583-5.583,5.583s-5.583-2.5-5.583-5.583c0-2.279,1.368-4.236,3.326-5.104V24.257C7.462,23.01,8.472,22,9.719,22";

// ptr += "s2.257,1.01,2.257,2.257V39.73C13.934,40.597,15.302,42.554,15.302,44.833z'fill=#F29C21 /></g></svg>";

ptr += "</div>";

ptr += "<div class='side-by-side text'>Fall</div>";

ptr += "<div class='side-by-side reading'>";

ptr += fall;

/\*ptr += "<span>cm</span></div>";\*/

ptr += "</div>";

// ptr += "<div class='data zacc'>";

// ptr += "<div class='side-by-side icon'>";

// ptr += "<svg enable-background='new 0 0 29.235 40.64'height=40.64px id=Layer\_1 version=1.1 viewBox='0 0 29.235 40.64'width=29.235px x=0px xml:space=preserve xmlns=http://www.w3.org/2000/svg xmlns:xlink=http://www.w3.org/1999/xlink y=0px><path d='M14.618,0C14.618,0,0,17.95,0,26.022C0,34.096,6.544,40.64,14.618,40.64s14.617-6.544,14.617-14.617";

// ptr += "C29.235,17.95,14.618,0,14.618,0z M13.667,37.135c-5.604,0-10.162-4.56-10.162-10.162c0-0.787,0.638-1.426,1.426-1.426";

// ptr += "c0.787,0,1.425,0.639,1.425,1.426c0,4.031,3.28,7.312,7.311,7.312c0.787,0,1.425,0.638,1.425,1.425";

// ptr += "C15.093,36.497,14.455,37.135,13.667,37.135z'fill=#3C97D3 /></svg>";

// ptr += "</div>";

// ptr += "<div class='side-by-side text'>Z Acceleration</div>";

// ptr += "<div class='side-by-side reading'>";

// ptr += (int)air;

// ptr += "<span>m/s<span class='superscript'>2</span></span></div>";

// ptr += "</div>";

ptr += "<div class='data irval'>";

ptr += "<div class='side-by-side icon'>";

// ptr += "<svg enable-background='new 0 0 40.542 40.541'height=40.541px id=Layer\_1 version=1.1 viewBox='0 0 40.542 40.541'width=40.542px x=0px xml:space=preserve xmlns=http://www.w3.org/2000/svg xmlns:xlink=http://www.w3.org/1999/xlink y=0px><g><path d='M34.313,20.271c0-0.552,0.447-1,1-1h5.178c-0.236-4.841-2.163-9.228-5.214-12.593l-3.425,3.424";

// ptr += "c-0.195,0.195-0.451,0.293-0.707,0.293s-0.512-0.098-0.707-0.293c-0.391-0.391-0.391-1.023,0-1.414l3.425-3.424";

// ptr += "c-3.375-3.059-7.776-4.987-12.634-5.215c0.015,0.067,0.041,0.13,0.041,0.202v4.687c0,0.552-0.447,1-1,1s-1-0.448-1-1V0.25";

// ptr += "c0-0.071,0.026-0.134,0.041-0.202C14.39,0.279,9.936,2.256,6.544,5.385l3.576,3.577c0.391,0.391,0.391,1.024,0,1.414";

// ptr += "c-0.195,0.195-0.451,0.293-0.707,0.293s-0.512-0.098-0.707-0.293L5.142,6.812c-2.98,3.348-4.858,7.682-5.092,12.459h4.804";

// ptr += "c0.552,0,1,0.448,1,1s-0.448,1-1,1H0.05c0.525,10.728,9.362,19.271,20.22,19.271c10.857,0,19.696-8.543,20.22-19.271h-5.178";

// ptr += "C34.76,21.271,34.313,20.823,34.313,20.271z M23.084,22.037c-0.559,1.561-2.274,2.372-3.833,1.814";

// ptr += "c-1.561-0.557-2.373-2.272-1.815-3.833c0.372-1.041,1.263-1.737,2.277-1.928L25.2,7.202L22.497,19.05";

// ptr += "C23.196,19.843,23.464,20.973,23.084,22.037z'fill=#26B999 /></g></svg>";

ptr += "</div>";

ptr += "<div class='side-by-side text'>Object Detected</div>";

ptr += "<div class='side-by-side reading'>";

ptr += (int)ir\_val;

/\*ptr +="<span class='superscript'>BPM</span></div>";\*/

ptr += "</div>";

ptr += "<div class='data pirval'>";

ptr += "<div class='side-by-side icon'>";

// ptr += "<svg enable-background='new 0 0 58.422 40.639'height=40.639px id=Layer\_1 version=1.1 viewBox='0 0 58.422 40.639'width=58.422px x=0px xml:space=preserve xmlns=http://www.w3.org/2000/svg xmlns:xlink=http://www.w3.org/1999/xlink y=0px><g><path d='M58.203,37.754l0.007-0.004L42.09,9.935l-0.001,0.001c-0.356-0.543-0.969-0.902-1.667-0.902";

// ptr += "c-0.655,0-1.231,0.32-1.595,0.808l-0.011-0.007l-0.039,0.067c-0.021,0.03-0.035,0.063-0.054,0.094L22.78,37.692l0.008,0.004";

// ptr += "c-0.149,0.28-0.242,0.594-0.242,0.934c0,1.102,0.894,1.995,1.994,1.995v0.015h31.888c1.101,0,1.994-0.893,1.994-1.994";

// ptr += "C58.422,38.323,58.339,38.024,58.203,37.754z'fill=#955BA5 /><path d='M19.704,38.674l-0.013-0.004l13.544-23.522L25.13,1.156l-0.002,0.001C24.671,0.459,23.885,0,22.985,0";

// ptr += "c-0.84,0-1.582,0.41-2.051,1.038l-0.016-0.01L20.87,1.114c-0.025,0.039-0.046,0.082-0.068,0.124L0.299,36.851l0.013,0.004";

// ptr += "C0.117,37.215,0,37.62,0,38.059c0,1.412,1.147,2.565,2.565,2.565v0.015h16.989c-0.091-0.256-0.149-0.526-0.149-0.813";

// ptr += "C19.405,39.407,19.518,39.019,19.704,38.674z'fill=#955BA5 /></g></svg>";

ptr += "</div>";

ptr += "<div class='side-by-side text'>Motion Detected</div>";

ptr += "<div class='side-by-side reading'>";

ptr += (int)pir\_val;

/\*ptr +="<span class='superscript'>%</span></div>";\*/

ptr += "</div>";

ptr += "<div class='data airq'>";

ptr += "<div class='side-by-side icon'>";

// ptr += "<svg enable-background='new 0 0 58.422 40.639'height=40.639px id=Layer\_1 version=1.1 viewBox='0 0 58.422 40.639'width=58.422px x=0px xml:space=preserve xmlns=http://www.w3.org/2000/svg xmlns:xlink=http://www.w3.org/1999/xlink y=0px><g><path d='M58.203,37.754l0.007-0.004L42.09,9.935l-0.001,0.001c-0.356-0.543-0.969-0.902-1.667-0.902";

// ptr += "c-0.655,0-1.231,0.32-1.595,0.808l-0.011-0.007l-0.039,0.067c-0.021,0.03-0.035,0.063-0.054,0.094L22.78,37.692l0.008,0.004";

// ptr += "c-0.149,0.28-0.242,0.594-0.242,0.934c0,1.102,0.894,1.995,1.994,1.995v0.015h31.888c1.101,0,1.994-0.893,1.994-1.994";

// ptr += "C58.422,38.323,58.339,38.024,58.203,37.754z'fill=#955BA5 /><path d='M19.704,38.674l-0.013-0.004l13.544-23.522L25.13,1.156l-0.002,0.001C24.671,0.459,23.885,0,22.985,0";

// ptr += "c-0.84,0-1.582,0.41-2.051,1.038l-0.016-0.01L20.87,1.114c-0.025,0.039-0.046,0.082-0.068,0.124L0.299,36.851l0.013,0.004";

// ptr += "C0.117,37.215,0,37.62,0,38.059c0,1.412,1.147,2.565,2.565,2.565v0.015h16.989c-0.091-0.256-0.149-0.526-0.149-0.813";

// ptr += "C19.405,39.407,19.518,39.019,19.704,38.674z'fill=#955BA5 /></g></svg>";

ptr += "</div>";

ptr += "<div class='side-by-side text'>Air Quality</div>";

ptr += "<div class='side-by-side reading'>";

ptr += (int)air\_quality;

ptr += "<span>ppm</span></div>";

ptr += "</div>";

// ptr +="<div class='data Body Temperature'>";

// ptr +="<div class='side-by-side icon'>";

// ptr +="<svg enable-background='new 0 0 19.438 54.003'height=54.003px id=Layer\_1 version=1.1 viewBox='0 0 19.438 54.003'width=19.438px x=0px xml:space=preserve xmlns=http://www.w3.org/2000/svg xmlns:xlink=http://www.w3.org/1999/xlink y=0px><g><path d='M11.976,8.82v-2h4.084V6.063C16.06,2.715,13.345,0,9.996,0H9.313C5.965,0,3.252,2.715,3.252,6.063v30.982";

// ptr +="C1.261,38.825,0,41.403,0,44.286c0,5.367,4.351,9.718,9.719,9.718c5.368,0,9.719-4.351,9.719-9.718";

// ptr +="c0-2.943-1.312-5.574-3.378-7.355V18.436h-3.914v-2h3.914v-2.808h-4.084v-2h4.084V8.82H11.976z M15.302,44.833";

// ptr +="c0,3.083-2.5,5.583-5.583,5.583s-5.583-2.5-5.583-5.583c0-2.279,1.368-4.236,3.326-5.104V24.257C7.462,23.01,8.472,22,9.719,22";

// ptr +="s2.257,1.01,2.257,2.257V39.73C13.934,40.597,15.302,42.554,15.302,44.833z'fill=#F29C21 /></g></svg>";

// ptr +="</div>";

// ptr +="<div class='side-by-side text'>Body Temperature</div>";

// ptr +="<div class='side-by-side reading'>";

// ptr +=(int)bodytemperature;

// ptr +="<span class='superscript'>&deg;C</span></div>";

// ptr +="</div>";

ptr += "</div>";

ptr += "</body>";

ptr += "</html>";

return ptr;

}