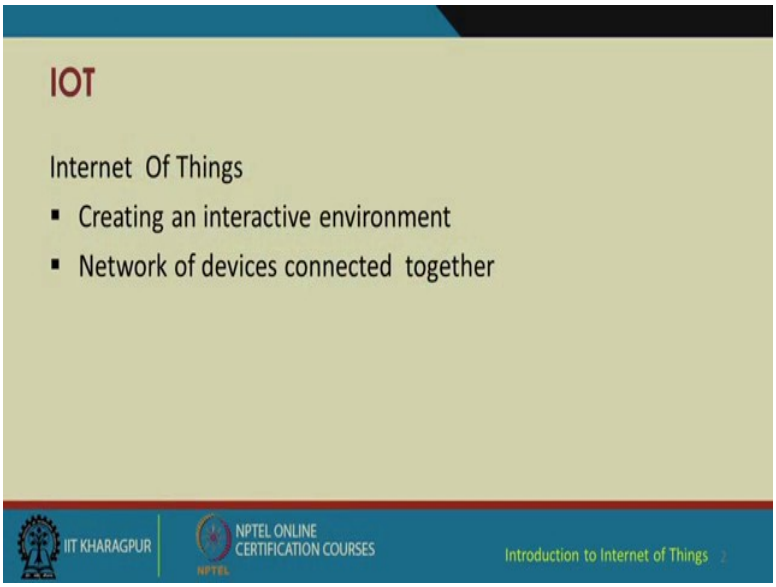
Introduction to Internet of Things
Prof. Sudip Misra
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 32
Implementation of IoT with Raspberry Pi– III

Hello in this third part of this implementation of IoT with raspberry pi. Will be covering the topics we discussed in the previous two lectures that is integration of sensor with a raspberry pi, then in the previous lecture that was integration of sensor the raspberry pi and making the raspberry pi configuring the raspberry pi as a client device, and we configure another desktop pc as a server device, and we initiate communication between this client and server and we also initiate storage of data.

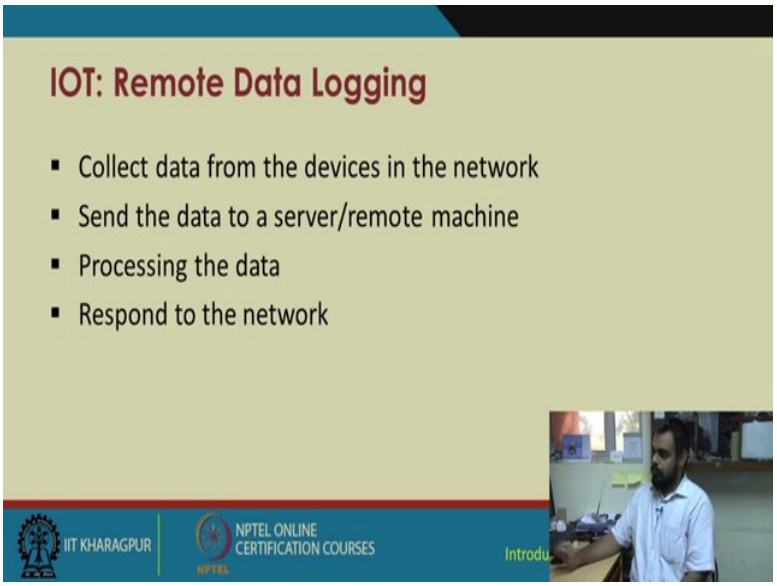
When the data is being uploaded on to the server, and thirdly in this part we will be focusing besides those previous applications will be focusing more on the server side that is once the data has arrived what exactly can be done with the data.

(Refer Slide Time: 01:18)



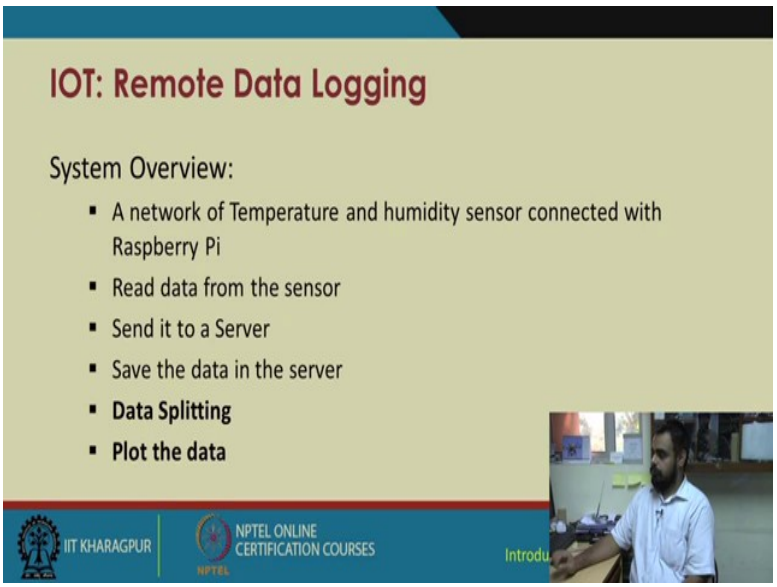
So, as we have covered previously we are mainly focusing on creating an interactive environment using a network of connected devices, and the data logging being done is remote which is connected to a network.

(Refer Slide Time: 01:25)



And the data may be processed on the client side as well as it may be processed on the server side itself.

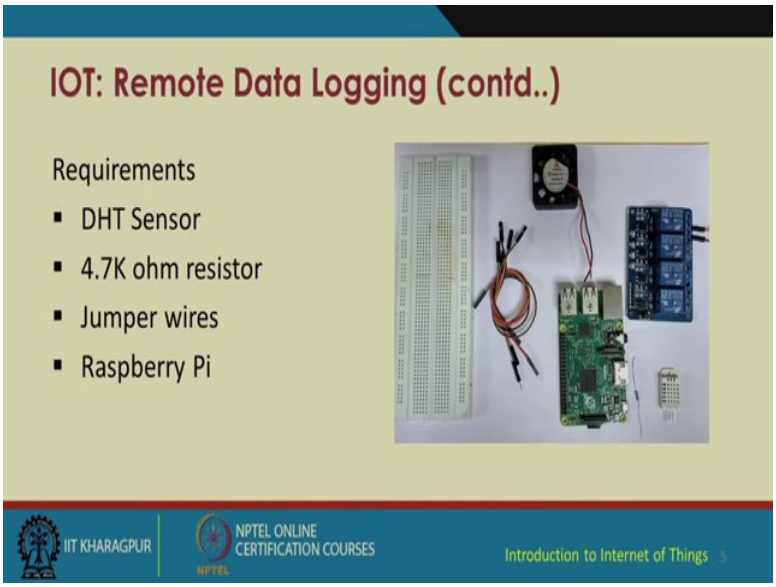
(Refer Slide Time: 01:42)



Again going through the system overview, as you can see we will first start off with a network of temperature and humidity sensors connected with raspberry pi board, for this demonstration we have connected only one sensor to a single raspberry pi we read data from that sensor we send it to a remote server over the network received the data at the server.

Then till this part we covered in our previous lectures over here we will be doing all these things, and after that we will be splitting the data which we saved in the data logs and will try to create a running plot of the data which is coming into the server.

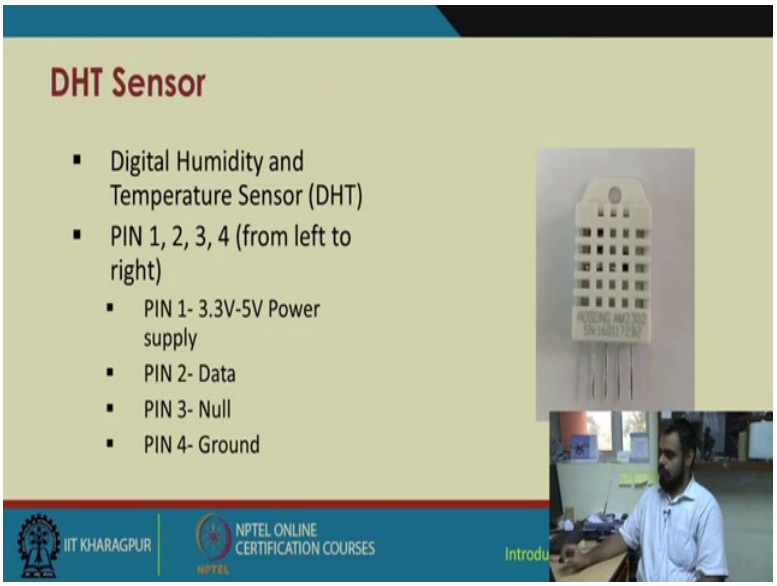
(Refer Slide Time: 02:32)



So, imagine regarding this running plot imagine your devices you have a network of devices spread over the city, and which are uploading data to a remote server and instead of running the script on a local server.

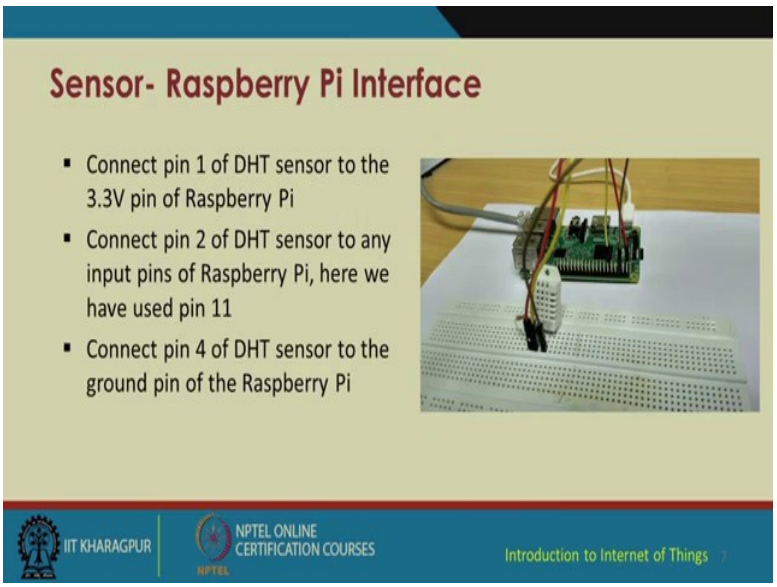
If this can be hosted on the internet, you will be able to access your sensor readings sensor data you may be able to remotely actuator sensors or actuators or IoT node from any part of the world and you may be able to visualize the changes in the environment in a particular area to which the node is deployed by accessing the whole system through the internet or through a browser.

(Refer Slide Time: 03:28)



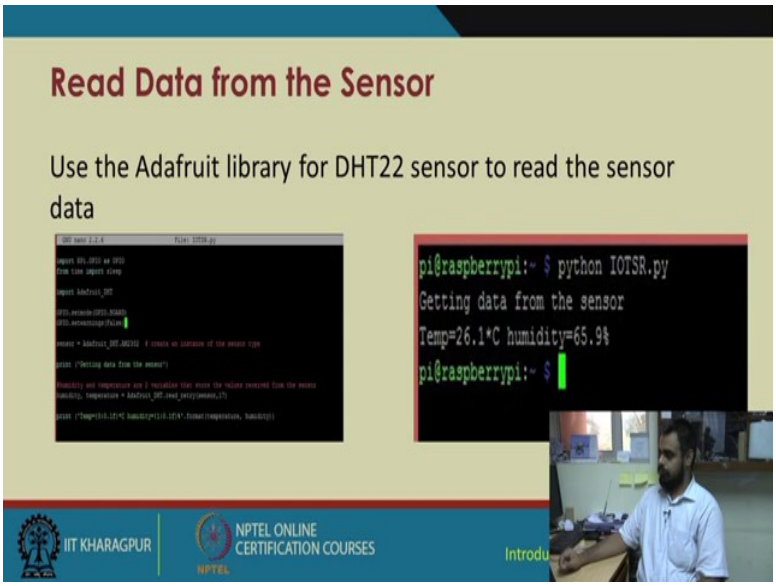
So, again back to this topic, for this remote data logging we will be using the DHT sensor and raspberry pi the pin configuration pin one is connected to power pin four to ground and pin two is connected to the data pin. The raspberry pi interface is the same as before only changes pin 2 of DHT is connected to pin 11 of the raspberry pi.

(Refer Slide Time: 03:38)



So, we already have the adafruit python library for this DHT sensor

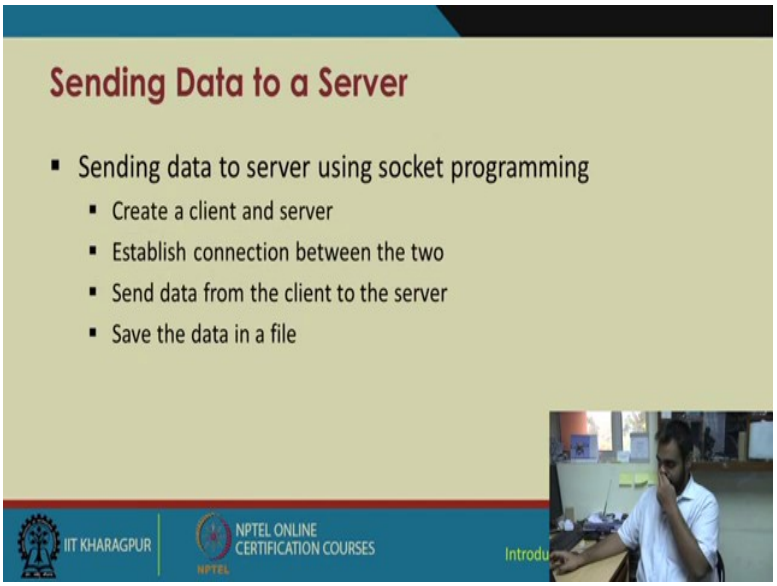
(Refer Slide Time: 03:50)



And we have all the libraries installed as covered in the previous lectures, the code is also similar to the previous lectures.

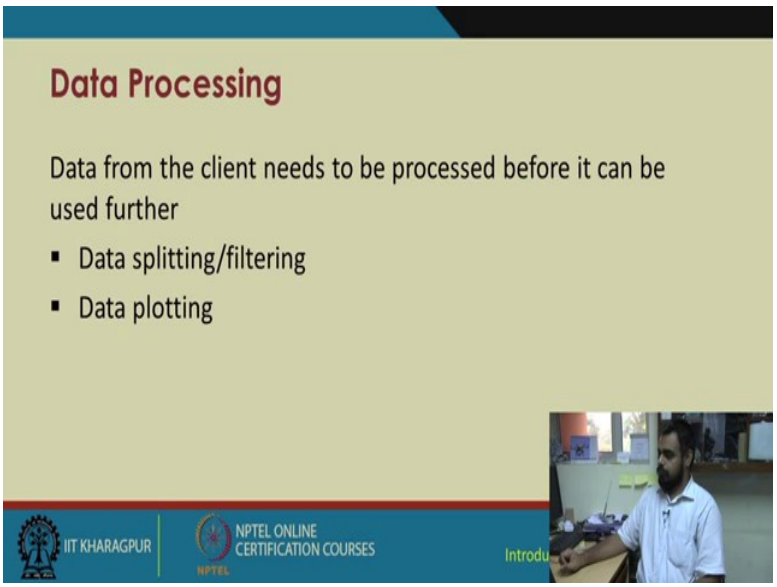
So, on the left hand side you can see this is a screenshot for that IOTSR.py file, and on the right hand side will be getting the output. Once this file is run on the raspberry pi you must remember this IOTSR.py file is kept on the raspberry pi and when you remotely run this file it will generate a temperature and humidity value. So, this will be on the client device itself sorry this will be on the sensor device itself which is the raspberry pi we cannot call it a client yet, because it has not been connected to the network as of now.

(Refer Slide Time: 04:47)



Now, when we attempt to connect this over the network, we create client server architecture and establish connection between one client device and a desktop which acts as the server device, and once the data transmission from the client initiates successive data will be in coming into the server and get stored into a log file.

(Refer Slide Time: 05:16)



Now till the previous slide we had covered all these things in details in lecture 2.

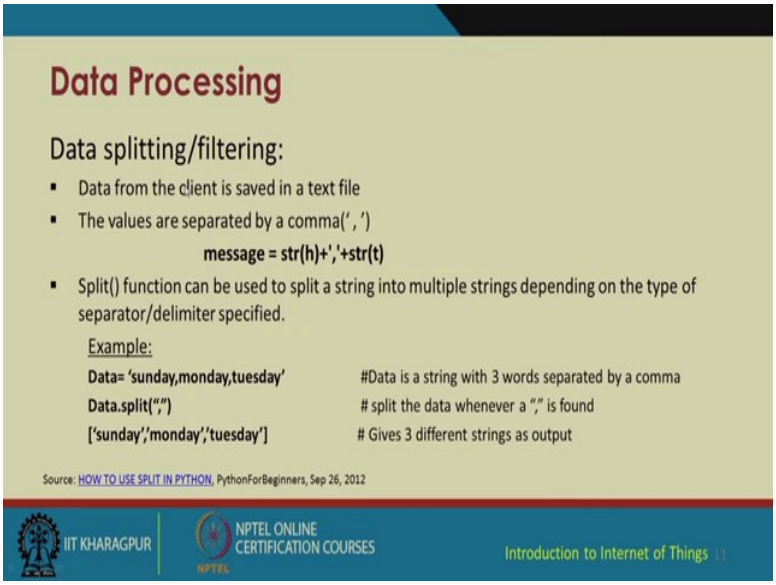
So, over here will be going through those as well as a basics of data processing like data splitting and filtering and data plotting, but big question is why do you need processing of the

data. As in imagine you have hundreds and hundreds of sensors all rapidly uploading their data to this remote server. So, at certain instances there maybe chances of data corruption, there may be chances of incomplete data being transmitted, there may be chances of server not able to receive the whole data despite the client being sent client sending the whole data.

So, there may be various reasons that data may not be properly formatted and so on. So, in those cases you need to filter the data to create a seamless and well structured data base and as regards to data splitting since you have multiple data coming in from a client which may be stored in the form of a csv file or even in the form of a text file. So, we need to if we need to access individual data.

Suppose you are getting temperature readings humidity maybe presence of gases light intensity and so on So, suppose you have five to six sensors connected to a single node, and these data are routinely updated to a server, but you only need to access the temperature value for this to happen you need to split your data at the data log. And once this data has been split and individual data can be accessed we can go about plotting this data.

(Refer Slide Time: 07:16)



Regarding this data splitting, now imagine the data from the client is saved into a text file and the values are separated by a comma as we have covered in previous lectures this is called a csv or comma separated value file. So, instead of a comma you can use any other delimiter, it may be a semicolon, it maybe colon, it maybe even a question mark better avoided, but still. So, these values which were incoming it were in the form of string h, string t.

So, we need to split these values on the server in the form of individual h and t's. So, we will be using this basic inbuilt function of python called split. So, it can be used to split a string into multiple strings depending on the separator or the delimiter we are specifying. For example, suppose we take a data string as within single quotes we assign Sunday, Monday, Tuesday or it can be any other data string separated by a delimiter over here the delimiter is this comma we have chosen this is a comma.

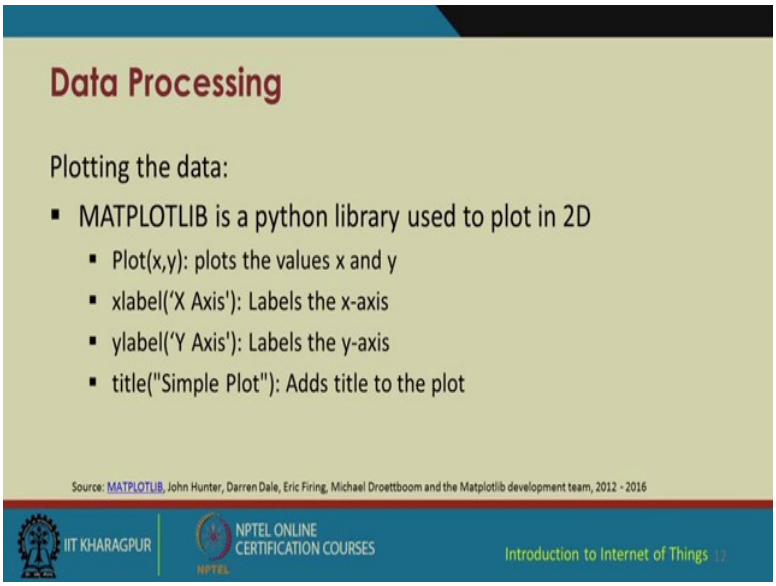
Now, when we call this Data.split function and we specify the delimiter in the arguments, you will see the result will be individual substrings Sunday, Monday Tuesday. We input the data as a single string Sunday, Monday, Tuesday, but we get the output as individual data strings. So, it would be better if we just do a bit of hands on. So, let us suppose our data is maybe 123, abc right.

So, I have given one 123 numeric value I have given abc alphabetical value and I have given a combination of both alphabets and numbers. So, this is my data let us check what has been stored. So, this has been stored this data variable has been assigned a single string that is 123, abc, xyz12 now I need to split this data. So, I use data dot split the separator I have used is a comma.

You see it has been split into three different parts 123, abc and xyz12 you must remember since these were separated by comma. So, these have been separated accordingly, you can obviously, go for another value let us say data equal to 123432454abc44. So, I have used to many 4's let us see what happens. So, this is my data right.

Now, I will use the number 4 as my delimiter. So, data.split I will give the delimiter as 4 you see. So, according to the position of four in the string your single string has been broken down into multiple substrings and the 4 delimiter is obviously, not included. So, you have 123 then you have 32 then you have 5 then abc then a blank since it has two 4's. So, it includes a blank and then rt right. So, it is quite simple.

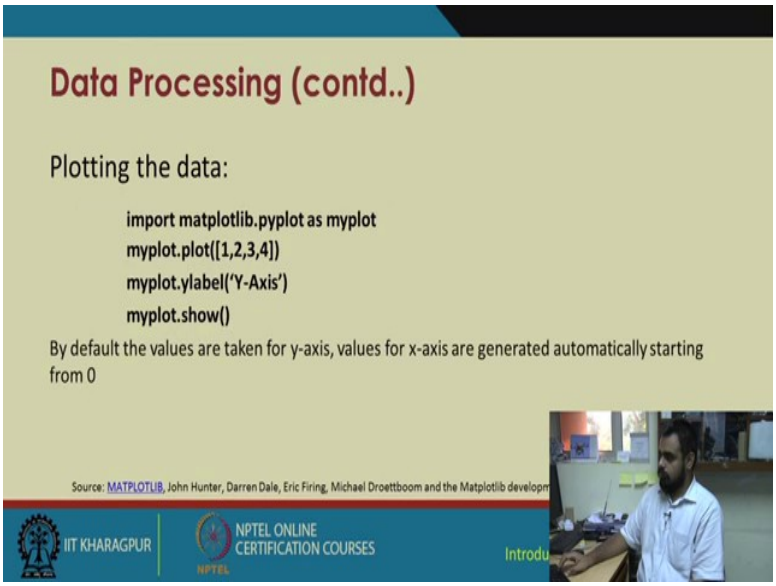
(Refer Slide Time: 11:38)



Now regarding data plotting in MATLAB, so MATLAB sorry regarding data plotting in python it is quite similar to plotting the data in MATLAB so, both of this support a very powerful library called MATPLOTLIB. So, this MATPLOTLIB library you can use the appropriate version in MATLAB as well as in python most of the functions are almost same with a bit of variations, but it is more or less same in both the languages.

So, basic commands for plotting in two dimension are you write Plot(x, y), where you put the x axis and the y axis values, x label gives provides the labels on the plot for the x axis y label provides the label on the plot for the y axis title gives title to the whole plot.

(Refer Slide Time: 12:37)



So, this is a sample code import matplotlib.pyplot as maybe a variable and then you call this myplot.plot you give a range of values then myplot.y label you name it y axis and you show this plot. So, we can again go for a bit of hands on let us say we import this MATPLOTLIB function as let us say plt, we define a variable as maybe I will give a range of values may be from one to twenty.

plt.plot I put in 'a' plt.show(). So, when I run this basic code see what I have done is I have imported this MATPLOTLIB.pyplot library, I have defined a variable 'a' by assigning a range of values between 1 and 20. So, this range function automatically assigns values let us see what it does, range 1, 20. So, it will give you range of values between 1 to 20.

Now, we initialize this plotting by calling a figure, within this figure we plot this value of a which will be; obviously, corresponding to the y axis, the x axis is if not provided it is automatically taken and the show function shows the plot. So, once we run this thing as you can see this plot has been generated.

Since it is a linearly increasing line 1, 2, 3, 4 upto 19. So, I will give you a straight line you can even change the plot type, let us say instead of plt.plot a straight line plot. Let us go for a scatter plot sorry better still it will be more interesting if we give two values assign a variable a1 as for this we have to import this random function.

If you see if you remember in one of the previous lectures this random function was also covered we need to import the random function first sorry starting and ending. So, whenever you are in doubt for this particular spyder base console even actually checking the syntax of this function or you need two arguments a and b.

Let us say I give a as 1 b as 20 and put this whole thing within range. So, this will be my range I will start with this I will change this to length of a1 if it is denoted by len(a1) when I run these two together. So, a1 is 0 to 5 a is 6 right sorry this is going to be only scatter. So, this seems to be a size mismatch

So, for this scatter plot we take two arguments one for x another for y and we have used two function for the x1 we need a linear spacing of numbers. So, we started off with -1 because this would not be included, it will go up to 50 with a step size of one. So, you will have 0, 1, 2, 3, 4, 5, 6 upto 50, so you have 50 points on the x axis where as we call a random function from numpy library module let us call it import numpy as np.

So, on this numpy module we would not be requiring this. So, np dot random dot random number within argument 50 means it will generate 50 random point. So, this will act as my y axis or the data points. So, corresponding to the points on the x axis will have corresponding randomly generated points on the y axis and within this same figure and show let us see what happens when we run this thing.

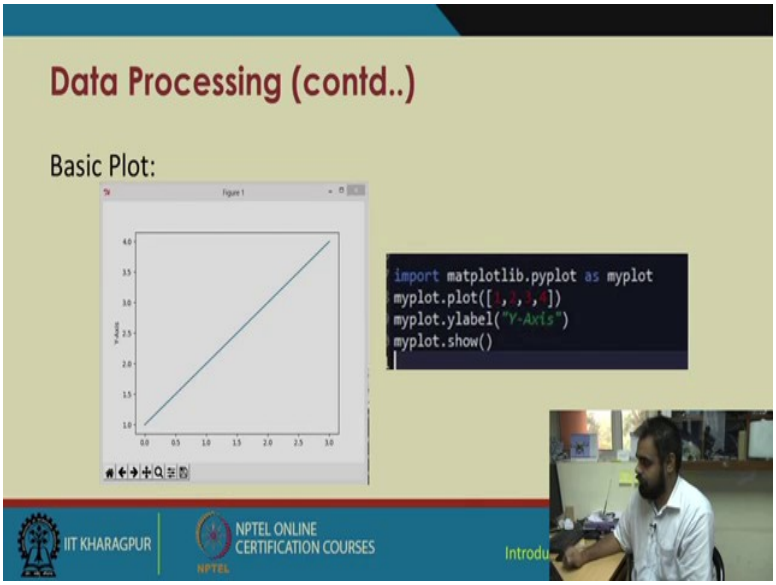
So, you see on the x axis and corresponding to the x axis you have 50 points on the y axis. So, let us add a few bits to it, if you run this part you just added a title for this plot you see this scatter plot example additionally you can have plt.y label you must remember these labels are strings. So, should be within quotations.

Let us put the y label as randomly generated, the x label as linearly spaced let us run this you see the y label has changed to randomly generated the x label has changed to linearly spaced you just add one more command to it. If you add this command it will show a it will turn up a grid on your plot. So, you have a linear grid on your plot.

So, I hope this basic plotting function is clear. So, you have a simple plotting you have labeling you show this plot and besides your normal 2d plot line plot and scatter plot, you have a huge range of plotting option you can even venture into 3d plots which are much more

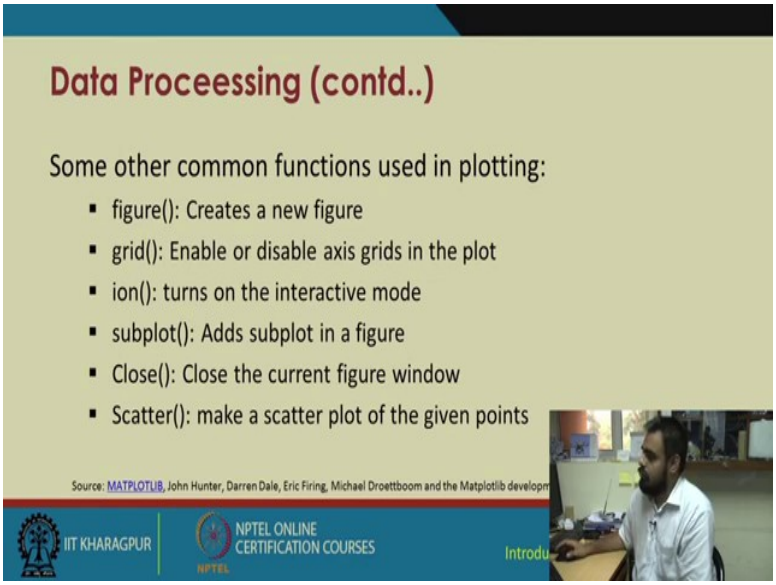
attractive in a browser base system or for various publication. So, this basic plotting you saw previously.

(Refer Slide Time: 23:04)



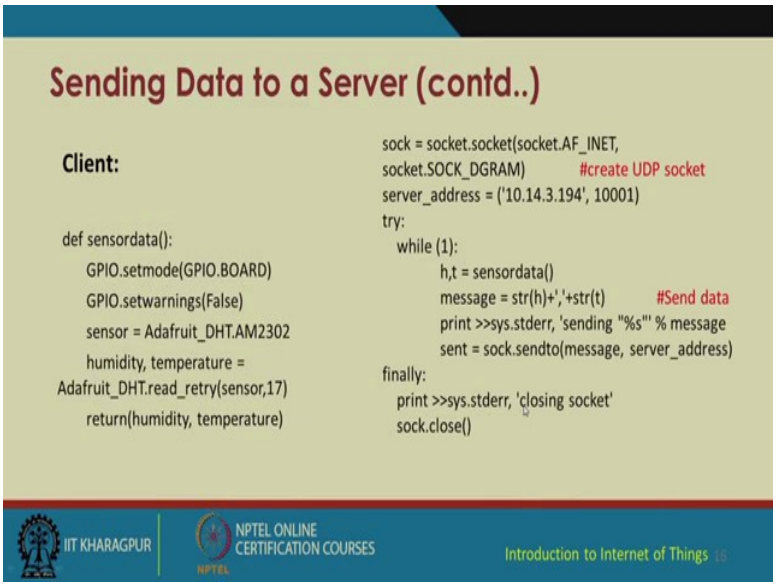
Now, some other functions used in plotting are covered this figure it creates a new figure grid it enabled or disabled an access grid on the plot ion is the interactive plot mode.

(Refer Slide Time: 23:12)



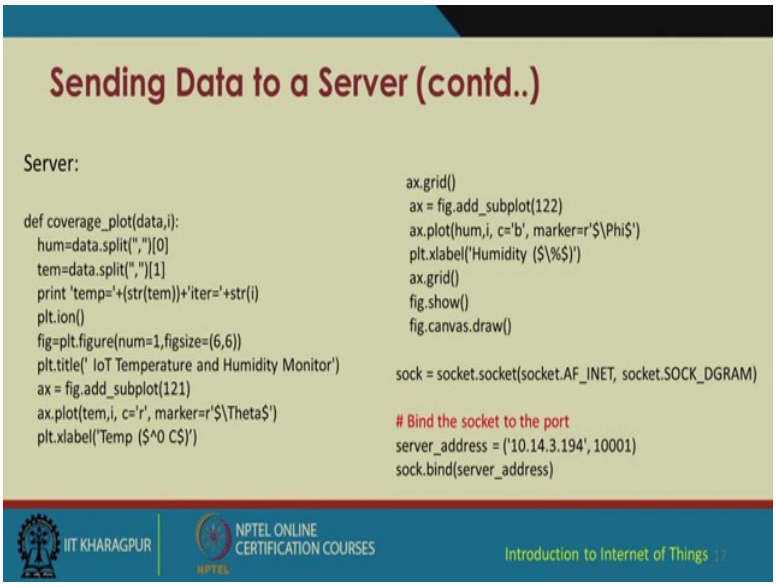
This is generally used for animations your subplot it adds a subplot in the figure close it closes the current figure scatter makes a scatter point plot of the given points, so will try this small. So, it would be better if we directly go into this model again.

(Refer Slide Time: 23:52)



So, if you remember for the client side this sensor data module is more or less same. In fact, it is the exact same one, the client does not require a IP address because we are aiming for a one way communication from the client to the server. So, we just put in the servers IP address and that required port to which on which the server is listening and just like before we are packaging the data in the form of h and t, and transmitting it over the socket to the server.

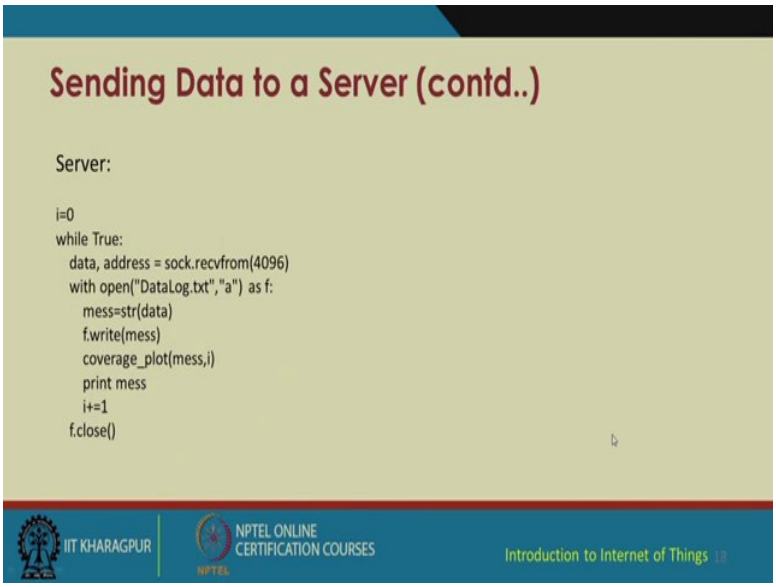
(Refer Slide Time: 24:32)



Now, on the server side we have added a new function called coverage plot. What it does is it animatedly plots whatever data is in coming into the server over the socket. So, for the time

being ignore this part this particular function, let us pay attention to this socket creation. So, it is somewhat similar to the previous examples over here your server address is the server address and the particular port to which the server is listening.

(Refer Slide Time: 25:09)



And while it is true it is receiving data in 4096 bytes junks you open a data log file store the data to it and additionally call this plotting function.

So, this plotting function is being called again and again basically iteratively within this loop. So, as long as your data is incoming this plotting function will keep on going. So, once the server is closed by the user or the client is closed by the user this plotting function will stop. So, and this I plus one this keeps on incrementing this normal counter which we sent to the coverage plot.

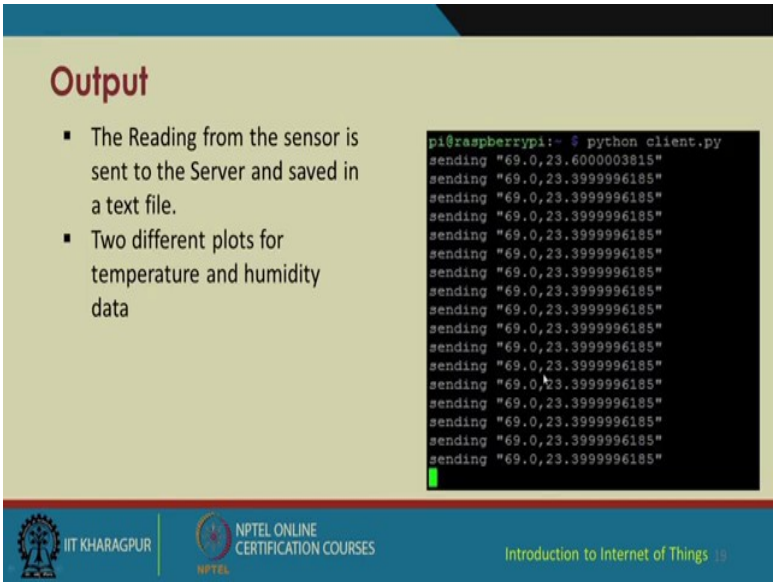
Coming back to the coverage plot it accept two arguments one is the data which comes from the server, and I which is the counter now the data is being split in terms of two parts humidity and temperature. So, after splitting since the data was in a csv format. So, after splitting the zeroth element becomes humidity and the first element is temperature. So, you need to actually study the data very carefully like what format the client is sending a data and so on.

Now, if you want to print the temperature, you just print the temperature and the iteration number then you start the interactive plot mode you set the figure two similar number as

number 1, otherwise it will keep on creating new windows every time it plots something new to set the window to 6 comma 6 it can be 8 comma 8. It is up to the user actually you give the plot a title, you add a subplot to it and just basic you plot the temperature versus the iteration that is normal temperature plot with respect to whatever data is incoming we label it.

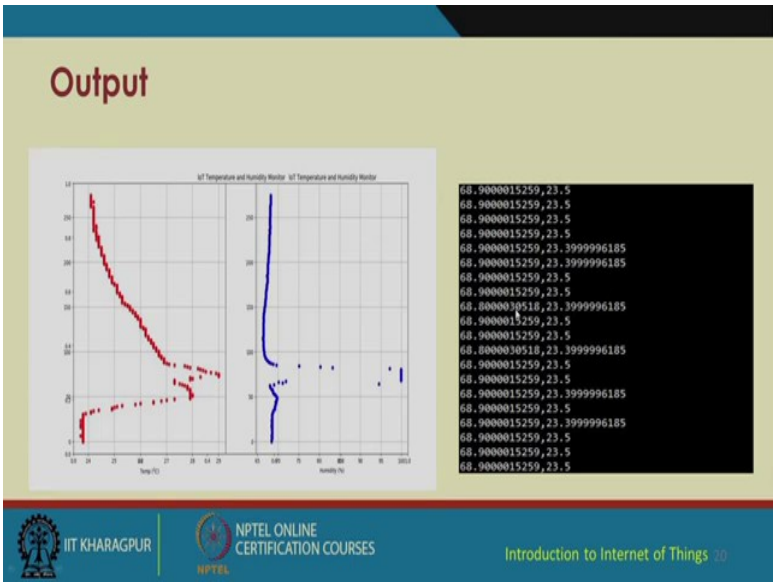
Similarly, for the humidity part also you iteratively plot the data against i, and eventually show the data and figure.canvas.draw it redraws the image without overwriting the previous points.

(Refer Slide Time: 27:36)



So, your output will be something like this. So, this will be on the raspberry pi this is the client code.

(Refer Slide Time: 27:47)



And on the server side you will be getting your data as your covered in the last lecture, and for the plot you will be getting something like this, left one side is the temperature plot the right one side is the humidity plot.

So, let us look at the plots, first of all I will start my server. So, this desktop I am operating acts as my server. So, I will open up my server. So, my server code is now running, now login to the remote client and I need to run the client code which is the same as the previous one. So, one more point you need to notices is suppose you have 100 clients, you do not need to update each and every client.

Even if you need some minor adjustments make some minor adjustments. So, you can just update a singular source code or a singular script file on the server end and that will save you a lot of time as well as a lot of effort. So, I will run this client code client.py . So, you see client has started sending the data and on the server side it is receiving the data this is a new addition this iteration number.

So, this keeps on increasing the left hand side denotes the temperature on the plot, the right hand side denotes the humidity, it is maybe will stop the plot run this thing again. So, see even after stopping this program data which was previously stored maintains there. So, there is very minute variation between 75.5 and 76.

(Refer Slide Time: 30:48)

So, it is giving changes where as the temperature is more or less constant. So, will try increasing the temperature little bit and you can see the plot changing. As you can see the temperature points now logged are raising see the temperature has risen and there have been changes in the humidity also, now let us wait for it to recover. Now you can again notice the temperatures gradually coming down on the left plot the red plot you can see the temperature gradually coming down in steps.

So, this will keep on coming down until it reaches the normal room temperature value. So, now, let us close the client program see the client program has been stopped manually and this data logging stops. So, one more thing you notice is in the previous examples your temperature humidity values were coming in concurrently, now in this one since we have spitted the data over here in these parts. So, we have a separate temperature component and iteration number and a separate humidity component which is saved in the same data log, data log is still the same only the scripting part is changed.

So, this thing will help save you a lot of effort you do not need to individually code the clients. So, once your client has been coded and deployed on the field if you need to make changes make design your architecture in such a way that only changing the server will help you only changing the scripts on the server will help you change your whole implementation and functionality. So, I hope this was of some values to you.

Thank you.