|  |
| --- |
| *School of Computer Science & Engineering* ***Network Programming*** *June 16 2019*  **Realtime Home Security system using Raspberry Pi IoT**  Class day ( **Network Programming / Wed** ) Student No. ( **2016118382** ) Name: ( **Libin Joseph** ) |

1. **Motivation & Objectives**

* **Motivation**: There is a need for realtime security systems in the world where crime rate is growing , people don't always have time to stay in their home guarding what is going on , so with my project people can monitor what is going on in there home while they are away.This project is modular so people can replace sensors easily to the raspberry pi system. without configuring any major portions in the server.
* **Objectives**: The Objective of the project is to make a modern GUI based server which can communicate with a client to receive sensor data and interpret the data to the user for a better User Experience.
* **Expected results**: The program is expected to show a GUI window where users can manage the sensor and see what the sensor is showing in realtime.

1. **Introduction of the Project**

* **Solutions of the problem:** The solution of the problem is to send continuous data from the raspberry pi to the server ,but its not that easy as it is said as there are a bunch of challenges associated with it when it comes to making it more interactive to the users.The biggest challenge was working with gui and realtime data that is the GUI runs on its own loop to elaborating - I used gtk library for gui and gtk runs on its own event loop and our socket also runs on its own separate loop to sync it in c is a bit complex and requires multiprocessing, but our challenge is to make it portable for low end devices or to be specific it should not take much resources and more importantly it should be easier for the installers to modify the code. So after trial and error of different methods i found that forcing the socket on the same event loop of the gtk would be plausible and pausing the gtk constantly and calling the update function of the gtk gives an illusion of a real time multi process system but that hack comes with its own cost, i.e we cannot pause the main event loop for less than 200 milliseconds it would crash but syncing it to the raspberry pi to be 200milliseconds is not much of a loss as considered for a low-end system , it could be barely noticed. Successfully updating the gui with real time data from the from raspberry pi to the server with the help of a socket syncing with the gtk event loop..
* **Overview of your solution:** Here is the function which i explained about in the solution where i synced the main gtk event loop with the reception loop of the socket for updation of GUI in real time.

Here the counter variable is first set to 0 and then when the connection is accepted successfully it is set to 1 where only that part of the code will work where the main execution goes on, It is done because the whole function will be called again and again.

Now an alias of the gui widget is created and the value to be updated is assigned to it and then the value we read is synced with the raspberry pi because the timeout specified is 200milliseconds.

now as the values are retreived , its split and then the distance and motion value is used update the animation on the GUI and also the main warning.

|  |
| --- |
| **static gboolean on\_timeout (gpointer user\_data) {   if(counter == 0){  clnt\_addr\_size = sizeof(clnt\_addr);  clnt\_sock = accept(serv\_sock, (struct sockaddr \*)&clnt\_addr, &clnt\_addr\_size);  if (clnt\_sock == -1) {  error\_handling("accept() error");  } else {  printf("Connected client %d \n", i + 1);  counter =1;  }  }  if(counter == 1){  GtkLabel \*label = GTK\_LABEL (user\_data);  GtkLabel \*lab2 = GTK\_LABEL (label2);  GtkLabel \*motlab = GTK\_LABEL (motOk);    GtkImage \*img = GTK\_IMAGE(doorclosed);  GtkImage \*img2 = GTK\_IMAGE(motDetect);  str\_len = read(clnt\_sock, &msg, BUF\_SIZE);  gchar \*text = g\_strdup\_printf (msg);  char \*array[2];  if(strlen(text)<=12){  int i = 0;  char \*p = strtok (text, ",");    while (p != NULL)  {  array[i++] = p;  p = strtok (NULL, ",");  }  }   double distance = atof(array[0]);  int sonar=0,mts=atoi(array[1]);  if(distance >=5){  sonar=1;  gtk\_image\_set\_from\_file(img,"doorOpen.png");  distance=roundf(distance \* 100) / 100;  gchar \*text2 = g\_strdup\_printf ("your door is opened by\n %.2f cms !\n",distance-5);  distance=0;  gtk\_label\_set\_label (label, text2);  }  else if(distance < 5 && distance >0){  gtk\_label\_set\_label (label, "your door is closed");  gtk\_image\_set\_from\_file(img,"doorClosed.png");  sonar=0;  }  if(mts==1){  gtk\_image\_set\_from\_file(img2,"motDetected.png");  gtk\_label\_set\_label (motlab, "Motion Detected!");  }else if(mts==0){  gtk\_image\_set\_from\_file(img2,"motionNormal.png");  gtk\_label\_set\_label (motlab, "No Motion Detected");  }  if(sonar==1 || mts==1){  gtk\_label\_set\_label (lab2, "Your Home is at risk âš ï¸");  }  else  gtk\_label\_set\_label (lab2, "Your Home is secure");   while (gtk\_events\_pending())  gtk\_main\_iteration();  g\_free (text); }   return G\_SOURCE\_CONTINUE;  }** |

* **Functionality of the project:**

1.) It can show a real time animation of the state of your door in the home or the state of the motion detector in your home i.e if it open or not and if there is a motion detected in a room respectively

2.) It can also show by how much amount your door opened in centimeters.

3.) You can Deactivate/Activate the sensor , its necessary to shut the sensor for some cases where you don't want to be warned. eg) you yourself getting into your house.

5) You can replace the sensors with other sensors which you might like , i.e upgrading wont take much time

4) Its simple and straightforward design , no cluttering and can be used by anyone.

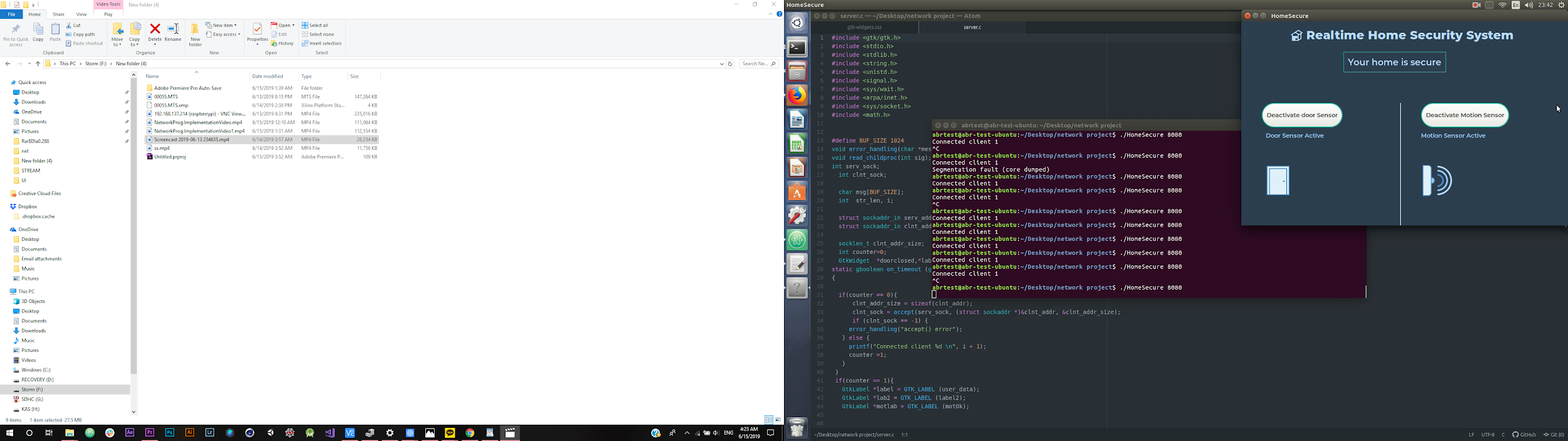
1. **Results**

Here is the video of my execution where you can see the screen capture clearly while im using the sensors, There are captions in the video which will clearly help explain the whole project , its parts and its working.

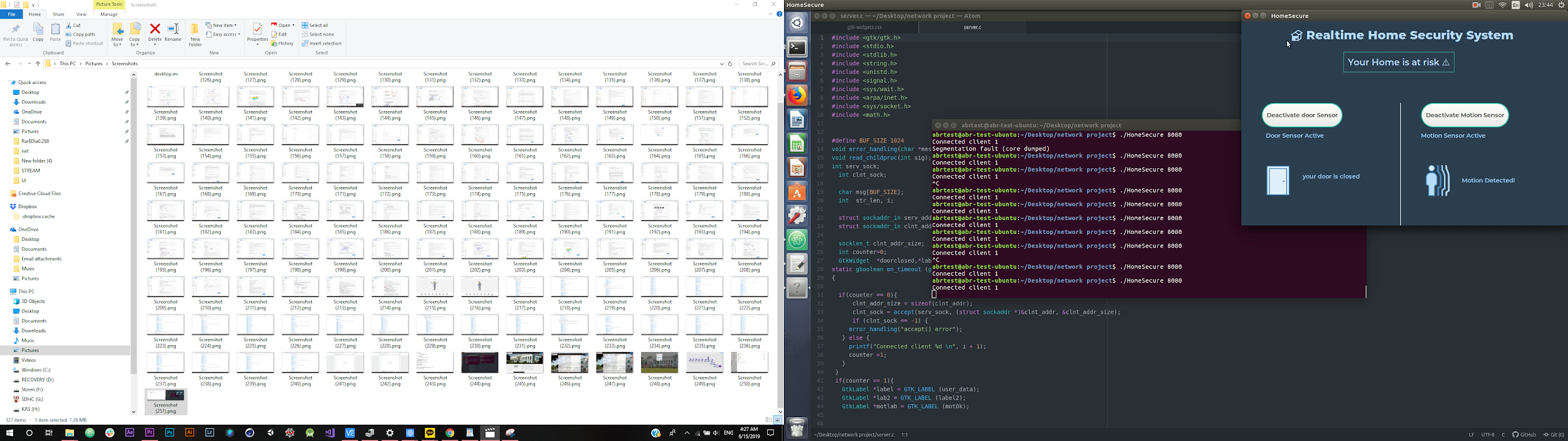
Video: <https://www.dropbox.com/s/sc5veuu5m97uahk/NetworkProg.ImplementationVideo1.mp4?dl=0>

I recommend to watch the video in a higher resolution for better experience

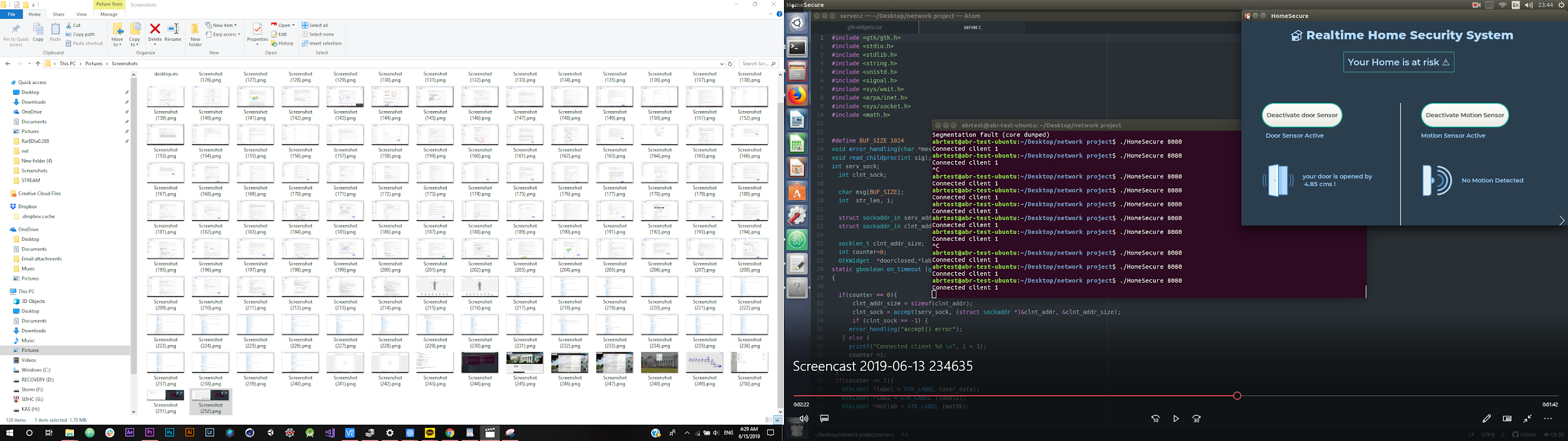
Some Screenshots of the GUI



Motion Detection -



Door open Detection with amount



Component setup

