Ömer Kibar 150119037

Müslim Yılmaz 150119566

Serkan Korkut 150119036

**Project Summary**

In this project we implemented a networked system using two virtual sensors, a gateway and a server. TCP and UDP communication protocols are used for transmitting temperature and humidity values. Also we have a user-friendly interface where both humidity and temperature data can be viewed as HTML objects.

**Solution Approach**

1. Temperature sensor generates values randomly between 20 and 30 every second and sends the values and timestamps to the gateway via TCP. **readTempature()** function is responsible for generating the random data and in the **main** function we create a socket and send temperature data to gateway.
2. The humidity sensor generates random values between 40 and 90 every second and sends the data to the gateway via UDP only if the value exceeds 80. Additionally, the humidity sensor sends an 'ALIVE' message every 3 seconds to indicate that it is working properly. Messages to the gateway are sent from main function in an infinite while loop. readHumidity function is responsible for generating humidity data.
3. Gateway receives data from both sensors, adds 'TEMPDATA' or 'HUMIDITYDATA' to the beginning of the data depending on which sensor it comes from, and forwards it to the server. In addition, if it doesn’t receive any value from the temperature sensor for 3 seconds, sends a 'TEMP SENSOR OFF' message to the server. Likewise, if it doesn’t receive an 'ALIVE' message from the humidity sensor for more than 7 seconds, sends a 'HUMIDITY SENSOR OFF' message to the server. **listenTempSocket()** and **listenHumiditySocket()** are responsible for getting data from the temperature and humidity sensors. Also it sends the data for server with the help of **send\_message\_to\_server()** function.
4. The server and the gateway are connected via TCP. After they receive a 'HELLO' message from each other, the gateway starts sending the data to the server. The server receives each data and stores separately depending on whether it starts with TEMPDATA or HUMIDITYDATA. The server also features a web interface with an HTTP process listening on port 8080. Accessing 'http://localhost:8080/temperature' displays an HTML object with all temperature data. Accessing 'http://localhost:8080/humidity' displays an HTML object with all humidity data.

The function **read\_data()** is tasked with identifying messages originating from the gateway and appropriately storing the provided data. **create\_HTTP\_server()** is tasked for creating html server on port 8080 and shows the data with correct endpoints with the help of the functions **prepare\_homepage\_html()**, **prepare\_temperature\_html()** and **prepare\_humidity\_html().** Above functions create the html content according to each endpoint and displays to the screen.

**Encountered Problems**

* We didn’t know how to handle http requests without using any http library but after researching we found a simple solution for that.
* Keeping sensors status in the server was also hard. We solved this problem using settimeout function of sockets used between gateway and sensors. If no messages arrives to socket within a specific seconds it raises exception this way understand a sensor is disconnected and inform the server. If a temperature or humidity data arrives to server it understands that sensor is connected.

**Unresolved Issues**

We are unable to implement the bonus feature. Additionally, on the HTML page, a manual page refresh is required to view incoming new data.

**Usage Explanation**

1. Run gateway.py

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Run server.py

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Run tempSensor.py and humiditySensor.py

metin, ekran görüntüsü, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, kalıp, desen, düzen, tasarım içeren bir resim

Açıklama otomatik olarak oluşturulduGateway.py and server.py after the sensors start running:

Output.log file before sensors start running:

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

Output.log file after sensors start running:

metin, ekran görüntüsü, menü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Accessing all temperature data via ‘http://localhost:8080/temperature’:

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

Accessing all humidity data via ‘http://localhost:8080/humidity’:

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

After disconnecting the temperature sensor:

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, yazılım, ekran, görüntüleme içeren bir resim

Açıklama otomatik olarak oluşturuldu

After disconnecting the humidity sensor:

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu