UNIVERSIDAD DE ORIENTE
NÚCLEO ANZOÁTEGUI
ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS
DEPARTAMENTO DE COMPUTACIÓN
LABORATORIO DE PROYECTOS DIGITALES AVANZADOS

Servidor web con Pi Pico W
Para recibir y enviar archivo

Prof. Rene Cabrera

Elaborado por:

Brazon Eulises

Barcelona, 4 de Marzo del 2023

ABSTRACCION

La presente práctica está basada en una práctica anterior denominada ¨ *Transmisión de información entre dos Pi Pico W*¨. El cual se encuentra alojado en https://github.com/EulisesBrazon/server_picoW, en él se encuentra el código empleado para entablar una comunican entre dos picos.que servirá como base para la transmisión y recuperación de un archivo de texto

OBJETIVOS

Objetivo General:

- Entablar una comunicación entre dos Rapsberry pi pico W para la transmisión de archivos.

Objetivos Específicos

- Crear un servidor web con su propia red wifi.
- Conectarse a la red wifi desde un pico cliente.
- Enviar información de temperatura, a través de la red wifi para visualizarla en una pantalla.
- Generar un archivo desde un pico w cliente que contenga todas las lecturas realizadas hasta el momento, y que se utilizara para ser enviado al pico servidor.
- Poder visualizar los datos que fueron enviados al pico w servidor a través de una página web.

MATERIALES REQUERIDOS

- Jumpers para la conexión
- Protoboard.
- Modulo RGB
- Pantalla LCD TFT 1,8 pulgadas chip de control ST7735, de 128X160px.
- Pantalla LCD TFT 1,8 pulgadas chip de control ST7735, de 128X160px.
- Sensor LM35
- 2 Raspberry pi pico w
- Alimentación 5V.
- Herramientas para la manipulación y testeo (multímetro, pinzas, entre otros)

DESCRIPCION DEL PROYECTO

En la práctica anterior se entabla la comunicación entre dos raspberripy pico w, se aprovechará el mismo circuito para emplear la nueva funcionalidad de trasmisión de archivo, a la identificación del tipo de información que se esta enviando, se le agrega lógica adicional, para ordenar los distintos valores, que forman parte del archivo.

Tanto del lado del cliente como del servidor, contara con una función dedicada al almacenamiento y lectura de archivo, en esta practica se trabajo con archivo de texto, el cual para identificar las distintas lecturas debe de contar con separador que debe ser el mismo en ambos picos.

Al momento de generar la página web para visualizar los datos, se debe colocar la información en un formato correcto que el navegador pueda entender, y asi poder visualizar las lecturas contenidos en el archivo.

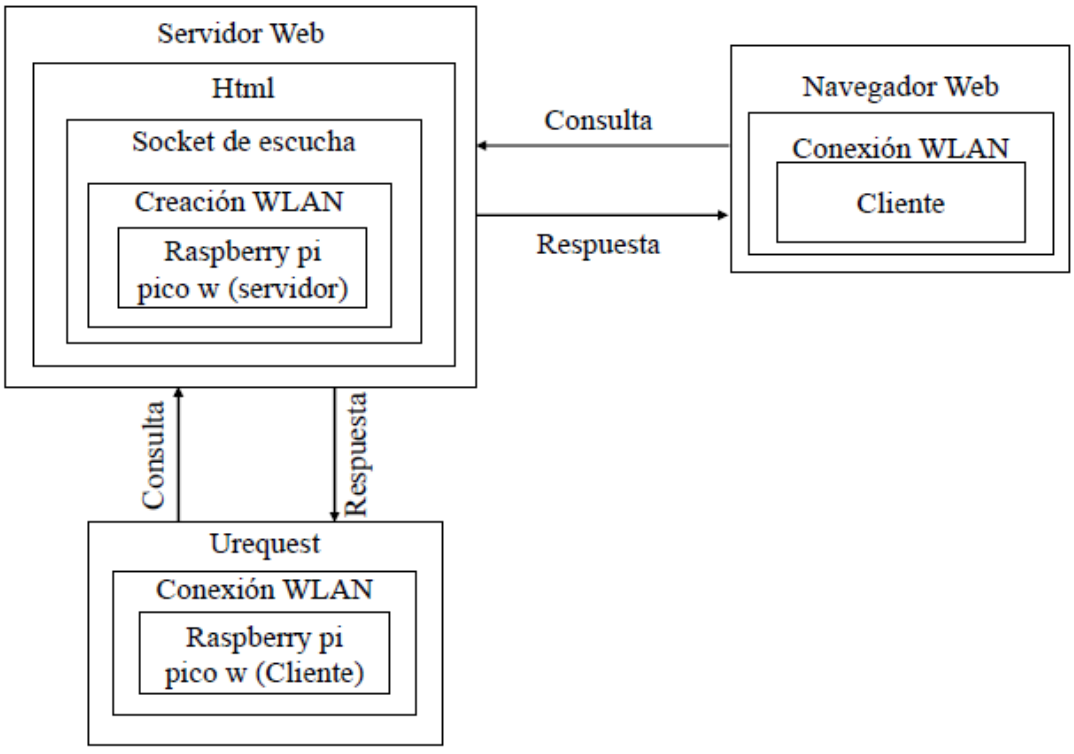
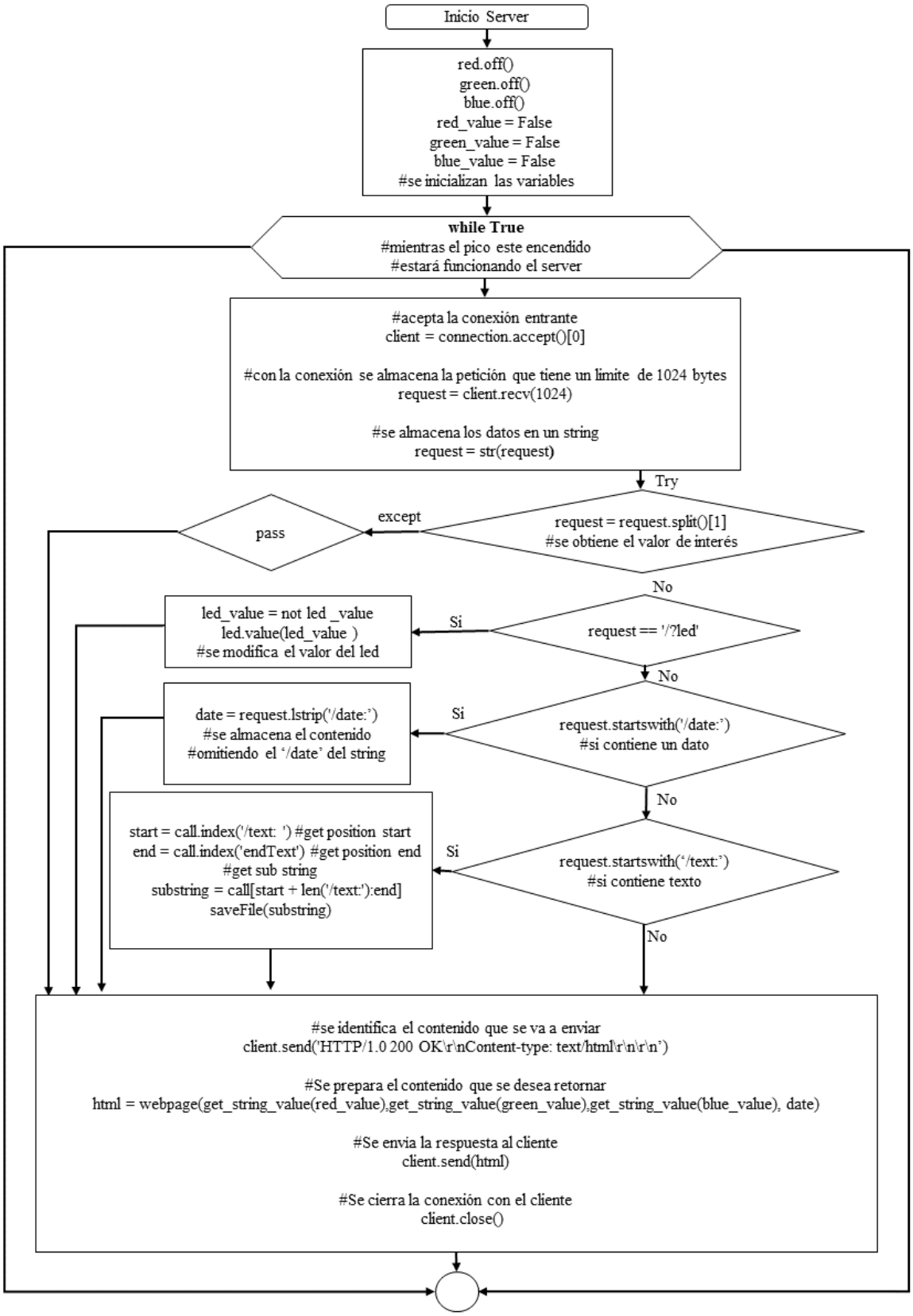
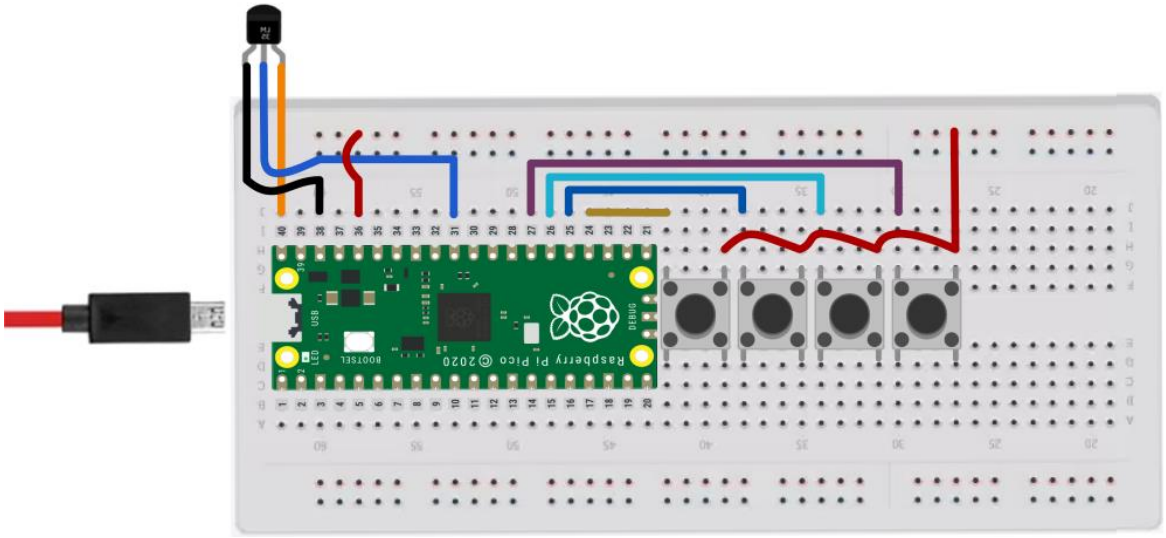


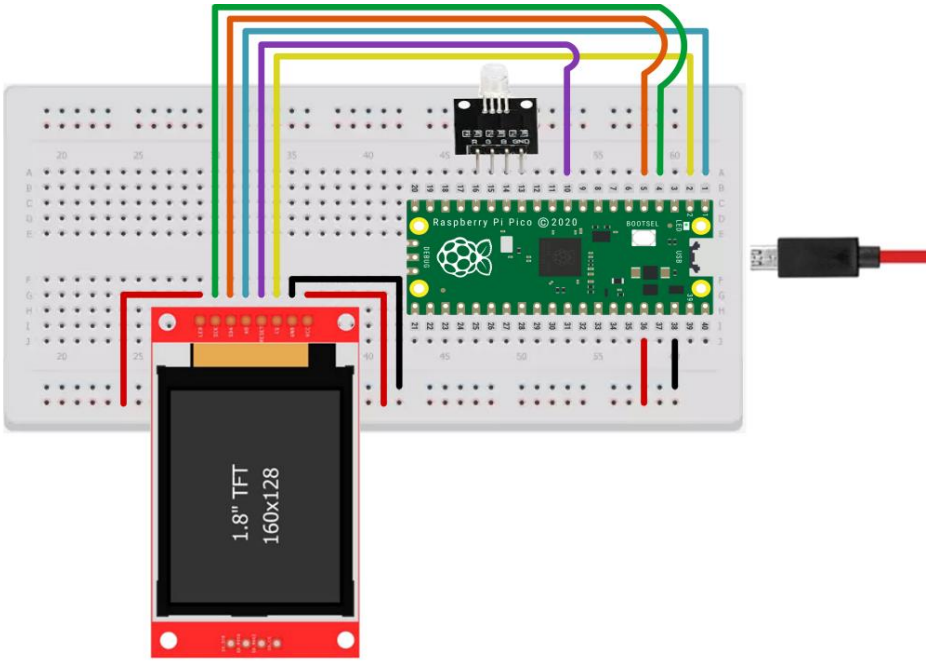
Diagrama de Flujo



Circuito Utilizado (Cliente)



Circuito Utilizado (Servidor)



Código de las funciones modificadas al Servidor del código del informe anterior

```
def saveFile(string):
    file = open("Info.txt", "w")#changed the opening mode, which creates the file
automatically
    string = string.split()#Separa la cadena de string, en un arreglo con los diferentes valores
    for word in string:
            #se guarda durectamente con la etiqueta para que lo reconozca el navegador
        file.write("<p>"+ word +"</p> ")
    #file.write(string)
    file.close()

def loadFile():
    try:
        file = open("Info.txt", "r")
        return file.read()
    except:
        return 'None'

def webpage(isOnRed, isOnGreen, isOnBlue):
    #Template HTML
    html = """
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>RGB Controller</title>
    <style>html {font-family: Helvetica;
                    display: inline-block;
                    margin: 0px auto;
                    text-align: center;}
                h1 {color: #0F3376; padding: 2vh;}
                p {font-size: 1.5rem;}
                table {margin: auto;            }
                td{padding: 10px ;}
                .Button {           border-radius: 31px;
                    display: inline-block;
                    cursor: pointer;
                    color: #ffffff;
                    font-family: Arial;
                    font-size: 17px;
                    font-weight: bold;
                    font-style: italic;
                    padding: 17px 19px;
                    text-decoration: none;          }
```

```
            .ButtonR {background-color: #ec4949;
               border: 6px solid #991f1f;
               text-shadow: 0px 2px 2px #471e1e;}
            .ButtonR:hover {background-color: #f51616;}
            .Button:active {position: relative;top: 1px;}
            .ButtonG {background-color: #49ec56;
               border: 6px solid #23991f;
               text-shadow: 0px 2px 2px #1e4723;}
            .ButtonG:hover {
               background-color: #29f516;}
            .ButtonB {background-color: #4974ec;
               border: 6px solid #1f3599;
               text-shadow: 0px 2px 2px #1e2447;}
            .ButtonB:hover {background-color: #165df5;}
   </style>
</head>
<body>
   <h1>Raspberry Pi Pico W Web Server</h1>
            <p>RGB Control</p>
            <table>
               <tbody>
                  <tr>
                     <td>
                        <p><a href="/led-r"><button class="ButtonR
                                 Button">R</button></a></p>
                     </td>
                     <td>
                        <strong> """+ isOnRed +"""</strong>
                     </td>
                  </tr>
                  <tr>
                     <td>
                        <p><a href="/led-g"><button class="ButtonG
                                       Button">G</button></a></p>
                     </td>
                     <td>
                        <strong> """+ isOnGreen +"""</strong>
                     </td>
                  </tr>
                  <tr>
                     <td>
                        <p><a href="/led-b"><button class="ButtonB
                                 Button">B</button></a></p>
                     </td>
                     <td>
                     <strong> """+ isOnBlue +""" </strong>
                     </td>
```

```
                    </tr>

                </tbody>
            </table>

</body>
<footer>
    """+ loadFile() +"""
</footer>
</html>

        """
    return str(html)

def startDisplay(date):

    size=2
    separation = 4
    h=0
    #data reading
    date = str(date)
    tft.fill(TFT.BLACK)#clean screen
    tft.text((0, h), "IP: ", TFT.RED, sysfont, size, nowrap=True)
    h += sysfont["Height"]*size+separation
    tft.text((0, h), str(ip), TFT.GREEN, sysfont, size, nowrap=True)
    h += sysfont["Height"]*size+separation
    tft.text((0, h), "SSID: ", TFT.RED, sysfont, size, nowrap=True)
    h += sysfont["Height"]*size+separation
    tft.text((0, h), ssid, TFT.GREEN, sysfont, size, nowrap=True)
    h += sysfont["Height"]*size+separation
    tft.text((0, h), "Temperatura: ", TFT.RED, sysfont, size, nowrap=True)
    h += sysfont["Height"]*size+separation
    tft.text((0, h), date, TFT.GREEN, sysfont, size, nowrap=True)

def refreshDisplay(date):
    global oldDate
    size=2
    #data reading
    date = str(date)
    tft.text((0, 100), oldDate, TFT.BLACK, sysfont, size, nowrap=True)
    tft.text((0, 100), date, TFT.GREEN, sysfont, size, nowrap=True)
    oldDate = date
```

```
def serve(connection):
    #Start a web server
    global date

    red.off()
    green.off()
    blue.off()

    red_value = False
    green_value = False
    blue_value = False


    while True:

        client = connection.accept()[0]
        call = client.recv(1024)
        call = str(call)

        try:
            request = call.split()[1]
        except IndexError:
            pass
        print(f'Request {request}')
        if request == '/led-r':
            red_value = not red_value
            pave = red_value
            red.value(red_value)

            client.send('HTTP/1.0 200 OK\r\nContent-type: text/html\r\n\r\n')
            html =
webpage(get_string_value(red_value),get_string_value(green_value),get_string_value(blue
_value))
            client.send(html)
        elif request == '/led-g':
            green_value = not green_value
            green.value(green_value)

            client.send('HTTP/1.0 200 OK\r\nContent-type: text/html\r\n\r\n')
            html =
webpage(get_string_value(red_value),get_string_value(green_value),get_string_value(blue
_value))
            client.send(html)
        elif request == '/led-b':
            blue_value = not blue_value
            blue.value(blue_value)
```

```
            client.send('HTTP/1.0 200 OK\r\nContent-type: text/html\r\n\r\n')
            html =
webpage(get_string_value(red_value),get_string_value(green_value),get_string_value(blue
_value))
            client.send(html)
        elif request.startswith('/date:'):
            date = request.lstrip('/date:')#remove /date:
            refreshDisplay(date)

            client.send('HTTP/1.0 200 OK\r\nContent-type: text/html\r\n\r\n')
        elif request == '/text:':
            print('entro')
            try:
                start = call.index('/text: ') #get position start
                end = call.index('endText') #get position end
                substring = call[start + len('/text:'):end] #get sub string
                saveFile(str(substring))
            except IndexError:
                pass

            client.send('HTTP/1.0 200 OK\r\nContent-type: text/html\r\n\r\n')
            html =
webpage(get_string_value(red_value),get_string_value(green_value),get_string_value(blue
_value))
            client.send(html)
        else:
            client.send('HTTP/1.0 200 OK\r\nContent-type: text/html\r\n\r\n')
            html =
webpage(get_string_value(red_value),get_string_value(green_value),get_string_value(blue
_value))
            client.send(html)
        client.close()
        print(gc.mem_free())

def runServer():
    try:
        global ip
        global date
        ip = accesPoint()
        connection = open_socket(ip)
        startDisplay(date)
        serve(connection)

    except KeyboardInterrupt:
        machine.reset()

runServer()
```

Código cliente

```
#define pins
buttonRed = machine.Pin(21,machine.Pin.IN, machine.Pin.PULL_DOWN)
buttonGreen = machine.Pin(20,machine.Pin.IN, machine.Pin.PULL_DOWN)
buttonBlue = machine.Pin(19,machine.Pin.IN, machine.Pin.PULL_DOWN)
buttonText  = machine.Pin(18,machine.Pin.IN, machine.Pin.PULL_DOWN)

def readTemperature():
    conversion_factor = 3.3/ 65535
    temp_voltage_raw = analog_value.read_u16()
    convert_voltage = temp_voltage_raw*conversion_factor
    tempC = convert_voltage/(10.0 / 1000)
    return tempC

delay = 0.3
def buttons():

    if buttonRed.value()==1:
        print("Red")
        http_get("led-r")
        sleep(delay)
        sentTemp()

    if buttonGreen.value()==1:
        print("Green")
        http_get("led-g")
        sleep(delay)
        sentTemp()

    if buttonBlue.value()==1:
        print("Blue")
        http_get("led-b")
        sentTemp()
        sleep(delay)

    if buttonText.value()==1:
        #http_get("led-b")
        sentText()
        sleep(delay)
```

```
def saveFile(temperature):
    try:
        file = open("Info.txt", "ab")#Try, if it doesn't exist
        file.write(str(temperature)+"C")
        file.close()
    except:
        file = open("Info.txt", "wb")#changed the opening mode, which creates the file
automatically
        file.write(str(temperature)+"C")
        file.close()

def loadFile():
    try:
        file = open("Info.txt", "r")#Try, if it doesn't exist
        return file.read()
    except:
        print("archivo no existente")

def sentText():
    text = loadFile()
    print("entro aqui")
    print(text)
    http_get('text: '+str(text)+'endText')
```

Código completo alojado en:

https://github.com/EulisesBrazon/server_picoW

Información Referencial:

Sensors_and_display_UART. Recuperado de:https://github.com/EulisesBrazon/sensors_and_display_UART

Raspberry Pi Pico W :08 web server - WIFI - RGB control. Recuperado de :https://youtu.be/606NxqqJr9I

Getting started with your Raspberry Pi Pico W. Recuperado de: https://projects.raspberrypi.org/en/projects/get-started-pico-w/0