**Assignment No. 14**

**Title:** Write a server application to be deployed on Raspberry-Pi /Beagle board. Write client applications to get services from the server application.

**Aim:** To understand the interface between web server and Raspberry Pi.

Demonstration of creating own web server in order to store information generated through sensors.

**Hardware Requirement**:

Raspberry Pi board with an SD card

**Software Requirement:** Raspbian O.S, Python.

**Theory:**

Following are the steps of implementing web server based application.

**Step 1:** Giving IP address to the Raspberry Pi Board.

* Connect the board to the monitor using the HDMI-VGA cable.
* Power ON the board and let the Raspbian OS boot.
* When the OS is booted connect the LAN cable to the Rpi and assign a static IP address to it.
* If your college setup has a DHCP server the Rpi will be dynamically assigned a IP Address.
* You can open a terminal as note down the IP Address by giving the command on the terminal ifconfig <enter>

**Step 2:** Update the Raspbian OS.

* on the terminal give the command

sudo apt-get update <enter>

followed by the command

sudo apt-get upgrade <enter>

**Step 3:** Install PHP.

* To install PHP on your raspberry pi give the command on the terminal

sudo apt-get install php5 libapache2-mod-php5 –y <enter>

**Step 4:** Install Apache Web Server.

* If Apache web server is not installed on your raspberry pi the install it using the command

sudo apt-get install apache2 –y <enter>

if the apache web server is installed successfully the you can see the folder www/html in /var directory

* Now restart the server with the command

sudo service apache2 restart <enter>

**Step 5**: Open a terminal and go to **/home/pi/GROUP\_D**

You can see the install.sh file.

In order to run this file give the following commands

sudo chmod +x install.sh <enter>

followed by

sudo ./install.sh

the install script will copy the folders into the /var/www/html folder.

**Step 6:** Executing the programs.

* Note down the IP Address of the raspberry pi board.
* Make sure it is connected to the LAN
* Now use a desktop /laptop connected to the same LAN network
* Open a browser on the desktop/laptop and type the ip address in the address bar.

E.g. if the ip address of your rpi is 192.168.1.50

* Then in the address bar of the browser type 192.168.1.50 and hit enter.

**Step 7:** Then run the program LED

* remoteLED

you can switch “ON” or “OFF” the selected LED from the webpage.

**Conclusion:** We have successfully executed the program of blinking LED remotely using Raspberry Pi.

**Code:**

import time

import RPi.GPIO as GPIO

RUNNING = True

HIGH = 1

LOW = 0

#Position LEDs

FL0\_LED\_POS = 10

FL1\_LED\_POS = 12

FL2\_LED\_POS = 13

FL3\_LED\_POS = 16

#CALL Position LEDs

FL0\_LED\_CALL = 6

FL1\_LED\_CALL = 7

FL2\_LED\_CALL = 8

FL3\_LED\_CALL = 9

#CALL Switch

FL0\_SW = 17

FL1\_SW = 18

FL2\_SW = 19

FL3\_SW = 20

#DIRECTION LED

LED\_D0 = 21

LED\_D1 = 22

LED\_D2 = 23

LED\_D3 = 24

LED\_D4 = 25

LED\_D5 = 26

LED\_D6 = 27

NO\_OF\_FLOORS = 4 # No of floors for Lift Simulation Operation

NO\_OF\_DIR\_LEDS = 7 # No of LEDs used for the lift direction (on Board)

DEFAULT\_LIFT\_POS = 0 # The floor no where lift is positioned when program is executed

DIR\_LED = [ LED\_D0,LED\_D1,LED\_D2,LED\_D3,LED\_D4,LED\_D5,LED\_D6]

FLOOR\_POS\_LED = [FL0\_LED\_POS,FL1\_LED\_POS,FL2\_LED\_POS,FL3\_LED\_POS]

FLOOR\_CALL\_LED =[FL0\_LED\_CALL,FL1\_LED\_CALL,FL2\_LED\_CALL,FL3\_LED\_CALL]

FLOOR\_SW =[FL0\_SW,FL1\_SW,FL2\_SW,FL3\_SW]

def InitElevator():

GPIO.setmode(GPIO.BCM)

GPIO.setup(FL0\_LED\_POS,GPIO.OUT)

GPIO.setup(FL1\_LED\_POS,GPIO.OUT)

GPIO.setup(FL2\_LED\_POS,GPIO.OUT)

GPIO.setup(FL3\_LED\_POS,GPIO.OUT)

GPIO.setup(FL0\_LED\_CALL,GPIO.OUT)

GPIO.setup(FL1\_LED\_CALL,GPIO.OUT)

GPIO.setup(FL2\_LED\_CALL,GPIO.OUT)

GPIO.setup(FL3\_LED\_CALL,GPIO.OUT)

GPIO.setup(LED\_D0,GPIO.OUT)

GPIO.setup(LED\_D1,GPIO.OUT)

GPIO.setup(LED\_D2,GPIO.OUT)

GPIO.setup(LED\_D3,GPIO.OUT)

GPIO.setup(LED\_D4,GPIO.OUT)

GPIO.setup(LED\_D5,GPIO.OUT)

GPIO.setup(LED\_D6,GPIO.OUT)

GPIO.setup(FL0\_SW,GPIO.IN,pull\_up\_down=GPIO.PUD\_UP)

GPIO.setup(FL1\_SW,GPIO.IN,pull\_up\_down=GPIO.PUD\_UP)

GPIO.setup(FL2\_SW,GPIO.IN,pull\_up\_down=GPIO.PUD\_UP)

GPIO.setup(FL3\_SW,GPIO.IN,pull\_up\_down=GPIO.PUD\_UP)

for i in range(0,4):

GPIO.output(FLOOR\_POS\_LED[i],LOW)

for i in range(0,4):

GPIO.output(FLOOR\_CALL\_LED[i],LOW)

for i in range(0,7):

GPIO.output(DIR\_LED[i],LOW)

return

def GoingUP():

for i in range(0,7):

GPIO.output(DIR\_LED[i],HIGH)

time.sleep(0.5)

for i in range(0,7):

GPIO.output(DIR\_LED[i],LOW)

return

def GoingDOWN():

for i in range(0,7):

GPIO.output(DIR\_LED[6-i],HIGH)

time.sleep(0.5)

for i in range(0,7):

GPIO.output(DIR\_LED[i],LOW)

return

def GetFloorCall():

while True:

for call\_sw in range(0,4):

input\_state = GPIO.input(FLOOR\_SW[call\_sw])

time.sleep(0.1)

if input\_state == 0:

return call\_sw

try:

print "\nLift Operation Simulation using Python\n"

print "-----------------------------------------------\n"

InitElevator()

cur\_flr = DEFAULT\_LIFT\_POS # Variable for current lift floor (initially 0)

while RUNNING:

GPIO.output(FLOOR\_POS\_LED[cur\_flr],HIGH)

new\_flr = GetFloorCall()

if new\_flr > cur\_flr: # if (new floor > current floor) means lift is called to upper floor

tmp = cur\_flr # store current floor no into tmp variable

GPIO.output(FLOOR\_CALL\_LED[new\_flr],HIGH)

print "LIFT going UP to floor #%d" %new\_flr # print destination floor

while (tmp != new\_flr): # Use tmp value (incremental); till it becomes destination

GoingUP() # Glow direction LEDs in upward direction

GPIO.output(FLOOR\_POS\_LED[tmp],LOW)

tmp += 1 # Increment tmp value by 1

GPIO.output(FLOOR\_POS\_LED[tmp],HIGH)

time.sleep(0.5) # Sleep for 0.5 second (500 ms)

elif new\_flr < cur\_flr: # if (new floor < current floor) means lift is called to lower floor

tmp = cur\_flr # store current floor no into tmp variable

GPIO.output(FLOOR\_CALL\_LED[new\_flr],HIGH)

GPIO.output(FLOOR\_CALL\_LED[cur\_flr],LOW)

print "LIFT going DOWN to floor #%d" %new\_flr # print destination floor

while (tmp != new\_flr): # Use tmp value (decremental); till it becomes destination

GoingDOWN() # Glow direction LEDs in downward direction

time.sleep(0.01) # Sleep for 10 ms

GPIO.output(FLOOR\_POS\_LED[tmp],LOW)

tmp -= 1 # Decrement tmp value by 1

GPIO.output(FLOOR\_POS\_LED[tmp],HIGH)

time.sleep(0.5) # sleep for 0.5 second (500 ms)

cur\_flr = new\_flr # Once lift reaches the destination; current floor points to destination floor no

GPIO.output(FLOOR\_CALL\_LED[cur\_flr],LOW)

time.sleep(0.1) # Sleep for 1 second

# If CTRL+C is pressed the main loop is broken

except KeyboardInterrupt:

RUNNING = False

print "\Stopping Elevator"

# Actions under 'finally' will always be called

finally:

# Stop and finish cleanly so the pins

# are available to be used again

GPIO.cleanup()