SRM Institute of Science and Technology

College of Engineering and Technology

Department of Electronics and Communication Engineering

18ECO109J-Embedded System Design Using Raspberry Pi
2022-23 (Even Semester)

Mini Project Report

Name	**: Harsh Yadav**
Register No.	**: RA2011003010746**
Day / Session	**: 1**
Venue	**: TP1317**
Project Title	**: Sending E-Mail using Raspberry Pi**
Lab Supervisor	**: SUGANTHI BRINDHA G**
Team Members	**: 1)Harsh Yadav (RA2011003010746)** **2)Kevin Pandya (RA2011003010745)** **3) Neel Gajjiwala (RA2011033010114)**

Particulars	**Max. Marks**	**Marks Obtained**
Objective & Description	05	
Algorithm,Flowchart,Program	20	
Demo verification	10	
Viva	10	
Report	05	
Total	**50**	

REPORT VERIFICATION

Date :

Staff Name :

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
COLLEGE OF ENGINEERING & TECHNOLOGY SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
(Under Section 3 of UGC Act, 1956)
S.R.M. NAGAR, KATTANKULATHUR

BONAFIDE CERTIFICATE

Register No: ______________________________

Certified to be the bonafide record of work done by ____of B.Tech. Degree course in the Practical **18ECO109J-Embedded System Design Using Raspberry Pi** in SRMIST, Kattankulathur during the academic year 2022-2023(EVEN SEM)

Staff In-Charge

Date:___________

Open Elective Co-Ordinator

…………………………………………………………………………………………

Submitted for University Examination held on ____at SRM Institute of Science and Technology, Kattankulathur.

Date:

Internal Examiner I

Internal Examiner II

Sending E-mail using raspberry pi

OBJECTIVE:

To send emails autonomously using raspberry pi as a primary platform.

ABSTRACT:

The project titled "Sending an Email via Raspberry Pi" aims to explore the capabilities of Raspberry Pi as a tool for sending emails. The project involves the use of a Raspberry Pi board connected to the internet and configured with the necessary software and libraries to send emails. The project includes a step-by-step guide on how to set up the Raspberry Pi to send emails, including configuring email credentials, setting up SMTP servers, and sending emails using Python code. The project also explores the potential use cases of sending emails via Raspberry Pi, including home automation, remote monitoring, and alert systems. The results of the project demonstrate the effectiveness of Raspberry Pi as a low-cost and versatile tool for sending emails and provide a useful resource for those looking to incorporate email functionality into their Raspberry Pi projects

HARDWARE / SOFTWARE REQUIRED:

- Raspberry Pi
- M/M and M/F jumper wires
- Breadboard
- 1µF Capacitor
- Input/Output devices

BLOCK DIAGRAM/CONNECTION DIAGRAM:

Raspberry Pi: The main computing platform for the project.

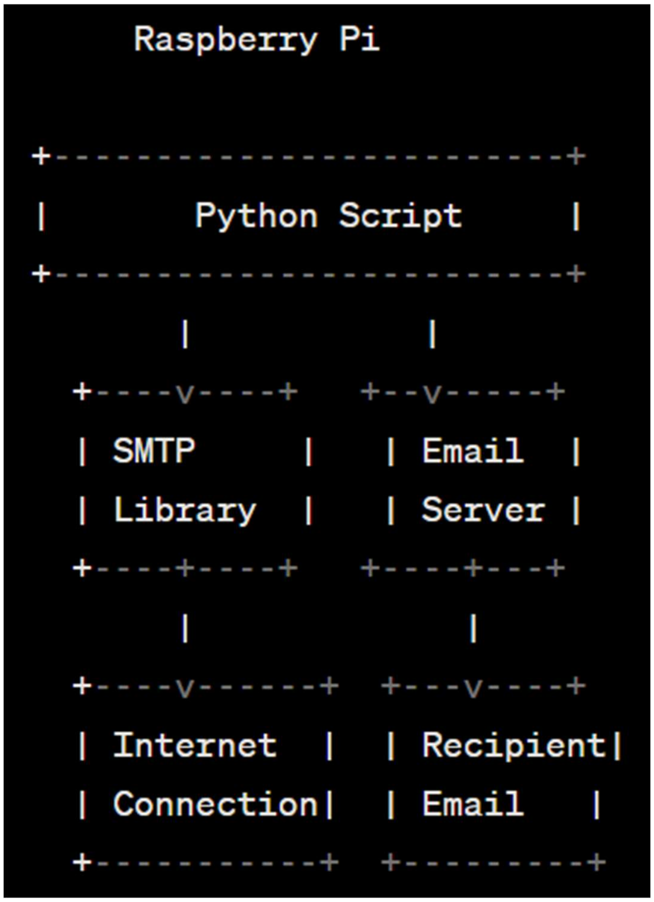
Python Script: A Python program that runs on the Raspberry Pi and sends emails.

SMTP Library: A Python library that provides functions to send emails over the Simple Mail Transfer Protocol (SMTP).

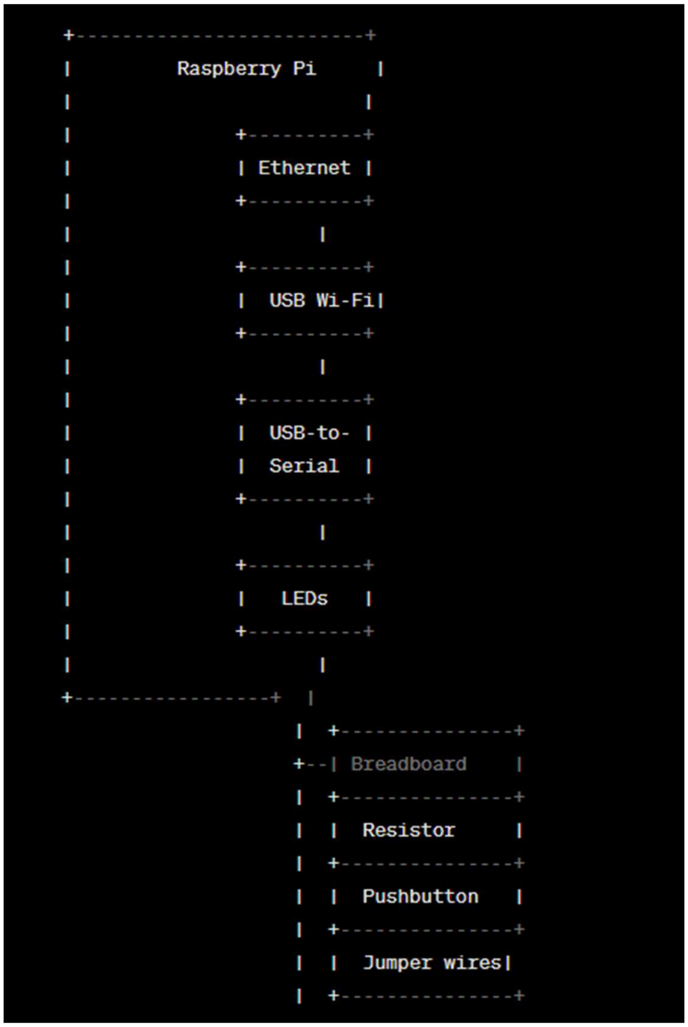
Email Server: The email server that is used to send emails. This can be a public email service like Gmail or a private email server.

Internet Connection: A connection to the internet is necessary to send emails.

Recipient Email: The email address of the recipient of the email.



```
     Raspberry Pi


+-------------------------+
|       Python Script     |
+-------------------------+
       |           |
  +----v----+   +--v-----+
  | SMTP     |   | Email  |
  | Library  |   | Server |
  +----+----+   +----+---+
       |             |
  +----v------+  +---v----+
  | Internet  |  | Recipient|
  | Connection|  | Email   |
  +-----------+  +---------+
```



```
+-------------------------+
|         Raspberry Pi     |
|                         |
|              +----------+
|              | Ethernet |
|              +----------+
|                    |
|              +----------+
|              |  USB Wi-Fi|
|              +----------+
|                    |
|              +----------+
|              |  USB-to- |
|              |  Serial  |
|              +----------+
|                    |
|              +----------+
|              |   LEDs   |
|              +----------+
|                    |
+-----------------+  |
                   |  +---------------+
                   +--| Breadboard    |
                   |  +---------------+
                   |  |  Resistor     |
                   |  +---------------+
                   |  |  Pushbutton   |
                   |  +---------------+
                   |  |  Jumper wires|
                   |  +---------------+
```

ALGORITHM:

Set up the Raspberry Pi with a Wi-Fi or Ethernet connection to access the internet.

Install the necessary software packages on the Raspberry Pi, including Python, the RPi.GPIO library for controlling the GPIO pins, and the smtplib library for sending email.

Connect the necessary hardware components to the Raspberry Pi, such as an LED and a pushbutton.

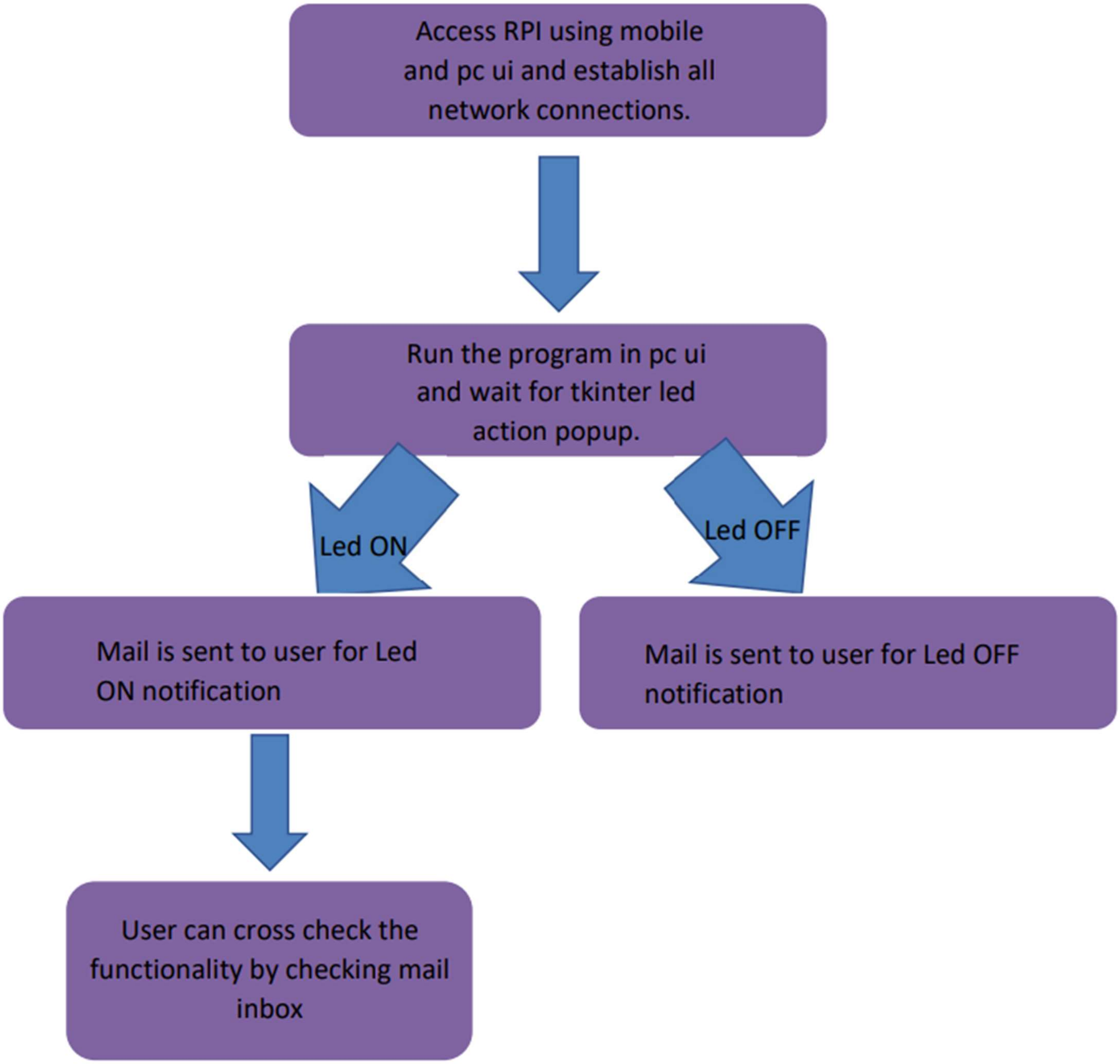
Write a Python script that reads the state of the pushbutton and sends an email when the button is pressed.

Configure the email settings in the Python script, including the SMTP server address, port number, and login credentials.

Write the code for sending an email using the smtplib library. This code should include the sender's email address, recipient's email address, subject line, and message body.

Set up the GPIO pins to control the LED and pushbutton using the RPi.GPIO library. This code should include the pin numbers and the input/output direction.

FLOW CHART:

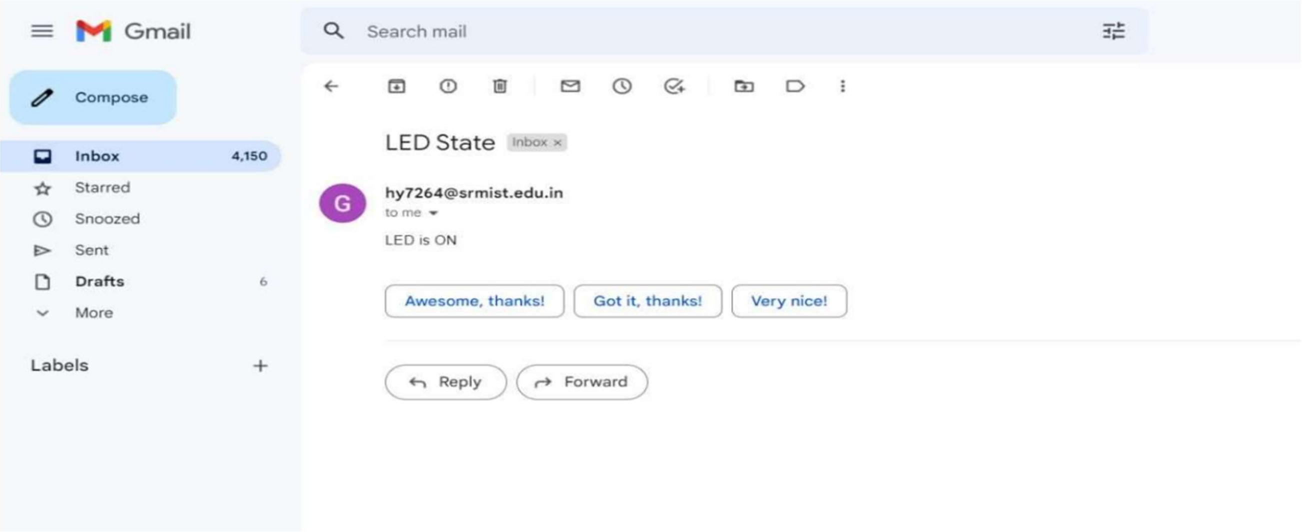

Access RPI using mobile and pc ui and establish all network connections.
Run the program in pc ui and wait for tkinter led action popup.
Led ON
Led OFF
Mail is sent to user for Led ON notification
Mail is sent to user for Led OFF notification
User can cross check the functionality by checking mail inbox

PROGRAM:

```
import tkinter as tk
import tkinter.font
import yagmail
from gpiozero import LED

win=tk.Tk()
win.title("IOT App")
myFont=tkinter.font.Font(family = 'Helvetica', size = 12, weight = "bold")

led1 = LED(21)
led2 = LED(26)
led3 = LED(24)
def send_mail(state):
	with open("/home/pi/.local/share/.email_password", "r") as f:
		password = f.read()

	yag = yagmail.SMTP('hy7264@srmist.edu.in',password)

	yag.send(to='hy717330@gmail.com',subject='LED State',contents=state)
	print('sent')

def ledToggle():
	if led1.is_lit:
		led1.off()
		ledButton["text"]="Turn LED ON"
		send_mail("LED is OFF")
	else:
		led1.on()
		ledButton["text"]="Turn LED OFF"
		send_mail("LED is ON")

def led2Toggle():
	#GPIO.output(26,True)
```

```
    if led2.is_lit:
        led2.off()
        ledButton2["text"]="Turn Yellow ON"
    else:
        led2.on()
        ledButton2["text"]="Turn Yellow OFF"

def led3Toggle():
    if led3.is_lit:
        led3.off()
        ledButton3["text"]="Turn White ON"
    else:
        led3.on()
        ledButton3["text"]="Turn White OFF"

def exitProgram():
    win.quit()

ledButton = tk.Button(win,text="Turn LED On", font=myFont, command=ledToggle, bg='bisque2',
height=1,width=24)
ledButton.grid(row=0,sticky=tk.NSEW)
#ledButton2 = tk.Button(win,text="Turn Yellow LED On", font=myFont, command=led2Toggle,
bg='bisque2', height=1,width=24)
#ledButton2.grid(row=1,sticky=tk.NSEW)
#ledButton3 = tk.Button(win,text="Turn White LED On", font=myFont, command=led3Toggle,
bg='bisque2', height=1,width=24)
#ledButton3.grid(row=2,sticky=tk.NSEW)
exitButton=tk.Button(win,text="Exit", font=myFont,command=exitProgram,bg='cyan',height=1,width=6)
exitButton.grid(row=3,sticky=tk.E)

tk.mainloop()
```

OUTPUT:

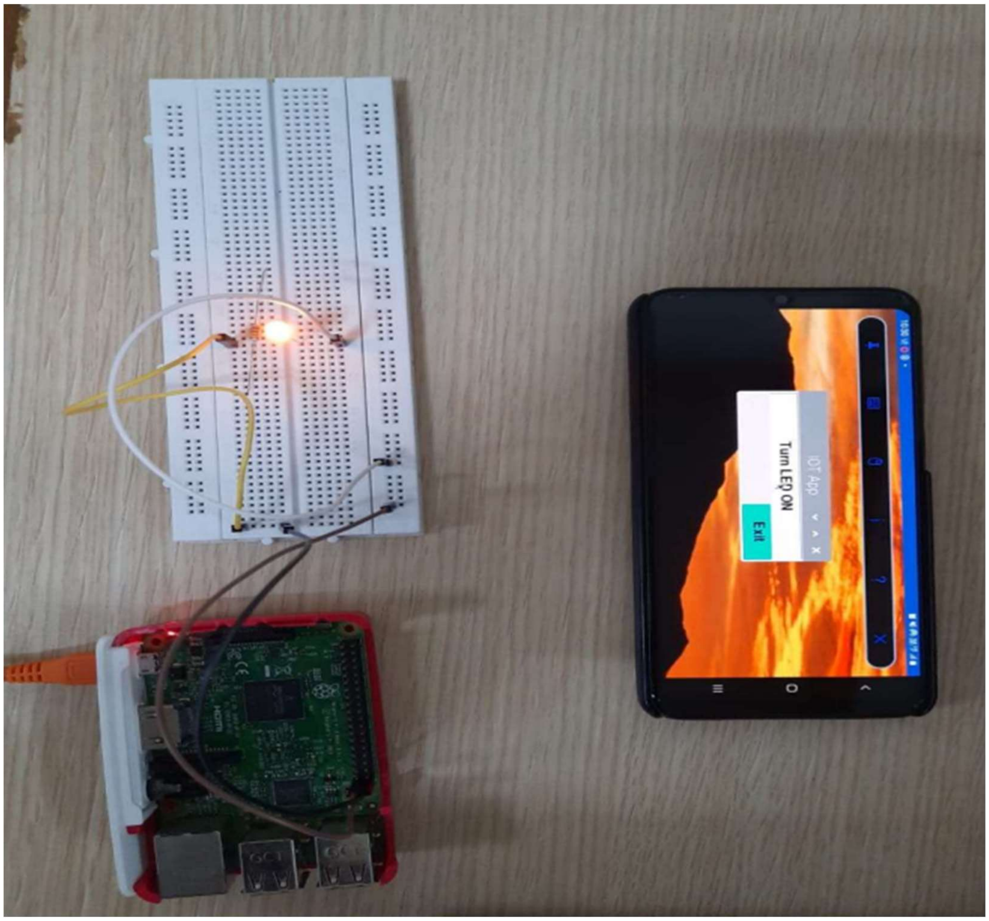


LED State

hy7264@srmist.edu.in

to me

LED is ON



REAL TIME CONSTRAINTS:

Network Connectivity: The Raspberry Pi must have a reliable and stable network connection in order to send emails. This means that the network configuration should be properly set up and the Pi should have access to the internet.

Latency: Emails need to be sent in a timely manner. The time it takes to send an email can be affected by factors such as network latency, server response times, and email size. The program should be designed to handle these factors and ensure that emails are sent in a timely manner.

Power Supply: The Raspberry Pi is a low-power device and requires a stable power supply. Any fluctuations or interruptions in power can cause the device to crash or malfunction. It is important to ensure that the Pi is connected to a stable power source throughout the project.

Memory and Processing Power: The Raspberry Pi has limited memory and processing power compared to a regular computer. This means that the program should be designed to be lightweight and efficient, in order to run smoothly and not overload the Pi's resources.

Security: Email contains sensitive information and it is important to ensure that the program is secure and that the emails are sent over a secure connection. The program should be designed to encrypt emails and authenticate users to prevent unauthorized access.

CONCLUSION:

Hence we came to a endpoint project which successfully sends the email using raspberry pi.

References:

www.openai.com

www.geeksforgeeks.com

www.raspberrypi.com