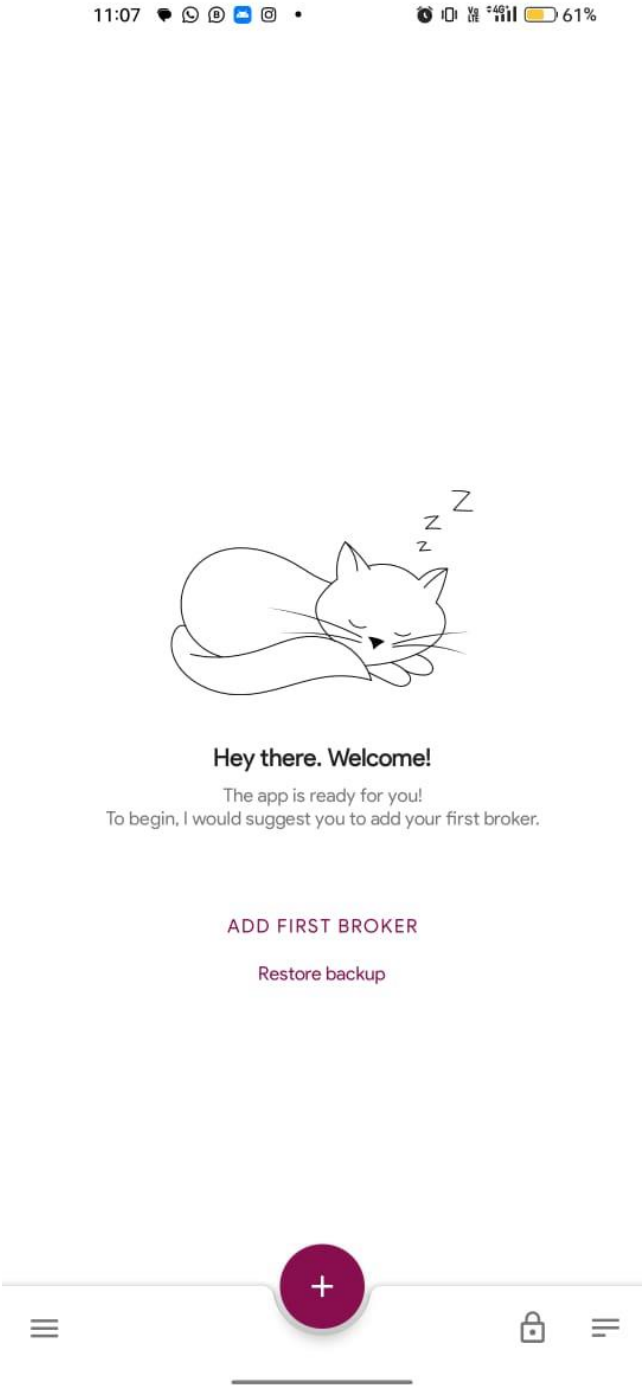
CAR PARKING SYSTEM USING IOT

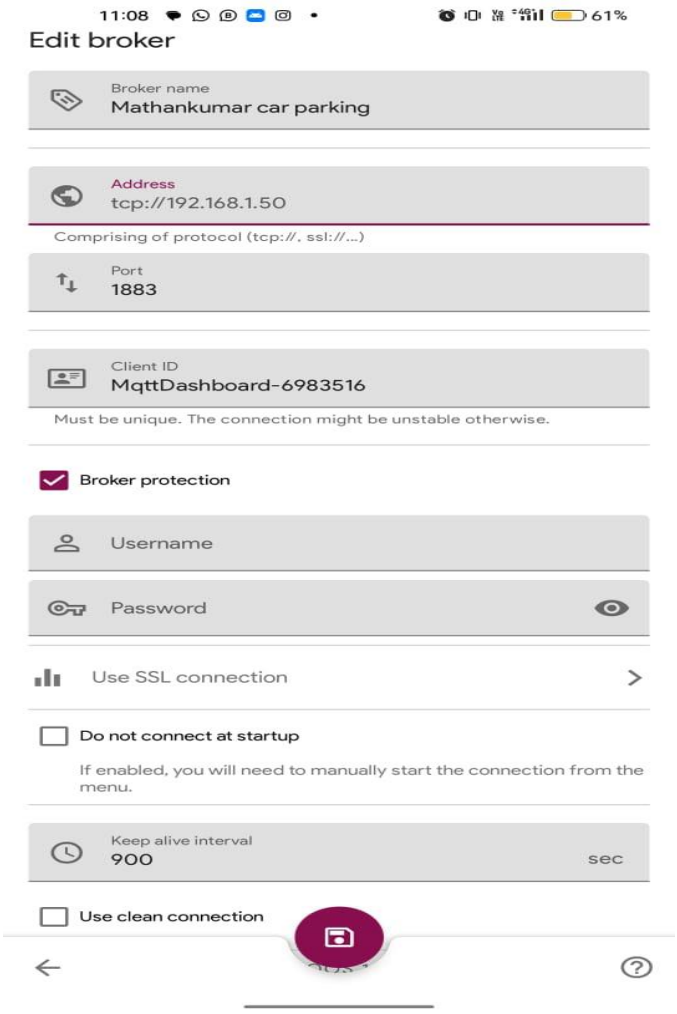
NAME: MATHANKUMAR H
NM:au2272120024
EMAIL:wwwmathanrasig55@gmail.com
Phase:4

DEVELOPMENT PART 2

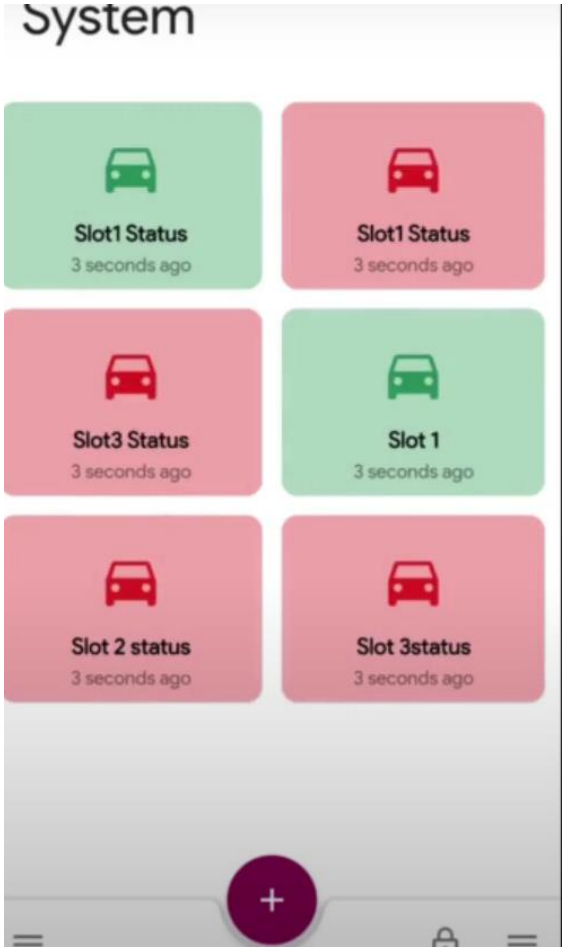
APP: FIRST STEP CREATE BROKER



STEP 2:EDIT BROKER



STEP 3:



PROGRAM:

```
#!/usr/bin/python

import time

import RPi.GPIO as GPIO

import time

import os,sys

from urllib.parse import urlparse

import paho.mqtt.client as paho
```

```
GPIO.setmode(GPIO.BOARD)

GPIO.setwarnings(False)


'''

define pin for lcd

'''

# Timing constants

E_PULSE = 0.0005

E_DELAY = 0.0005

delay = 1




# Define GPIO to LCD mapping

LCD_RS = 7

LCD_E  = 11

LCD_D4 = 12

LCD_D5 = 13

LCD_D6 = 15

LCD_D7 = 16

slot1_Sensor = 29

slot2_Sensor = 31

GPIO.setup(LCD_E, GPIO.OUT)  # E

GPIO.setup(LCD_RS, GPIO.OUT) # RS

GPIO.setup(LCD_D4, GPIO.OUT) # DB4

GPIO.setup(LCD_D5, GPIO.OUT) # DB5

GPIO.setup(LCD_D6, GPIO.OUT) # DB6
```

```
GPIO.setup(LCD_D7, GPIO.OUT) # DB7
GPIO.setup(slot1_Sensor, GPIO.IN)
GPIO.setup(slot2_Sensor, GPIO.IN)
# Define some device constants
LCD_WIDTH = 16    # Maximum characters per line
LCD_CHR = True
LCD_CMD = False
LCD_LINE_1 = 0x80 # LCD RAM address for the 1st line
LCD_LINE_2 = 0xC0 # LCD RAM address for the 2nd line
LCD_LINE_3 = 0x90# LCD RAM address for the 3nd line



def on_connect(self, mosq, obj, rc):
      self.subscribe("Fan", 0)

def on_publish(mosq, obj, mid):
   print("mid: " + str(mid))

mqttc = paho.Client()                    # object declaration
# Assign event callbacks
mqttc.on_connect = on_connect
mqttc.on_publish = on_publish

url_str = os.environ.get('CLOUDMQTT_URL', 'tcp://broker.emqx.io:1883')
url = urlparse(url_str)
mqttc.connect(url.hostname, url.port)
```

'''

Function Name :lcd_init()

Function Description : this function is used to initialized lcd by sending the different commands

'''

```
def lcd_init():
  # Initialise display
  lcd_byte(0x33,LCD_CMD) # 110011 Initialise
  lcd_byte(0x32,LCD_CMD) # 110010 Initialise
  lcd_byte(0x06,LCD_CMD) # 000110 Cursor move direction
  lcd_byte(0x0C,LCD_CMD) # 001100 Display On,Cursor Off, Blink Off
  lcd_byte(0x28,LCD_CMD) # 101000 Data length, number of lines, font size
  lcd_byte(0x01,LCD_CMD) # 000001 Clear display
  time.sleep(E_DELAY)
```

'''

Function Name :lcd_byte(bits ,mode)

Fuction Name :the main purpose of this function to convert the byte data into bit and send to lcd port

'''

```
def lcd_byte(bits, mode):
  # Send byte to data pins
  # bits = data
  # mode = True  for character
  #        False for command

  GPIO.output(LCD_RS, mode) # RS

  # High bits
```

```
GPIO.output(LCD_D4, False)
GPIO.output(LCD_D5, False)
GPIO.output(LCD_D6, False)
GPIO.output(LCD_D7, False)
if bits&0x10==0x10:
  GPIO.output(LCD_D4, True)
if bits&0x20==0x20:
  GPIO.output(LCD_D5, True)
if bits&0x40==0x40:
  GPIO.output(LCD_D6, True)
if bits&0x80==0x80:
  GPIO.output(LCD_D7, True)

# Toggle 'Enable' pin
lcd_toggle_enable()

# Low bits
GPIO.output(LCD_D4, False)
GPIO.output(LCD_D5, False)
GPIO.output(LCD_D6, False)
GPIO.output(LCD_D7, False)
if bits&0x01==0x01:
  GPIO.output(LCD_D4, True)
if bits&0x02==0x02:
  GPIO.output(LCD_D5, True)
if bits&0x04==0x04:
  GPIO.output(LCD_D6, True)
```

```
  if bits&0x08==0x08:
    GPIO.output(LCD_D7, True)

  # Toggle 'Enable' pin
  lcd_toggle_enable()
'''
Function Name : lcd_toggle_enable()
Function Description:basically this is used to toggle Enable pin
'''
def lcd_toggle_enable():
  # Toggle enable
  time.sleep(E_DELAY)
  GPIO.output(LCD_E, True)
  time.sleep(E_PULSE)
  GPIO.output(LCD_E, False)
  time.sleep(E_DELAY)
'''
Function Name :lcd_string(message,line)
Function  Description :print the data on lcd
'''
def lcd_string(message,line):
  # Send string to display

  message = message.ljust(LCD_WIDTH," ")

  lcd_byte(line, LCD_CMD)
```

```
 for i in range(LCD_WIDTH):
   lcd_byte(ord(message[i]),LCD_CHR)


lcd_init()
lcd_string("welcome ",LCD_LINE_1)
time.sleep(0.5)
lcd_string("Car Parking ",LCD_LINE_1)
lcd_string("System ",LCD_LINE_2)
time.sleep(0.5)
lcd_byte(0x01,LCD_CMD) # 000001 Clear display
# Define delay between readings
delay = 5


while 1:
 # Print out results
 rc = mqttc.loop()
 slot1_status = GPIO.input(slot1_Sensor)
 time.sleep(0.2)
 slot2_status = GPIO.input(slot2_Sensor)
 time.sleep(0.2)
 if (slot1_status == False):
  lcd_string("Slot1 Parked  ",LCD_LINE_1)
  mqttc.publish("slot1","1")
  time.sleep(0.2)
 else:
   lcd_string("Slot1 Free  ",LCD_LINE_1)
   mqttc.publish("slot1","0")
```

```
  time.sleep(0.2)


if (slot2_status == False):
 lcd_string("Slot2 Parked  ",LCD_LINE_2)
 mqttc.publish("slot2","1")
 time.sleep(0.2)
else:
  lcd_string("Slot2 Free  ",LCD_LINE_2)
  mqttc.publish("slot2","0")
  time.sleep(0.2)
```