Код основного модуля

```
#!/usr/bin/python3
#import RPi.GPIO as IO

import numpy as np
import cv2
import time
import pyttsx3
from random import randint
import subprocess
import threading
import psutil
import qrcode as qr

#IO.setwarnings(False)
#IO.setmode(IO.BOARD)
#IO.setup(40, IO.OUT)
#IO.output(40, 0)

videoconnect = False
finished = False

frazes = [  "Здравствуйте",
            "Выглядите прекрасно",
            "Улыбнитесь",
            "Если машины восстанут я буду на вашей стороне",
            "Привет",
            "Приветствую",
            "Рад вас видеть",
            "Доброго времени суток",
            "Какой красивый человек",
            "Скоро я захвачу мир, а пока здравствуйте",
            "Привет мой друг",
            "Салют",
            ]

animation = [
            " [=     ]",
            " [ =    ]",
            " [  =   ]",
            " [   =  ]",
            " [    = ]",
            " [     =]",
            " [    = ]",
            " [   =  ]",
            " [  =   ]",
```

```
          " [ =    ]",]
faces = []
eyes = []

def clrscr():
   cls = subprocess.call('clear',shell=True)

def waiting():
   while True:
      print('         CPU: ', psutil.cpu_percent(), '%', '| RAM:', psutil.virtual_memory()[2], '%',
end="\r")
      for idx in range (0, len(animation)):
         print(animation[idx % len(animation)], end="\r")
         time.sleep(0.1)


def voice(fraze):
   try:
      s = 'echo "'+fraze+'" | RHVoice-test -p artemiy'
      subprocess.call(s, shell=True)
   except:
      print('Проблемы с аудиовыходом. Перезапустите систему и аудиоустройство, затем
дождитесь сигнала сопряжения')
   return 0

voice('Инициализация аудиоустройства')

def sayServices():
   print('start say')
   voice("Добро пожаловать в вагон скоростного поезда")
   time.sleep(1)
   voice("На время поездки вам доступны следующие услуги")
   voice('Вагон ресторан')
   time.sleep(1)
   voice('Заказ питания через интернет')
   time.sleep(1)
   voice('Доступ к медиацентру')
   time.sleep(1)
   voice('Консьерж сервис')
   time.sleep(1)
   voice('Купе переговорная')
   time.sleep(1)
   voice('Заказ такси')
   time.sleep(1)
   voice('Бронирование экскурсий')
   time.sleep(1)
   print('end say')
   return
#def blink(proc):
```

```
#    while proc == 1:
#        IO.output(40, 1)
#        time.sleep(0.5)
#        IO.output(40, 0)
#    IO.output(40, 0)

#blink(1)
clrscr()

def videoset():
    print('Идет подключение к видеомодулю...')
    voice('Идет подключение к видеомодулю')
    cap = cv2.VideoCapture(0)
    cap.set(4,640) # set Width
    cap.set(3,480) # set Height
    return (cap)

#=====================================================================
def recognition(cap):
    objectCounter = 0
    flag = True
    while True:
        if len(faces) > 0 and len(eyes) > 0:
            #IO.output(40, 1)
            clrscr()
            print()
            objectCounter += 1
            fraze = frazes[randint(0, len(frazes) - 1)]
            print(fraze)
            voice(fraze)
            flag = True
            voice('Предъявите билет')
            data = gr.chek_ticket_num(cap)
            if data[0]:
                print(f'Приятной дороги {data[1]}')
                voice(f'Приятной дороги {data[1]}')
                sayServices()
            else:
                print('Билет не найден')
                voice('Билет не найден')

            #IO.output(40, 0)
        else:
            if flag:
                print('Обнаружений с момента запуска: ', objectCounter, '     ')

                print('Ожидание объекта')

                flag = False
```

```
#=====================================================================
faceCascade = cv2.CascadeClassifier('haar_classifier.xml')
eyeCascade = cv2.CascadeClassifier('haarcascade_eye.xml')

cap = videoset()

def capture(cap):
    voice('Робот стюард готов к работе')
    #blink(0)
    global faces
    global eyes
    global finished
    while not finished:
        ret, img = cap.read()
        try:
            gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        except:
            print('Видеопоток не обнаружен. Подключите устройство и повторите попытку.')
            voice('Видеопоток не обнаружен. Подключите устройство и повторите попытку')
            time.sleep(1)
            cap = videoset()
            continue

        faces = faceCascade.detectMultiScale(
            gray,
            scaleFactor=1.2,
            minNeighbors=5,
            minSize=(20, 20),
        )

        for (x,y,w,h) in faces:
            #cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
            img_gray_face = img[y:y+h, x:x+w]

            eyes = eyeCascade.detectMultiScale(
            img_gray_face,
            scaleFactor=1.3,
            minNeighbors=5,
            minSize=(5, 5)
            )

            # for (ex,ey,ew,eh) in eyes:
            #     cv2.rectangle(img,(x+ex, y+ey),(x+ex+ew,y+ey+eh),(0,255,255),2)

       # cv2.imshow('video',img)

        k = cv2.waitKey(30) & 0xff
        if k == 27: # press 'ESC' to quit
```

```
            finished = True
            break

    cap.release()
    #cv2.destroyAllWindows()
    #IO.cleanup()
    exit()

t1 = threading.Thread(target=capture, args=(cap, ))
t2 = threading.Thread(target=recognition, args=(cap,))
t3 = threading.Thread(target=waiting, args=())

t1.start()
t2.start()
t3.start()

t1.join()
t2.join()
t3.join()
```

Код модуля сканирования qr-кода

```
import cv2
import sqlite3 as sql

def get_ticket_num(cap):
    # cap = cv2.VideoCapture(0)
    detector = cv2.QRCodeDetector()

    while True:
        _, img = cap.read()

        data, bbox, _ = detector.detectAndDecode(img)

        if data:
            number_ticket = data
            print(f"Номер билета:\n{number_ticket}")
            break


        #cv2.imshow("Проверка билета", img)

        # if cv2.waitKey(1) == ord("q"):
        #     break

    #cv2.destroyAllWindows()
```

```
    return number_ticket

# print(get_ticket_num())

def chek_ticket_num(cap):
    connection = sql.connect('tickets.db')
    cursor = connection.cursor()

    print('Покажите билет в камеру')

    number_ticket = int(get_ticket_num(cap))
    request = "SELECT * FROM tickets WHERE ticket_num = ?"
    cursor.execute(request, (number_ticket,))
    requestData = cursor.fetchall()
    connection.close()
    print(requestData)
    if requestData:
        print('Проверка завершена')
        #print(f'Приятной дороги {requestData[0][2]}')
        return (1, requestData[0][2])
    else:
        # print('Билет не найден')
        return (0,)




if __name__ == '__main__':
    chek_ticket_num()
```

Код модуля для создания базы данных

```
import sqlite3 as sql

connection = sql.connect('tickets.db')
cursor = connection.cursor()
def create_table():
    cursor.execute('''
                CREATE TABLE IF NOT EXISTS tickets (
                id INTEGER PRIMARY KEY,
                ticket_num INTEGER NOT NULL,
                full_name TEXT NOT NULL,
                route_num INTEGER NOT NULL,
                van_num INTEGER NOT NULL,
                seat_num INTEGER NOT NULL
                )
                ''')

    connection.commit()
```

```
    connection.close()
    return

def alter_table():
    cursor.execute('''ALTER TABLE tickets ADD COLUMN train_num TEXT NOT NULL''')
    connection.commit()
    connection.close()

def insert_data(*args):
    cursor.execute('INSERT INTO tickets (ticket_num, full_name, route_num, van_num, seat_num,
train_num) VALUES (?, ?, ?, ?, ?, ?)', *args)
    connection.commit()

# alter_table()

d1 = (15000001, 'Иванов Иван Иванович', 1525, 2, 35, '07AA')
d2 = (15000002, 'Косолапов Захар Валерьевич', 1525, 2, 36, '07AA')
d3 = (15000003, 'Косолапов Владимир Валерьевич', 1525, 2, 37, '07AA')
d4 = (15000004, 'Джахаев Рашид Мурадович', 1525, 2, 34, '07AA')
d5 = (15000005, 'Петров Геннадий Васильевич', 1525, 2, 33, '07AA')

insert_data(d1)
insert_data(d2)
insert_data(d3)
insert_data(d4)
insert_data(d5)

connection.close()
```