**7. óra: Kamera streaming**

1. 5 perces beszámoló tetszőleges tartalommal és üzenettel valamint tetszőlegesen választott retorikai eszközökkel. Nekünk kell eldönteni, hogy mi lehetett az üzenet. Figyelni kell a beszéd felépítésére (bevezető, fő rész és konklúzió) és arra, hogy a tartalom minden eleme támogassa az üzenetet, illetve, hogy használjunk retorikai eszközöket.
2. 3 perces véleményalkotás  
   - bevezető, fő rész, konklúzió  
   - 2 pozitív és egy konstruktív javaslat  
   - konklúzióban összefoglalni mi lehetett az üzenet, volt-e retorikai eszköz használva és adni még egy pozitív megjegyzést.
3. Távirányítású autó készítés:
   1. *Bevezető*:  
      Manapság már nagyon sok olyan honlap létezik, ahol élőben nézhetünk kamera képeket valamiről, pl. valaki közvetíti ahogy alszik, Abbey road-i átkelőhely élő képe (ahol a Beatles sétált), vagy akár ürrakéták kilövése. Egy sokkal hétköznapiasabb dologra is lehet használni, lakásunk/házunk megfigyelése távolból. Léteznek térfigyelő kamerák, amivel az ingatlanunk biztonságát erősíthetjük, de akár mi is megcsinálhatjuk ezt egy egyszerű webcamerával. A Raspberry Pi-on keresztül élőben közvetíthetjük a megfigyelt terület képét egy IP címre, amit aztán bármilyen internetet elérő eszközről, pl. mobil telefon, megnézhetünk.
   2. *Előkészületek*:  
      - Elkészíteni a kamerát
   3. *Részletekre bontás:*

- Elkészíteni az áramkört.

- Beimportálni a csomagokat amik segítik a munkánkat: cv2, flask.

- Definiálni egy flask applikációt.

- Definiálni az index oldalt a streaming honlaphoz.

- Megírni az indexhez tartozó HTML kódot.

- Definiálni függvényeket a kamera kép megjelenítésére.

* 1. *Elektronika:*  
     - Legyen a webcam előkészítve.
  2. *Kód:*  
     - mutassuk be mik azok a generátorok.  
     [x+2 for x in [1,2,3,4,5]]  
     - Mutassuk be mik azok az iterátorok:  
     def my\_iterator():

yield 'First iteration'

yield 'Second iteration'

yield 'Third iteration'  
a = my\_iterator()

next(a)  
for item in my\_iterator():

print(item)

- Vegyük át a html alapjait:  
  
<html>

<head>

<title>Video Streaming Demonstration</title>

</head>

<body>

<h1>Hello, World!</h1>

</body>

</html>  
  
- Készítsünk egyszerű honlapot *Flask*kal:  
  
from flask import Flask, render\_template, Response

import cv2

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

# rendering webpage

return render\_template('index.html')

if \_\_name\_\_ == '\_\_main\_\_':

# defining server ip address and port

app.run(host='0.0.0.0',port='5000', debug=True)  
  
- Mutassuk be, hogy lehet pythonból változó értéket bevinni a html felületre jinja segítségével:  
  
<html>

<head>

<title>Video Streaming Demonstration</title>

</head>

<body>

<h1>Hello, {{value}}!</h1>

</body>

</html>

from flask import Flask, render\_template, Response

import cv2

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

# rendering webpage

Name = ”Raspi”

return render\_template('index.html', value=Name)

if \_\_name\_\_ == '\_\_main\_\_':

# defining server ip address and port

app.run(host='0.0.0.0',port='5000', debug=True)

* 1. *Tesztelés:*  
     - Teszteljük a megírt kód működését.
  2. *Fejlesztés:*  
     - Az opencv könyvtárat felhasználva módosítsuk a kamera képét (pl. szürke kép, filterelés stb.) és azt jelenítsük meg a honlapon.

- Adaptáljuk a streamingelős kódunkat úgy, hogy egyben mozgásérzékelő is legyen. Ha a képen mozgás történik, akkor a honlapon jelenjen meg egy felirat erről.

Kód minta:

from flask import Flask, render\_template, Response

import cv2

app = Flask(\_\_name\_\_)

#print(cap.isOpened())

@app.route('/')

def index():

    # rendering webpage

    return render\_template('index.html')

def gen(camera):

    while True:

        #get camera frame

        ret, frame = # olvasd ki a kepet a camera objektumbol

        ret, frame = cv2.imencode('.jpg', frame)

        frame = frame.tobytes()

        yield (b'--frame\r\n'

               b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n\r\n')

@app.route('/video\_feed')

def video\_feed():

    cap = # inicializald a camerat, ha linux alatt vagy, akkor '/dev/video0' a bemeno parameter

    cap.open(0)

    return Response(gen(cap),

                    mimetype='multipart/x-mixed-replace; boundary=frame')

if \_\_name\_\_ == '\_\_main\_\_':

    # defining server ip address and port

    app.run(host='0.0.0.0',port='5000', debug=True)

<html>

  <head>

    <title>Video Streaming Demonstration</title>

  </head>

  <body>

    <h1>Video Streaming Demonstration</h1>

    <img id="bg" src="{{ url\_for('video\_feed') }}">

  </body>

</html>