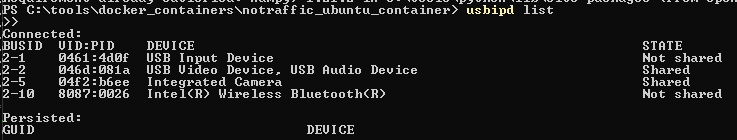
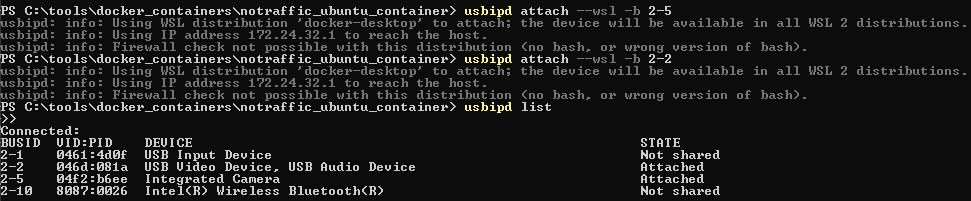
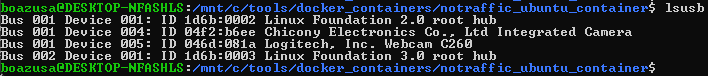
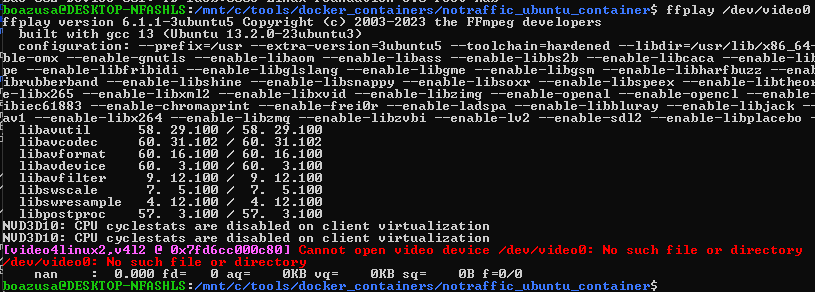
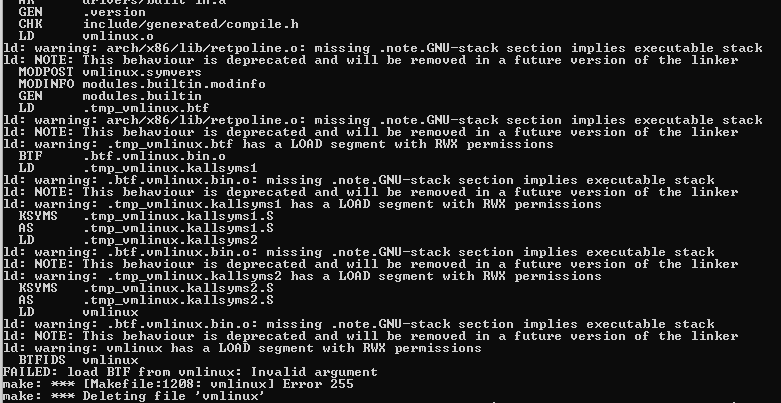
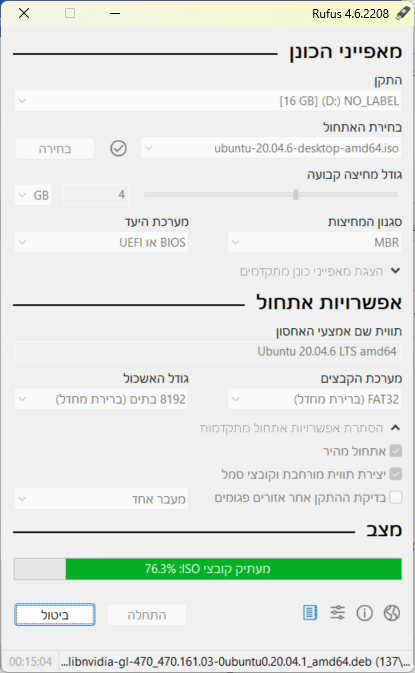
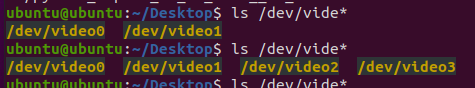
Technological steps tried to access cameras from wsl on windows machine:

1. usbipd & lsusb bind method
   1. Installed 'usbipd-win-4.4.0' on windows to enable cameras to be viewed in wsl (Windows Subsystem for Linux).
      1. Attached and shared them on windows  
         Before:  
           
         bind:  
         

attach (final step):  


* 1. Installed 'lsusb' on wsl.
     1. Verify they are viewed in wsl:  
        
  2. However, this is not enough for having the cameras viewed as '/dev/video\*'  
     

1. Modifying the wsl kernel to startup the cameras drivers within the wsl startup.
   1. followed the ['Linux Tips - Record Video from USB Camera in WSL (2022)](https://www.youtube.com/watch?v=t_YnACEPmrM)' tutorial, which seems to work on the video, but it failed at the vmlinux image.  
      failed at:  
      
   2. the result was that no '/dev/video\*' showed up as well.
2. Running windows script from wsl workaround.
   1. Installing 'opencv-python' for cameras control
   2. Launching cmd (windows command-line) inside wsl, using the '/mnt/c/Windows/System32/cmd.exe /C' command, enabled powering up the camera.
      1. Cons: it is not really ubuntu script that runs the cameras, it is 'Ubuntu powered windows-python'.
      2. Cannot be ran from inside a container that is built in wsl (directly from wsl)  
         
3. Run Ubuntu from USB (without installation, on my Windows PC).
   1. Download Ubuntu 20.04 from <https://releases.ubuntu.com/20.04/ubuntu-20.04.6-desktop-amd64.iso>
   2. Download rufus from <https://rufus.ie/he/>
   3. Create bootable Ubuntu 20.04 USB
   4. Reboot the PC while pressing the F12 key while startup.
      1. Select "Try Ubuntu" option
   5. Install Python and testing libraries (pytest, v4l2-ctl)  
      
   6. Ubuntu could access my built-in and external webcams  
        
      (/dev/video0 and /dev/video2)
   7. Issues encountered with ‘Try Ubuntu’ on my Windows machine:

* OS freezes within less than 3 hours, and work must be saved periodically, otherwise, it gets lost.
* Each time starting Ubuntu trial version, it requires installation of all apps (vlc, pycharm) and ‘sudo’ apt/snap’ installations and updates; takes ~15 minutes per initialization.
* Ubuntu trial version allocates only 4Gb of memory. Also I was managed to find how to save files (after couple times I lost my work due to OS freeze), installation of required tools couldn’t be achieved, such as:
  + Docker (size)
  + Ubuntu packages:
    - ffprobe (image/video detailed information).
    - fswebcam
    - gstreamer
* videos captured with Ubuntu v4l2-ctl commands could only be played via Ubuntu vlc (not with the default player, and neither in windows; only the screenshots could).

**Shall be implemented:**

1. Python Pytest with more details.
   1. Create tests with '@pytest.mark.parametrize()' such that:
      1. Each parameter/value has separate test
      2. Results can be understood clearer
   2. Create '@pytest.fixture' (more than one) such that same videos/screenshots can be tested for different features [within different tests] instead of creating video or image for each test.
   3. Install pytest-html and generate pytest html reports, which are more readable and user-friendly than plain text ones.
   4. Parse output results and verify the against test requirements (assert results in pytest).
2. Create more tests.
   1. Further system and requirements investigation could lead for additional tests.
   2. Details from v4l2-ctl, commands such as "v4l2-ctl --device=/dev/video0 –all", I generated a lot of information that can be verified against specs.
3. Create Ubuntu 20.04 Docker container that includes all the required SW and dependencies, and can run independently (without host machine installation).

Additional (out of scope for this assignment)

1. Rebuild docker image periodically (daily)
2. Create Jenkins pipeline that execute the tests every day
3. Push/pull the test code to a git repository