# Research software possibilities

Shihaab Rouine
Semester six

# Individual project research: software possibilities

To create the testable prototype described in the project plan, I need suitable software to scan and detect faces and emotions. My first thought is using C# as I have experience making C# applications. However I will research other languages as well.
I have never made a live emotion detection application, so having some sort of example code to work form is preferable.

## C#

When researching for C# emotion detection, Microsoft documentation popped up. Azure has 'Vision API', essentially a very powerful api that can detect emotions quickly and also from a live video feed. This sounds exactly like what I need for the prototype. To use the api you need the credentials, these can be claimed by a free account. However, a credit card is required to create this 'free' account, which I don't have. — paywall on azure

## Python

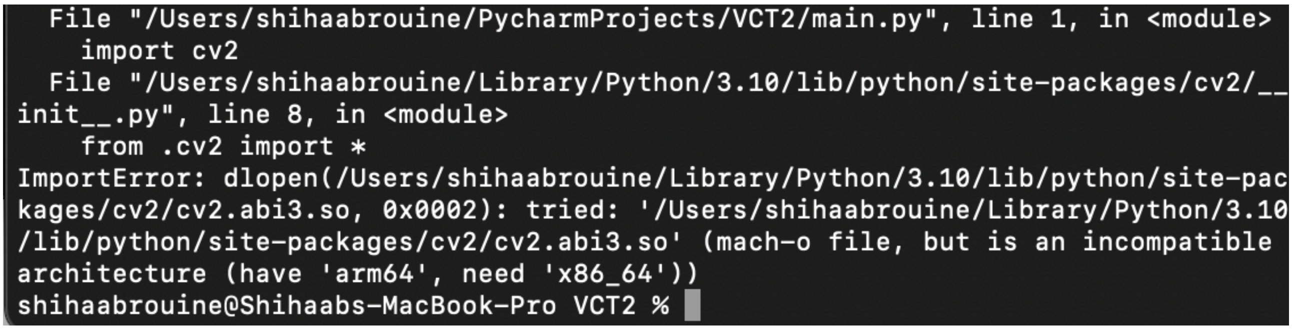
After a simple Google search of "create emotion detection" I quickly stumbled across Python. Python is a well known and high-level programming language that is can be used broadly.

Python is a valid option for the reason that there are is a lot of documentation of it on the internet.

## Experimenting possibilities

I found many examples using 'DeepFace', a framework that combines multiple high end frameworks to analyse facial attributes. All that is required to use the framework is an instalment via the Terminal. Installing the framework should have been simple and quick, however I faced a number of errors. After hours of Googling I understand that the issue is the M1 chip in my MacBook. Older MacBooks use an Intel chip as most Windows computers, these use a *x86* or *x64 architecture*. The M1 chip uses the *arm64* architecture. To work around this I found a tutorial on how to use Python on the M1 chip but mimicking the *x86 architecture* with *Rosseta, "*Rosetta is a dynamic binary translator developed by Apple Inc. for macOS, an application compatibility layer between different instruction set architectures." **Wikipedia**.

This led me to a new error, the DeepFace framework has been installed. But when running the example code, that tries to import the OpenCV framework that is required, "…`incompatible architecture (have 'arm64', need 'x86_64'))`".



```
  File "/Users/shihaabrouine/PycharmProjects/VCT2/main.py", line 1, in <module>
    import cv2
  File "/Users/shihaabrouine/Library/Python/3.10/lib/python/site-packages/cv2/__
init__.py", line 8, in <module>
    from .cv2 import *
ImportError: dlopen(/Users/shihaabrouine/Library/Python/3.10/lib/python/site-pac
kages/cv2/cv2.abi3.so, 0x0002): tried: '/Users/shihaabrouine/Library/Python/3.10
/lib/python/site-packages/cv2/cv2.abi3.so' (mach-o file, but is an incompatible
architecture (have 'arm64', need 'x86_64'))
shihaabrouine@Shihaabs-MacBook-Pro VCT2 % 
```

My computer at home uses a *x86 architecture*, so I switched to that platform for this project. After trying this on my computer, I also got al lot of errors regarding the char length of certain imported files. I tried to fix this but I did not find a solution.

I tried an older MacBook that still uses an Intel processor. The frameworks installed successfully, but I then found out that the code I was using was erroneous. By reading the documentation of

DeepFace I then made my own code. This worked and I managed to find a XML file that would translate the emotion that is detected to something paste-able on the screen. The recording of this prototype test isn't clear and the lighting seems darker than it was. **Recording DeepFace Prototype**. The prototype is slow and the displayed emotions are not always correct. But the conditions were not perfect either so we should keep in mind that the laptop was at <10% battery, the lighting was not great and it was an at least five years old MacBook Air.

# Conclusion

Python is the way to go for this project, combined with the powerful open source facial analytic framework DeepFace.

I initially started looking at software possibilities however, I learned more about the hardware requirements of the application. Like the needed processor architecture and the processing power of the computer.